Proceedings of the International Multiconference on Computer Science and Information Technology



Proceedings of the International Multiconference on Computer Science and Information Technology

Volume 2 (2007)

M. Ganzha, M. Paprzycki, T. Pełech-Pilichowski (editors)

ISSN 1896-7094

Polskie Towarzystwo Informatyczne
Oddział Górnośląski
ul. Raciborska 3
40-074 Katowice

TEXnical editor: Alexander Denisjuk

Proceedings of the International Multiconference on Computer Science and Information Technology

October 15 – 17, 2007. Wisła, Poland



PTI PROKOM SOFTWARE SA &tp

Volume 2 (2007)

Dear Reader,

It is our pleasure to present to you Proceedings of the *2007 International Multiconference on Computer Science and Information Technology* (*IMCSIT'07*), will take place in Wisła, Poland, on October 15-17, 2007. *IMCSIT* will be co-located with the *XXIII Autumn Meeting* of *Polish Information Processing Society* (*PIPS*).

IMCSIT is a result of the evolutionary process. In 2005 a *Scientific Session* took place during the *XXI Autumn Meeting* of *PIPS* and consisted of 27 refereed presentations. After this relative success (we have advertised the *Session* very late in the year) we have decided to expand and extend it into a full-blown conference but continue cooperation (co-location) with the Autumn Meeting of the Polish Information Processing Society. As a result the *IMCSIT'2006* was organized, and consisted of the "main session" and 6 sub-conferences. It received 111 submissions. After refereeing (at least 2 reviewers for every paper) 66 papers have been included in the *IMCSIT'06 Conference Proceedings*. This year, we have decided to omit the "main session" and utilize the *IMCSIT* name and logo as an "umbrella" for multiple conferences. As a result, *IMCSIT'07* consists of the following events (and Proceedings are organized into sections that correspond to each of them):

- *Workshop on Agent Based Computing* (*ABC IV*),
- *2nd International Symposium: Advances in Artificial Intelligence and Applications* (*AAIA'07*),
- *Workshop on Computer Aspects of Numerical Algorithms* (*CANA'07*),
- *7th International Interdisciplinary Conference on Electronic Commerce* (*ECOM-07*),
- *International Conference on Principles of Information Technology and Applications* (*PITA'07*),
- *International Workshop on Real-Time Software* (*RTS'07*) including a *Special Session* on *RTS Education,*
- *2nd International Workshop on Secure Information Systems* (*SIS'07*),
- *Workshop on Ad-Hoc Wireless Networks: Urban Legends and Reality* (*WAHOC'07*),
- *1st Workshop on Advances in Programming Languages* (*WAPL'07*).

Each of these conferences had its own group of *Organizers* and a *Program Committee* (listed in these Proceedings). We would like to express our warmest gratitude to all of them, as it is their hard work in attracting and later refereeing 162 submissions (growth of 68% over the number of papers submitted to *IMCSIT 2006*).

IMCSIT'2007 was organized in cooperation with the **Poland Chapter of the IEEE Computer Society** and the **Systems Research Institute Polish Academy of Science**. Furthermore, the *2nd International Symposium: Advances in Artificial Intelligence and Applications* (*AAIA'07*) was devoted to celebration of Janusz Kacprzyk's 60th birthday and organized in cooperation with:

- *Poland Chapter of the IEEE Computational Intelligence Society*
- *World Federation of Soft Computing*

- *EUropean Society for Fuzzy Logic And Technology*
- *European Neural Network Society*

We would also like to note that the *1st International Symposium: Advances in Artificial Intelligence and Applications* (*AAIA'2006*) was devoted to the memory of Professor Zdzisław Pawlak. During this event a special awards carrying his name and recognizing "a general contribution" and "a contribution by a young researcher" were awarded to: Łukasz Slabinski (in the category *Best Student Paper*) and Krzysztof Jassem (in the category *General Contribution*). This year the Zdzislaw Pawlak Awards will be presented again. They are sponsored by the *Mazowsze Section* and the *Board* of the *Polish Information Processing Society*.

Finally, we would also like to acknowledge our sponsors. Support for the *IMCSIT'2007* was provided by *Prokom S. A.* and *TPSA*. We are really grateful for its contribution.

We hope that you will enjoy this volume and find it useful in your work. We also hope that you will come to Wisła in October of next year, for the *2008 International Multiconference on Computer Science and Information Technology* (*IMCSIT'08*) and the conferences that it will consist of.

Members of the *2007 IMCSIT* Organizing Committee:
Maria Ganzha, Conference Chair, Systems Research Institute Polish Academy of Science, Warsaw, Elbląg University of Humanities and Economy.
Marcin Paprzycki, Systems Research Institute Polish Academy of Science, Warsaw.
Tomasz Pełech-Pilichowski, AGH University of Science and Technology, Krakow.

2007 International Multiconference on Computer Science and Information Technology

October 15-17, 2007
Hotel Gołębiewski, Wisła, Poland

http://www.imcsit.org/

CALL FOR PAPERS

IMCSIT is organized in cooperation with the IEEE Computer Society (Poland Chapter), and the Systems Research Institute of the Polish Academy of Science, by the Polish Information Processing Society.

The 2007 International Multiconference on Computer Science and Information Technology will consist of the following conferences:

- Workshop on Agent Based Computing (ABC IV) *http://www.abc.imcsit.org*
- 2nd International Symposium Advances in Artificial Intelligence and Applications (AAIA'07) *http://www.aaia.imcsit.org.*
- Comprising also special tracks:
  1. Computational Biology
  2. Heuristics in Continuous Global Optimization
  3. Computer Linguistics – Applications
- Computer Aspects of Numerical Algorithms (CANA'07) *http://www.cana.imcsit.org.*
- 7th International Interdisciplinary Conference on Electronic Commerce (ECOM-07) *http://www.ecom.imcsit.org*
- International Conference on Principles of Information Technology and Applications (PITA'07) *http://www.pita.imcsit.org.*
- 2nd International Workshop on Secure Information Systems (SIS'07) *http://www.sis.imcsit.org.*
- International Workshop on Real Time Software (RTSCS'07) *http://www.rts.imcsit.org.*
- Workshop on Ad-Hoc Wireless Networks: Urban Legends and Reality (WAHOC'07) *http://www.wahoc.imcsit.org.*
- 1st Workshop on Advances in Programming Languages (WAPL'07) *http://www.wapl.imcsit.org.*

During the AAIA Symposium we will celebrate 60th birthday of Professor Janusz Kacprzyk.

AWARDS

The AAIA Symposium awards two „Professor Zdzisław Pawlak Best Paper Awards" for contributions which are outstanding in their scientific quality. The two award categories are:

- Best Student Paper – for graduate or PhD students. Papers qualifying for this award must be marked as „Student full paper" to be eligible for consideration.
- Best Paper Award – for the authors of the best paper appearing at the Symposium.

In addition to a certificate, each award carries a prize of 300 EUR provided by the Mazowsze Chapter of the Polish Information Processing Society.

CONFERENCE LANGUAGE:

The language for paper submission and presentation is English.

PAPER PUBLICATION

- Proceedings of the 2007 IMCSIT will be published on a CD (ISSN 1896-7094).
- Each conference will then publish extended versions of selected papers in Special Issues of Journals and/or as Edited Volumes – for more details see sites of each event.

PAPER SUBMISSION

- Authors should submit draft papers (as Postscript, PDF of MS Word file, OpenDocument Text).
- The total length of a paper should not exceed 10 pages (LNCS style). LNCS style templates are available at the conference WWW site.
- Papers will be refereed and accepted on the basis of their scientific merit and relevance to the specific conference.
- Conference organizers reserve the right to move submissions between Conferences.

IMPORTANT DEADLINES:

Paper submission:	June 25, 2007
Notification of acceptance/rejection:	August 8, 2007
Camera-ready paper due:	September 7, 2006
Conference:	October 15-17, 2007

ORGANIZING COMMITTEE:

- Maria Ganzha (Chair), EUH-E, Elblag and IBS PAN, Warsaw,
- Marcin Paprzycki, SWPS and IBS PAN, Warsaw,
- Tomasz Pelech, AGH University of Science and Technology, Kraków,
- Krzysztof Węcel, Poznań University of Economics, Poznań.

CONTACT:

In case of questions related to the conference, please send an e-mail to: *contact@imcsit.org.*

Proceedings of the International Multiconference on Computer Science and Information Technology

Volume 2

October 15 – 17, 2007. Wisła, Poland

TABLE OF CONTENTS

Workshop on Agent Based Computing IV:

Call For Papers **1**

A Practical Approach for Researching Trading Agents' Behavior in a CDA Environment **3**
Galina Ilieva

A Common Base for Building Secure Mobile Agent Middleware Systems **13**
Guido van 't Noordende, Benno Overeinder, Reinier Timmer, Frances Brazier, Andrew Tanenbaum

Abstract software migration architecture towards agent middleware interoperability **27**
Jordi Cucurull, Benno J. Overeinder, Michel A. Oey, Joan Borrell, Frances M.T. Brazier

Overview of an approach for a Web based Mobile Agent architecture **39**
Marius Feldmann

Utilization of Software Agents and Web Services as Transducers for Legacy Software; Case Study Based on an SMTP Server **47**
Michal Oglodek, Maciej Gawinecki, Marcin Paprzycki

An Architecture for Resource Bounded Agents **59**
Slawomir Nowaczyk, Jacek Malec

Employment of mobile multiagent systems for attendance of transaction of distributed database **71**
Zofia Kruczkiewicz

2nd International Symposium Advances in Artificial Intelligence and Applications:

Call For Papers **83**

Using Regular Expressions in Translation Memories **87**
Jacek Gintrowicz, Krzysztof Jassem

A Rule–Based Characterization of Clusters of Genes **93**
Aleksandra Gruca

Open-ended Evolution in Flocking Behaviour Simulation **103**
Urszula Markowska-Kaczmar, Halina Kwasnicka, Marcin Mikosik

A Hybrid Clustering Method for General Purpose and Pattern Recognition **121**
Jan W. Owsinski, Mariusz T. Mejza

Application of Genetic Algorithm in Median Filtering **127**
Sandra Sovilj-Nikic, Ivan Sovilj-Nikic

Linguistic Knowledge Representation for Stochastic Systems **141**
Anna Walaszek-Babiszewska

Implementation and evaluation of the fuzzy system for the assessment of cadastre operators' work **151**
Bogdan Trawiński, Tomasz Karczyński

Fuzzy contrast enhancement for images in the compressed domain **161**
Camelia Popa, Aurel Vlaicu, Mihaela Gordan, Bogdan Orza

A Comparative Analysis of Crossover Variants in Differential Evolution **171**
Daniela Zaharie

Information Extraction Systems and Nominal Anaphora Analysis Needs **183**
Ireneusz Matysiak

Adaptive Differential Evolution: Application to Nonlinear Regression **193**
Josef Tvrdik

Color Mining of Images Based on Clustering **203**
Krzysztof Walczak, Lukasz Kobylinski

Using of a Graph of Action Dependencies for Plans Optimization **213**
Luká Chrpa

Manifestation of selective attention in Sigma-if neural network **225**
Maciej Huk

Recognition of Structured Collocations in A Inflective Language **237**
Maciej Piasecki, Bartosz Broda, Magdalena Derwojedowa

Polish Morphological Guesser Based on a Statistical A Tergo Index **247**
Maciej Piasecki, Adam Radziszewski

Computational Efficiency of Suboptimal Nonlinear Predictive Control with Neural Models **257**
Maciej Ławryńczuk

Improved TBL algorithm for learning context-free grammar **267**
Marcin Jaworski, Olgierd Unold

Pattern Extraction for Event Recognition in the Reports of Polish Stockholders **275**
Michał Marcińczuk, Maciej Piasecki

Collecting Polish-German Parallel Corpora in the Internet **285**
Monika Rosińska

A Rule Based Approach to Temporal Expression Tagging **293**
Pawel Mazur, Robert Dale

Truncated Importance Sampling for Reinforcement Learning with Experience Replay **305**
Paweł Wawrzyński, Andrzej Pacut

Adaptive Temporal Planning at Airports **317**
Pim van Leeuwen, Lian Ien Oei, Pieter Buzing, Cees Witteveen

Advantages of an Easy to Design Fuzzy Predictive Algorithm: Application to a Nonlinear Chemical Reactor **327**
Piotr Marusak

Preferences in Evolutionary Multi-Objective Optimisation with Noisy Fitness Functions: Hardware in the Loop Study **337**
Piotr Woźniak

On Dimensionality of Latent Semantic Indexing for Text Segmentation **347**
Radim Rehurek

Information System Based on Natural Language Generation from Ontology **357**
Wojciech Górka, Łukasz Bownik, Adam Piasecki

Intelligent system for docking ligands to protein active sites **365**
Zbigniew Starosolski, Andrzej Polański

Integrating Fuzzy Logic and Chaos **375**
Zhong Li, Wolfgang Halang

Automatic Form Filling Based on Ontology-Controlled Dialogue With the User **389**
Łukasz Bownik, Wojciech Górka, Adam Piasecki

Genetic Techniques in Modeling Visual Aspects of Non-Natural 3D Objects **399**
Elzbieta Hudyma, Marcin Koszow

An investigation of the mutation operator using different representations in Grammatical Evolution **409**
Jonatan Hugosson, Erik Hemberg, Anthony Brabazon, Michael O'Neill

Knowledge based segmentation of fundus eye images **421**
Leslaw Milosz Pawlaczyk

Genetic Programming for Dynamic Environments **437**
Zheng Yin, Anthony Brabazon, Conall O'Sullivan, Michael O'Neill

Computer Aspects of Numerical Algorithms:

Call For Papers **447**

Applications of Finite Element Methods in synovial joint numerical calculations **449**
Anna Kucaba-Piętal, Jarosław Sęp

Molecular Dynamic computer simulations of nanoflows **457**
Anna Kucaba-Piętal, Janusz Bytnar, Zbigniew Walenta

Linking of direct and iterative methods in Markovian models solving **467**
Beata Bylina, Jaroslaw Bylina

Airfoil shape optimization by coupling computational fluid dynamics with evolutionary multiobjective optimization **479**
Daniela Zaharie, Silviu Panica, Marius Stoia-Djeska, Mircea Dragan, Dana Petcu

Comparative Analysis of High Performance Solvers for 3D Elliptic Problems **483**
Ivan Lirkov, Yavor Vutov

On the computer simulation of heat and mass transfer in vacuum freeze-drying **493**
Krassimir Georgiev, Ivan Lirkov, Nikola Kosturski, Svetozar Margenov

Image reconstruction from incomplete data projections by means of iterative algebraic algorithms **503**
Nadiya Gubareni, Mariusz Pleszczynski

Parallel PCG algorithms for voxel FEM elasticity systems **517**
Svetozar Margenov, Yavor Vutov

Implementing the Conjugate Gradient Method on a grid computer **527**
Tijmen Collignon, Martin van Gijzen

Automatic First- and Second-Order Adjoints for Truncated Newton **541**
Uwe Naumann, Michael Maier, Jan Riehme, Bruce Christianson

7th International Multidiscipinary Conference on Electronic Commerce:

Call For Papers **557**

Legal Aspects of Deep Links on the Internet **559**
Artur Strzelecki

Application of stochastic processes in Internet survey **565**
Elżbieta Getka-Wilczyńska

Data model standardization for real-time e-commerce **579**
Przemysław Kacprzak, Mariusz Kaleta, Piotr Pałka, Kamil Smolira, Eugeniusz Toczyłowski

A purchasing power indicators as a tool for predicting market attractiveness for SMEs in a changing currency environment **589**
Jacek Wachowicz

Even Swaps Method for Developing Assessment Capabilities of E-Negotiation System **597**
Tomasz Wachowicz

The Brand Equity - Marketing and Financial Approach **607**
Urszula Świerczyńska-Kaczor, Paweł Kossecki

2nd International Workshop on Secure Information Systems:

Call For Papers **615**

Integrated, Business-Oriented, Two-Stage Risk Analysis **617**
Andrzej Białas, Krzysztof Lisek

Securing Voice over Internet Protocol **629**
Ahmad Ghafarian, Randolph Draughorne, Steven Grainger, Shelley Hargraves, Stacy High, Crystal Jackson

Anomaly Based Intrusion Detection Based on the Junction Tree Algorithm **641**
Evgeniya Nikolova, Veselina Jecheva

Model based code generation for fast-deployment security applications **651**
Gyula Simon, László Szabados, András Tóth

Dealing With Network Security in Academic Institutions - a Case Study **661**
Ivan Dolezal, Jiri Grygarek, Ondrej Jakl, Karel Krecmer

A Joint Meta-Linguistic Taxonomy of Intrusion Detection and Testing / Verification **671**
Krzysztof Brzeziński

Picture Passwords Superiority and Picture Passwords Dictionary Attacks **681**
Krzysztof Golofit

Design and Implementation of a Portable ID Management Framework for a Secure Virtual Machine Monitor **691**
Manabu Hirano, Takeshi Okuda, Eiji Kawai, Suguru Yamaguchi

Access Control for Cross-Organisational Web Service Composition **701**
Michael Menzel, Christian Wolter, Christoph Meinel

International Conference on Principles of Information Technology and Applications:

Call For Papers **713**

Application of Model Transformation in the Generic Framework for Traceability **715**
Anna Derezinska, Jacek Zawlocki

Performance of Index trees on Flash Memory **725**
Hyun Seung Lee, Ha Yoon Song, Kyung-Chang Kim

Monitoring services in Service Oriented Architecture **735**
Ilona Bluemke, Marcin Warda

Compositional Abstractions for Interacting Processes **745**
Maciej Koutny, Giuseppe Pappalardo, Marta Pietkiewicz-Koutny

The Concept of Quasi-objects in a Temporal Intelligent System **755**
Maria Antonina Mach

Federated Method Invocation with Exertions **765**
Michael Sobolewski, Michael Sobolewski

Dependability of the Explicit DMC Algorithm for a Rectification Process **779**
Piotr Gawkowski, Maciej Ławryńczuk, Piotr Marusak, Janusz Sosnowski, Piotr Tatjewski

On communication management as a key element of successful IT program **789**
Rafal W. Cegielski, Jaroslaw Chudziak, Joanna Meyer

Shared Ontologies to Increase Systems Interoperatibiliy in University Institutions **799**
Richard Gil, Ana María Borges, Leonardo Contreras

International Workshop on Real Time Software:

Call For Papers **809**

Interactive real-time control labs with truetime and easy java simulations **811**
Gonzalo Farias, Karl-Erik AArzén, Anton Cervin

Modelling and schedulability analysis of real-time sequence patterns using Time Petri nets and Uppaal **821**
Angelo Furfaro, Libero Nigro

Improving Dependability of Automation for Free Electron Laser FLASH **837**
Boguslaw Koseda, Tomasz Szmuc, Wojciech Cichalewski

Task jitter measurement under RTLinux operating system **849**
Pavel Moryc, Jindrich Cernohorsky

Robust Real-Time Communication for Large Scale Distributed Control Systems **859**
Mariusz Postol

Veriest: Reusing Verilog Designs in Esterel **871**
Menachem Leuchter, Shmuel Tyszberowicz, Yishai Feldman

Towards the Safety Verification of Real-Time Systems with the Coq Proof Assistant **883**
Olga Tveretina

A speed classification method for real-time controlled dynamic systems **893**
Paweł Piątek, Wojciech Grega

Dependability of Explicit DMC and GPC Algorithms **903**
Piotr Gawkowski, Maciej Ławryńczuk, Piotr Marusak, Janusz Sosnowski, Piotr Tatjewski

Incorporating Fault Tolerance into Component-based Architectures for Embedded Systems **913**
Shourong Lu, Wolfgang Halang

Using Preemption For Dependable Urban Vehicle Traffic **923**
Tiberiu LETIA, Sergiu BARBU, Florin Dinga

Invited Papers for the Round Table Discussion on Real Time Software Engineering Education:

The modular approach of a real-time course **933**
Shmuel Tyszberowicz

A Model for Educating Real-Time Software Engineers On-Demand **939**
Janusz Zalewski

ILERT - International Learning Environment for Real-Time Software-Intensive Control Systems **943**
Andrew J. Kornecki, Thomas B. Hilburn, Wojciech Grega, Jean-Marc Thiriet, Miroslav Sveda

Educational Objectives for Embedded Systems **949**
Wolfgang Halang

Workshop on Ad-Hoc Wireless Networks: Urban Legends and Reality:

Call For Papers **953**

Acoustic Target Classification In Wireless Audio–Sensor Networks **955**
Baljeet Malhotra, Ioanis Nikolaidis, Janelle Harms

Analysis of IEEE 802.11b/g Card Behavior in Multirate Ad-hoc Networks **965**
Katarzyna Kosek, Szymon Szott, Marek Natkaniec, Andrzej R. Pach

Self-organizing Ad-hoc Architecture as a Solution for Coverage Issues in WiMAX Metropolitan Area Networks **975**
Krzysztof Gierlowski, Józef Woźniak, Krzysztof Nowicki

Ad-hoc networking with low-cost devices: how to do it right **985**
Pawel Gburzynski, Wlodek Olesinski

Problems When Realizing Ad Hoc Networks: How A Hierarchical Architecture Can Help **995**
Stefan Bouckaert, Dries Naudts, Ingrid Moerman, Piet Demeester

1st Workshop on Advances in Programming Languages:

Call For Papers **1005**

Post object-oriented paradigms in software development: A comparative analysis **1009**
Adam Przybylek

Slicing wxHaskell modules to derive the User Interface Abstract Model **1021**
Daniela da Cruz, Pedro Rangel Henriques

Pattern-based Program Visualization **1025**
Daniela da Cruz, Pedro Rangel Henriques, Maria Joao Varanda Pereira

Machine Code Can Be Representation of Source Code With Optimization **1037**
Samir Ribić, Adnan Salihbegović

Executable form of the IEC 61131-3 ST language programs in the CPDev environment **1041**
Dariusz Rzońca, Jan Sadolewski, Bartosz Trybus

Adapting Service-finding for Component Flexibility in Object-Oriented Languages **1055**
Dominik Dahlem, William Harrison

OORS: An Object-Oriented Rewrite System with Applications in Retargetable Code Generation and Optimization **1057**
Gernot Gebhard, Philipp Lucas

Exstensions Of Deductive Concept in Logic Programing and Some Applications **1071**
Ivana Berkovic, Biljana Radulovic, Petar Hotomski

Adaptive Language Approach to Software Systems Evolution **1081**
Jan Kollar, Jaroslav Poruban, Peter Vaclavik, Jana Bandakova, Michal Forgac

Separate compilation of grammars with Tatoo **1093**
Julien Cervelle, Rémi Forax, Gilles Roussel

Flexible Object Representation using Properties **1103**
Koen Vanderkimpen, Marko van Dooren, Eric Steegmans

A Generator of SQL Schema Specifications **1113**
Slavica Aleksić, Ivan Luković, Miro Govedarica, Pavle Mogin

A QTI Metamodel **1123**
Sonja Radenkovic, Nenad Krdzavac, Vladan Devedzic

An L-Attributed Grammar for Adjoint Code **1133**
Uwe Naumann, Jan Riehme

On applying stochastic programming in mathematical theory of programming **1147**
Viliam Slodicak, Valerie Novitzka, Anita Verbova

Workshop on Agent Based Computing IV (ABC'07)

Organized within the framework of the International Multiconference on Computer Science and Information Systems and co-located with the XXIII Autumn Meeting of the Polish Information Processing Society.

Agent-based computing has been hailed as the next significant breakthrough in software development, with the potential to affect many aspects of computer science, from artificial intelligence to the technologies and models of distributed computation. Agent-based systems are capable of autonomous action in open, dynamically-changing environments. Agents are currently being applied in domains as diverse as business information systems, computer games and interactive cinema, information retrieval and filtering, user interface design and industrial process control. The aim of the ABC '07 is to bring together researchers and developers from industry and academia in order to report on the latest scientific and technical advances, discuss and debate the major issues, and showcase the latest systems.

The special session welcomes submissions of original papers concerning all aspects of software agents. Submissions involving both implementation and theoretical issues are encouraged.

Topics include but are not limited to:

- Agent architectures
- Agent-oriented software engineering
- Agent-based simulations
- Agent benchmarking and performances
- Agent communication, coordination and cooperation
- Agent languages
- Agent learning and planning
- Agent mobility
- Agent modeling, calculi, and logics
- Agent security
- Agents in the semantic web
- Applications and Experiences
- Simulating and verifying agent systems
- Multi-Agent Systems in Applications
    - Disaster Management
    - E-commerce
    - Workflow management
    - Other areas

International Programme Committee

- Myriam Abramson, Naval Research Laboratory, USA
- Costin Badica, University of Craiova, Romania
- Wernher Behrendt, Salzburg Research, Austria
- Walter Binder , University of Lugano, Switzerland
- Benno Overeinder, Vrije Universiteit Amsterdam, The Netherlands
- Giacomo Cabri, Università di Modena e Reggio Emilia, Italy
- Krzysztof Cetnarowicz, AGH, Poland
- Bengt Carlsson, Blekinge Institute of Technology, Sweden
- Vladimir Gorodetsky, SPII RAS, Russia
- Ewa Grabska, Jagiellonian University, Poland
- Dominic Greenwood, Whitestein Technologies AG, Switzerland
- Henry Hexmoor, Southern Illinois University, USA
- Pawel Kalczynski, University of Toledo, USA
- Tomas Klos, Centrum voor Wiskunde en Informatica, The Netherlands
- Ryszard Kowalczyk, Swinburne University of Technology, Australia
- Zofia Kruczkiewcz, Wroclaw University of Technology, Poland
- Michele Loreti, Università di Firenze, Italy
- Beniamino di Martino, Seconda Università di Napoli, Italy
- Ngoc-Thanh Nguyen, Wroclaw University of Technology, Poland
- Michal Pechouczek, Czech Technical University, Czech Republic
- Joaquin Pena, University of Seville , Spain
- Fred Petry, Naval Research Laboratory, USA
- Jan Peters, Fraunhofer Institute, Germany
- Thomas Potok, Oak Ridge National Laboratory, USA
- Volker Roth, FX Palo Alto Laboratory, Inc., USA
- Jarogniew Rykowski, Poznan University of Economics, Poland
- Sattar B. Sadkhan, University of Babylon, Iraq
- Janusz Sobecki, Wroclaw University of Technology, Poland
- Stanislaw Stanek, Katowice University of Economics, Poland
- Paolo Torroni, University of Bologna, Italy
- Walt Truszkowski, NASA - Goddard Space Flight Center, USA
- Rainer Unland, University of Duisburg-Essen, Germany
- Andrzej Uszok, Florida Institute for Humane and Machine Cognition, USA
- Tatyana Yakhno, Dokuz Eylul University, Turkey
- Yu Zhang, Trinity University, USA

Organizing Committee

- Raheel Ahmad, Southern Illinois University, USA
- Maria Ganzha, EUH-E and IBS PAN, Poland
- Marcin Paprzycki, SWPS and IBS PAN, Poland
- Shahram Rahimi, Southern Illinois University, USA

Proceedings of the International Multiconference on
Computer Science and Information Technology, pp. 3 – 12
ISSN 1896-7094
© 2007 PIPS

A Practical Approach for Researching Trading Agents' Behaviour in a CDA Environment[1]

Galina Ilieva

University of Plovdiv "Paisii Hilendarsky",
4000 Plovdiv, Bulgaria
galili@uni-plovdiv.bg

Abstract. During the last decade, there has been a huge increase in the volume of Internet business transactions which in turn led to the emergence of a new field of scientific research - agent-based computational economics. In order to be able to conduct successful transactions, agents must have appropriate strategies to participate in trading negotiations. This paper is aimed at investigating the process of multi-agent negotiations in continuous double auction (CDA) and describes one possible approach for solving the problem of resource allocation in a distributed computational environment. Our goal is to present a new software tool for e-auctions simulation and to investigate bidding agents' strategies. Via a series of experiments we prove that this new software application sufficiently models the process of making transactions in CDA. We propose a new method for quick and reliable strategy comparisons. Using this novel method, we can choose a bidding strategy that ensures the most effective resource allocation.

1 Introduction

The field of agent-based economics is an intersection point of computer science and microeconomics. Research in this field is carried out in several basic directions:

- Search for efficient mechanisms for decentralized and distributed computational systems control;
- Investigation of economic problems via an agent-based simulation and
- Implementation of intelligent agents for managing the buying and selling processes performed on behalf of their users in an e-commerce environment [1, 9].

The reasons which led to the increased interest in applying agent-based technologies in e-commerce are:

- Internet has turned from a hypermedia environment for information exchange into a convenient and easy to use business environment. This provides not only varied communication channels, but also an integral infrastructure for conducting e-business;

1 With the financial support of the Scientific Research Center at The University of Plovdiv under Contract No. 07И23 from 20.03.2007

- The amount of information concerning different goods and services on the Internet has increased immensely and it's processing takes much more time then before;
- Software agents possess certain qualities and characteristics which enable them to effectively represent (and in some cases entirely replace) their owners in gathering and analyzing relevant information and making decisions with whom, when and at what price to strike a deal.

The second part of this paper presents the existing software systems for e-market simulation in brief. The necessity for creating a new environment that models the behavior of agents participating in e-auctions is substantiated. In the third part of the paper MASECA, a new software environment for agent-based e-auctions modelling, is described. The experiments that have been conducted with the system, prove its applicability for researching agent strategies for bidding in the fourth part. Emphasis is put on the proposition for an effective method of comparison between the bidding strategies. At the end, the obtained results have been summarized.

2 Modern Systems for E-Market Modelling via Auctions

There are various research projects in the field of e-commerce for scarce resources allocation via auctions, but only a small number of them use agents as actors and e mploy different market mechanisms. For example, Kasbah [2] enables agents to engage in bilateral negotiations on the basis of three simple functional dependencies for specifying the value of next bid/offer. The system stimulates only one double auction and after improving some of its functions it is now being used for teaching students at MIT and is known as Market Maker. Fish Market [6] is a project of Artificial Intelligence Research Institute in Barcelona. It emulates a very narrow field in e-commerce by supporting only Dutch auction. Michigan Internet Auction Bot [11] is a configurable server for several types of auctions. It has been created on the basis of the annual Trading Agent Competition, TAC, which stimulates interesting research on autonomous agents' behavior in bidding for a packet of various services.

The abovementioned systems have many disadvantages. Firstly, they mainly simulate only one type of auction. Exception from this rule is the Auction Bot System, which models many types of auctions, but there it is bid for a batch of interrelated goods and services. Secondly, these MAC do not provide possibility for choosing from many predefined bidding strategies. Thirdly, it has to be pointed out that there is a lack of options for creating and experimenting with strategies defined by the user of the corresponding system.

In order for these limitations to be overcome, we created a new platform for e-auctions modeling called *M*ulti *A*gent *S*ystem for *E*lectronic *C*ommerce with *A*uctions (MASECA). MASECA is a software system for simulating and experimental researching of multi-agent negotiations via auctions. Participants try to cooperatively maximize market efficiency by using different built-in bidding strategies. Up to now, the bidding strategies most commonly employed in online auctions have been realized in the systems: Fixed Markup (FM), Zero Intelligence Unconstrained (ZI-U) and Zero Intelligence Constrained (ZI-C) [4], Zero Intelligence Plus (ZIP) [3], RB Strategy (RB) [10] etc.

3 General characteristic of Multi-Agent System for E-Commerce via Auctions – MASECA

3.1 Functional Capabilities of the System

MASECA is a software instrument for investigating different negotiation scenarios by online auctions. Under these scenarios the object of interaction is coming to an agreement between agents for a win-win type of deal and the goal is maximizing total profit of all participants in a specific auction under given market conditions. Not only does the environment provide virtual space for the existence of auctions and agents, but it also enhances interaction between agents and market and is a place for information storage. It gathers deals history, processes statistical information about negotiations and struck deals. This way MASECA infrastructure helps market participants in making bidding decisions. This platform allows experimenting with a variety of built-in agent strategies and for the purposes of such an experiment the values of some or all of their parameters can be changed. On the other hand, there is a possibility for using outside strategies and the connection can be established during experiments via a proper protocol for sending and receiving messages.

3.2 MASECA's Program Implementation

In choosing a development environment for system realization, comparative analysis for finding the strengths and weaknesses of the several types of systems has been applied. We concentrate our attention on the integrated development environment of Borland Delphi as a tool for creating a system for e- market modeling because of the high quality, security and reliability of the developed applications. Delphi ensures a relatively large multi platform, because from the source code a new code for the platforms Microsoft Windows, Microsoft .NET Framework and Linux can be generated. The new system will be installable and will mainly use Wintel-computer systems.

All basic possibilities for communicability (activation and deactivation of a separate thread process, sending and receiving messages from the environment, processing registered events, etc.) are present in an objects' hierarchy. The functionality of separate objects is enhanced in the hierarchy's upper levels, in accordance with objects' specifications.

While realizing the environment for modeling e-commerce, an effective protocol for exchanging information between market participants has been developed. It has been optimized and contains the minimum number of types of messages, which are exchanged between an agent and an auction. More details about MASECA's peculiarities are given in [5].

4 Comparative Analysis between Strategies Applicable to Agent-Based Negotiations under CDA Conditions for Heterogeneous Populations

4.1 Methods and Criteria

Methods for strategy evaluation for heterogenous populations are based on two diffe rent approaches. The first one of them (applied by [7, 8] etc.) consists of a comparison between the efficiencies of the strategies for balanced populations (where strategies are in the same ratio during the experiment). The second approach was suggested by Walsh et al. (2002) and studies unbalanced populations by means of evaluating replicator dynamics (RD). The criteria that are used for comparing and evaluating strategies are once again market efficiency, average transaction price, convergence coefficient α , etc. Unfortunately, all these indices reflect rather the overall efficiency of the whole population (group) of agents (regardless of the strategy one or another group of it uses), than direct impact of separate strategies. On the other hand, it is necessary that a large number of planned experiments be conducted in order for each of the participating strategies' impacts to be shown clearly and this involves a certain expense of machine resources and processing time.

For removing the abovementioned shortcomings of the methods for strategy evaluation and comparison applied till now, a new methodology is proposed in this work, it is based on the former approach and it consists of the following:

1. The strategies that are to be compared are specified and this includes identification of the type and specific parameters of each of them.
2. All the agents that are to participate in negotiations are divided into 4 groups:
a) group A - sellers with strategy S1;
b) group B - sellers with strategy S2;
c) group C - buyers with strategy B1;
d) group D - buyers with strategy B2.
The number of agents in each group for a given experiment is fixed.
3. The market conditions under which the experiments is to be conducted are determined. They include constraints of every agent (producer price if the agent has the role of a seller and reasonable price if it is a buyer). Usually, modeling market dynamics by a market shock is expected.
4. Duration of the time interval in which certain market conditions apply is provided (in days).
5. The number of experimental tries is provided. With their help accidental factor influence on experimental results is minimized.

The main peculiarity of the proposed methodology is that, for each experimental session, conducting two mutually complement tries is in order: one for the given strategies by groups (see point 2 of methodology) and the other one is for the same groups but with swapped strategies. This means that if at the beginning there have been provided groups with numbers and strategies as follows:

A - S1; B - S2; C - B1; D - B2,

then in the complementary try the following groups, presented as pairs of numbers and strategies, will be generated:

A - S2; B - S1; C - B2; D - B1.

The number of agents (and the agents themselves together with their price limitations by groups is preserved, but the strategies with which each group participates are changed (sort of inverted).

The main advantage of this approach is that, each pair of strategies (for sellers: S1, S2 and for buyers B1, B2) is consequently placed in one market situation (first try) and in the situation in which its opponent strategy was placed during the first try (complementary try). If total profit by groups (A,B,C,D) is registered for the main and the complementary tries, a fairly objective comparable index of the studied strategies (S1,S2,B1,B2) can be obtained.

During the conducting of experiments the following criteria of comparison of analysed strategies have been investigated:

a) average profit for a given strategy and the whole group of agents using this strategy (for session, for a try);

b) average profit for a given strategy for an agent (for session, for a try);

c) portion of the profit for a given strategy, as a portion of the total theoretically possible and actually realized profit.

4.2 Experimental Setting

As a tool for making the analysis the system MASECA was used, which fully co mplies with the requirements set by the new methodology for evaluation and comparison of strategies. Windows XP Professional Edition operating system with the following configuration: CPU AMD Athlon (tm) XP 1800+, 1.53 GHz, 768 MB RAM is used.

The experimental conditions of the e-market that is being modeled are:

- number of sellers 11;
- number of buyers 11;
- marginal prices of sellers before market shock:

0.75	1.00	1.25	1.50	1.75	2.00	2.25	2.50	2.75	3.00	3.25

- marginal prices of buyers before market shock:

0.75	1.00	1.25	1.50	1.75	2.00	2.25	2.50	2.75	3.00	3.25

- marginal prices of sellers after market shock:

1.25	1.50	1.75	2.00	2.25	2.50	2.75	3.00	3.25	3.50	3.75

- marginal prices of buyers after market shock:

1.25	1.50	1.75	2.00	2.25	2.50	2.75	3.00	3.25	3.50	3.75

Theoretically possible profit (total for all struck deals) for one try before the shock is 7.50, and equilibrium price is 2.00. After the shock theoretical profit is preserved (7.50), and equilibrium price rises by 0.50 to 2.50.

Strategy parameters are given in accordance with the recommended ones, presented by their authors.

The number of sessions before market shock has been set to 5.

The number of sessions after market shock has been set to 5.

The number of tries (pairs: a try and a complementary try) has been set to 50.

A series of experiments has been conducted, in which various ratios between the numbers of agents in groups A and B (and correspondigly C and D) have been set: 1:10, 2:9, 3:8, 4:7 и 5:6.

4.3 Estimating the Relevance of the Methodology - Experiments, Results and Analysis

Before actual experiments for comparing and evaluating different strategies can be conducted, it is necessary to be proven that the suggested methodology can provide relevant (objective) results and sufficiently small (permissible for analysis purposes) error.

In order for the relevance of the methodology to be determined and for the possible error to be estimated a model experiment was conducted in the beginning, in which instead of different strategies only one was used (in this case the most complicated realized strategy, RB, was applied). It is expected that if the methodology is relevant, then with one and the same strategy the adopted evaluation criteria will be the same or similar with precision near the error of the evaluation method, which is being tested. The same market conditions apply that were described above and the conducted experiments involve the following ratios of the numbers of bearers of competing strategies: 1:10, 2:9, 3:8, 4:7 and 5:6. The conditions of the model experiment are identical with those of the experiments that are to be really conducted for comparing different strategies

The results obtained experimentally are summarized in the following Table 1.

Table 1. Summary of results – average profit by strategies from all conducted experiments, with the ratios between strategies taken into account

Strategies' ratio	**1-10**	**2-9**	**3-8**	**4-7**	**5-6**
S1	0.43	0.54	1.06	0.91	1.66
S1' (complementary to S1)	2.96	2.74	2.33	2.43	1.65
S2	3.03	2.75	2.16	2.45	1.64
S2' (complementary to S2)	0.41	0.56	1.14	0.91	1.63
B1	0.12	0.56	1.07	0.58	1.02
B1' (complementary to B1)	3.93	3.59	2.93	3.42	3.17
B2	3.84	3.58	3.17	3.45	3.10
B2' (complementary to B2)	0.14	0.54	1.03	0.65	1.00
AvgS1-S1'	1.70	1.64	1.69	1.67	1.65
AvgS2-S2'	1.72	1.66	1.65	1.68	1.64
AvgB1-B1'	2.03	2.07	2.00	2.00	2.10
AvgB2-B2'	1.99	2.06	2.10	2.05	2.05
DeltaS1S2	-0.02	-0.01	0.05	-0.01	0.01
DeltaB1B2	0.04	0.01	-0.10	-0.05	0.04

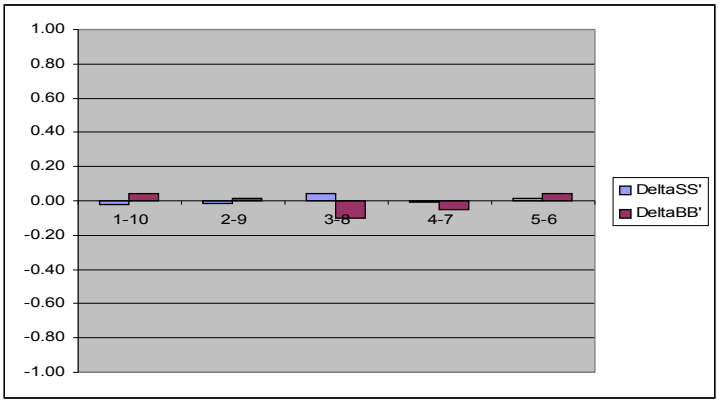


Fig. 1. Maximum difference (error) when comparing obtained results for average profit

It can be seen that the maximum difference (error) in comparison of obtained results for average profit according to strategies does not exceed 0.10 (obtained for ratio between strategies 3-8), which is amounts to 5% of the average realized profit of all strategies (1.86). This error is due not to the methodology of comparison but rather on the random determination of marginal prices for agents with different strategies and also on the random character of an array of elements in the process of negotiation, which is built in the strategies on purpose [3, 4, 10]. To a certain extent this error in strategy comparison according to suggested methodology would be reduced (to 3-4%) when increasing the number of tries in the experiment, but for the purposes that this development has a deviation of 5% totally complies with the relevance that is required by the methodology.

4.4 Comparison of the RB and ZiC strategies

In order for evaluation and comparison of the strategies RB and ZiC to be made a series of experiments under the conditions that have been described above was conducted. The results were presented in the following table and graph:

Table 2 Average profit for individual strategies (S-RB,S-ZiC,B-RB,B-ZiC) with ratio 1-10, 2-9, 3-8, 4-7, 5-6.

Strategy Description	1-10	2-9	3-8	4-7	5-6
Avg-RB	1.935	2.040	2.089	2.079	2.150
Avg-ZiC	1.705	1.633	1.550	1.549	1.489

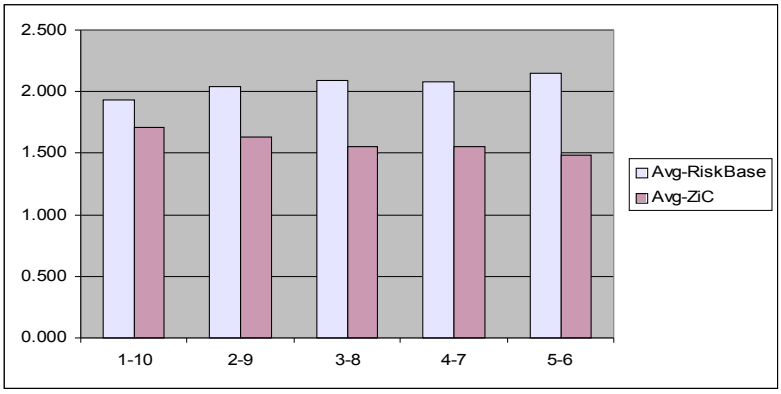


Fig. 2. Average profit for individual strategies (RB and ZiC) with different ratios

In order for the presented results to be analyzed, it is useful that the whole sequence of se ssions (from 1 to 10) be divided into several parts:

-P1: initial (including the first 2 sessions), in which agents act without preliminary information about the market and make their initial proposals in accordance with the algorithms built in their strategies;

-P2: established (Session 3, 4 and 5). In these sessions there is some collected initial information about market conditions (from the previous two sessions) and adaptive algorithms (if there are such) are applied.;

-P3: including Session 6 and 7, where a price shock is set, in which agents must aim at striking deals at the higher new level of the equilibrium price;

-P4: again established part (Session 8, 9 and 10), where agents can search for an optimal transaction level and try to maximize their profits.

When comparing data about activities of agents with different strategies, the following regularities can be observed:

1. In part P1 regardless of the ratio of the number of agents with different strategies, for the RB strategy there is a clear big difference between profit of seller and profit of buyer (on behalf of buyer). This is explained by the asymmetry while generating the initial proposal: with buyers initial proposals are limited to a price of 0.00 and may be very close to it (under the condition that there are no seller propositions on the market). Unlike this, sellers put initial price without having information about the price ceiling (maximum price at the market) and this is realized as a function of agent's limited price and some random element. This is why agents – sellers with low limited price can start with very low random propositions which will drive the price of initial transaction drastically down and as a result sellers's profit (as a whole) will be lower than that of buyers.

With the other strategy – ZiC – there is no such a great difference between the profits of seller and buyer groups, but still there is a differentiation depending on roles. In this part (P1) reaching equilibrium price when striking deals using the ZiC strategy is slower in comparison with the RB strategy and continues even in the beginning of part P2.

2. In part P2 agents with the RB strategy have already reached equilibrium market prices and the profits gained are significantly higher than those of the agents using the

ZiC strategy. The strategy is more stable and does not allow random deviations, as those that can be observed with ZiC.

3. In part P3 (price shock) there is variation in the values of realized profits for both agents using the RB strategy, and those using the ZiC strategy. With RB hesitations in profits immediately after the shock clearly have a large amplitude and small duration. However, for agents using ZiC profit varies with smaller amplitude but for a longer period of time (in some cases it continues to the middle of next part (P4)). It must be noted that buyers' profit is again higher because of the direction fo the price shock (increase in limiting prices and consequently in the equilibrium one), because agents' reaction is inert and in the period before finding the new equilibrium price (transition from old to new) deals are struck at lower prices.

4. In part P4 variations in realized profit have a tendency to fading, and this can be observed clearly for agents using the RB strategy.

5. In all parts the profit realized by the groups (sellers and buyers) with the RB strategy are higher (with no exception) than those of the opponent strategy. As it can be observed from the graphical data enclosed above, there are variations in the level of average profit for a given strategy depending on market conditions (respectively the number of Session), but no doubt agents employing the RB strategy reap higher profits.

6. Depending on the ratio of the numbers of agents in each of the groups with different roles and strategies (see **Fig. 2**) it can be seen, that the difference in profits for individual strategies strengthens when approaching the ratio of the numbers of agents with different strategies to 1.

7. Regardless of the ratio of the numbers of agents in each of the groups with different roles and strategies very high efficiency cannot be obtained (realization of theoretically possible transactions, e.g. a total profit of 7.50), because random element is heavily grounded in the ZiC strategy. In reality the total actual profit is between 7.20-7.30, which is 95-97% of theoretical profit.

This new research approach for comparison between stratagies reveals new relationships between the ratio of the numbers of agents using corresponding strategies and the results CDA shows for heterogeneous population.

Conclusions

In this work the newly created flexible program system for e-market simulation, named MASECA, has been described. The goal is by means of this software platform to research the agents' behavior in online auctions. Our efforts are aimed at studying CDA, since this is the most widespread mechanism for trading with currency and securities. Experimental results of the software proves that it is an appropriate instr ument for evaluating agent strategies. A new methodology for experimental research of the realized strategies in heterogeneous populations of agents has been proposed. On the basis of the results obtained by the new methodology for agent strategy comparison, proposals for increasing their efficiency have been made.

Future research directions are related to prove through dispersion analysis whether strategies' parameters affect their effectiveness, tuning parameter specifications and optimization proposals.

References

1. Chavez A., Dreilinger D., Guttman R. and Maes, P. (1997): *A Real Life Experiment in Creating an Agent Marketplace,* In Proceedings of the Second International Conference on the Practical Applications of Agents and Multi Agent Systems, Blackpool, UK: Practical Application Co., 1997, pp. 159-178.
2. Chavez A. and Maes P. (1996): *Kasbah: An Agent Marketplace for Buying and Selling Goods,* In Proceedings of the First International Conference on the Practical Application of Intelligent Agents and Multi-Agent Technology (PAAM'96), pp. 75-90. London, UK.
3. Cliff D. and Bruten J. (1997): *Minimal-intelligence agents for bargaining behaviors in market-based environments,* Technical Report, Hewlett Packard Research Labs, Bristol, England.
4. Gode D. and Sunder S. (1993): *Allocative efficiency of markets with zero-intelligence traders: Market as a partial substitute for individual rationality,* Journal of Political Economy, 101(1):119–137.
5. Illieva G. (2006): *An E-Market System Supporting Simultaneous Auctions with Intelligent Agents,* In Proceedings of the young researchers session of the IEEE 2006 John Vincent Atanasoff International Symposium on Modern Computing, October 3 - 6, 2006, Sofia, Bulgaria, ISBN:10:954-91743-5-2, ISBN:13:978-954-91743-5-9, pp. 41-47.
6. Rodríguez J. A., Noriega P., Sierra C. and Padget J. (1997): *FM96.5 A Java-based Electronic Auction House,* PAAM97, pp. 207-224.
7. Tesauro G. and Bredin J. L. (2002): *Strategic sequential bidding in auctions using dynamic programming,* In Proceedings of First International Joint Conference on Autonomous Agents and MultiAgent Systems, pages 591–598, 2002.
8. Tesauro G. and Das R. (2001): *High-performance bidding agents for the continuous double auction,* In Proceedings of Third ACM Conference on Electronic Commerce, pages 206–209, 2001.
9. Tesfatsion L. (1997): *How Economists can get Alife,* In Arthur, B., S. Durlauf and D. Lane (eds.), The Economy as an Evolving Complex System II, Addison Wesley 1997.
10. Vytelingum P., R. K., Dash E. David and N. R. Jennings (2004): *A risk-based bidding strategy for continuous double auctions,* In Proceedings of 16th European Conference on Artificial Intelligence, Valencia, Spain, pp. 79-83.
11. Wurman P., Wellman M., and Walsh W. (1998): *The Michigan Internet Auctionbot: a configurable auction sever for human and software agents,* In Proceedings of the Second International Conference on Autonomous Agents, pp. 301–308.

Proceedings of the International Multiconference on
Computer Science and Information Technology pp. 13–25
ISSN 1896-7094
© 2007 PIPS

A Common Base for Building Secure Mobile Agent Middleware Systems

Guido J. van 't Noordende*, Benno J. Overeinder**, Reinier J. Timmer,
Frances M. T. Brazier, and Andrew S. Tanenbaum

Department of Computer Science, Vrije Universiteit Amsterdam,
Amsterdam, The Netherlands
{guido,bjo,rjtimmer,frances,ast}@cs.vu.nl

Abstract. The Agent Operating System (AOS) provides the basic functionality needed for secure and reliable mobile agent platforms: support for secure communication, secure agent storage and migration, and minimal primitives for agent life-cycle management. Designed as a layer between local operating systems and higher level agent platform middleware, it supports interoperability between agent platforms and between different implementations of AOS itself. AOS has been tested on interoperability, both with regard to different higher-layer middleware platforms and interoperability between two implementations of AOS in C++ and Java.

1 Introduction

Multi-agent system applications often rely on agent platforms (agent middleware systems) for agent life-cycle management, communication, possibly migration, and security [7, 1, 8]. Most (mobile) multi-agent systems to date are monolithic systems, where all functionality is integrated in a single code-base, often implemented in Java. Even if systems have a more or less modular design (e.g., JADE [1]), they are generally not designed for interoperability with other systems, or easily allow for integration of components written in different languages. Some solutions and specifications for interoperability exist (e.g., FIPA), however these specifications often define only higher-level functionality such as interagent communication protocols, and do not define low-level protocols for agent migration or setting up (secure) connections between middleware systems.

The system described in this paper, called Agent Operating System (AOS), is intended to facilitate designing mobile agent middleware systems in a more modular way, by providing a common, language-neutral base for layering multi-agent middleware systems upon. AOS offers a well-defined interface that provides primitives for secure packaging and migration of agents written in various languages, and for the establishment of secure channels between different mobile agent middleware components.

This paper proposes a multilevel architecture for agent middleware: a common minimal AOS layer, and higher level middleware layers for platform specific functionality.

* Current address: Informatics Institute, University of Amsterdam, The Netherlands
** Current address: NLnet Labs, Amsterdam, The Netherlands

The common minimal base, the "kernel" to higher level middleware systems, is the main focus of this paper. The requirements and design considerations for the AOS kernel are identified in Section 2. Section 3 presents the architectural design of AOS and evaluates the design with respect to the requirements. Section 4 evaluates two implementations of the AOS kernel: one in C and one in Java. Related work is discussed in Section 5, and the paper concludes with a summary in Section 6.

2 Design Requirements and Motivation

Most mobile agent middleware systems are designed to support specific agent models and programming environments. These systems share functionality. AOS has been designed to provide a minimal common base which provides this shared functionality to mobile agent middleware systems. By design, AOS supports interoperability, facilitating open and extensible design of agent middleware, and enabling interaction and/or integration with (existing) middleware services.

The commonalities found between agent middleware systems can be broadly classified as: (i) mobile agent (code and data) storage and transport, (ii) primitives for agent life-cycle management, and (iii) secure communication between middleware processes (irrespective of what is actually stored in an agent or being communicated). In addition, all current multi-agent systems require security mechanisms that allow for, e.g., authentication and authorization of remote processes, and for integrity verification of migrated agents and content.

2.1 Requirements

To realize a common base for implementing secure and modular mobile agent middleware, the following requirements for the AOS kernel are defined:

- AOS should be a common layer for a broad range of agent middleware systems.
- The AOS design and specification should be language and operating system neutral, so that it can be implemented in any programming language and ported to any operating system. All these implementations should be able to interoperate.
- AOS should be minimal, in that it should only provides the minimal set of primitives required for building (secure) mobile agent middleware. Minimality ensures that the AOS code base becomes manageable and can be implemented in a robust and secure way. This implies that some mechanisms have to be implemented by the agent middleware itself, which is inherent to the idea that such mechanisms are middleware specific. In short, AOS should be "lean and mean", and provide only the basics needed for implementing (secure) mobile agent middleware.
- AOS should be reasonably efficient, within the expected performance boundaries of (secure) agent middleware. In particular, it should not add significant overhead compared to a distributed mobile agent system written from scratch, assuming that such a system is built with comparable (security) requirements in mind.
- AOS should not impose design limitations or a specific model on the mobile agent middleware designer. For example, it should not require the designer to adopt a specific deployment or security model (e.g., using a specific public key infrastructure).

- The AOS should be a stand-alone component which can be compiled and used without administrative privileges on a hosting machine.
- AOS should be usable by different mobile agent middleware systems concurrently, although stand-alone (non-shared) usage should also be possible.

2.2 Motivation of Design Requirements

The requirements outlined above have several important implications for the design and implementation of a minimal kernel. This section describes the rationale for the requirements and discusses some of the consequences.

Making AOS language-neutral is an important requirement to provide a support layer for agent middleware. Most current agent middleware systems have been implemented in Java, and consequently support Java as a programming language for agent development. However, this may not always be the most obvious programming language for agent development; C++ or Python may be preferable for the task at hand, maybe for code reusability, for interoperability, or for performance reasons.

The minimality and reasonable efficiency constraints have implications for interactions with the underlying system and management of resources. AOS only support local interactions, i.e., local processing accessing local resources, which should not block or interfere with other (local) operations. The only non-local interaction is communication. Restricting AOS to managing local resources only, avoids having the kernel wait for remote services to answer before it can complete a task. Management tasks spanning more than one machine are the responsibility of the higher-level middleware systems.

Security is very important in mobile agent systems, both from the perspective of the agent as well as of the host. As mobile agents move to foreign hosts (which may not always be trusted or trustworthy), their data and code should be protected from tampering. Mechanisms must be provided for timely agent tampering detection, and for authentication of remote agent middleware (see Section 3.2).

From a host's perspective, mechanisms are needed to protect hosts from malicious or erroneously programmed agents. Sandboxing (for interpreted executables) or jailing (for binary executables) [9], two examples of mechanisms that allow for protection of a host from malicious agents, require interaction with higher-level middleware systems in which life-cycle management is regulated. AOS provides mechanisms for agent code and data management, which provide the basis for implementing agent life-cycle management in the middleware layer.

The requirement that AOS should be usable by multiple mobile agent systems at the same time allows for AOS to provide an access point to multiple mobile agent middleware systems behind a firewall, among other things. The number of TCP ports for incoming connections should therefore be minimal. Sharing one AOS instance on a host requires an authentication mechanism for agent middleware to allow separation of operation. This mechanism is discussed in Section 3.

3 Architecture of the AOS Kernel

This section discusses the architecture and design of the AOS kernel.

3.1 Architectural Model of AOS

The intended use of AOS is to provide a common base to a range of specific *Mobile Agent Middleware (MAM)* systems. This common base can be seen as a kernel component in a layered middleware system design. Higher-level agent middleware systems use AOS for agent code and state management, agent migration, and communication. Higher-level MAMs extend the basic common AOS layer with agent middleware specific components and possibly one or more services, e.g., for agent life-cycle management, middleware management, and agent naming and location services. The general architectural model is shown in Fig. 1.

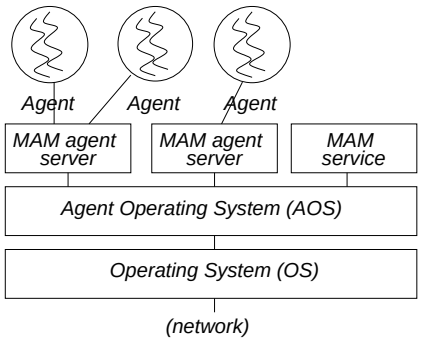


Fig. 1. Example of a layered agent middleware architecture using AOS. This example system consists of two agent server processes and one service (e.g., a naming service) running on the same machine. MAM processes communicate with other local or remote MAM components using AOS. Agents communicate with their runtime environment (e.g., agent server), and do not normally access AOS directly.

AOS provides a means for middleware processes to securely authenticate services and other MAM components in a system, to communicate with these components and services, and to migrate agents to other locations in a secure way. AOS has two external entries for receiving agents and for incoming connections to middleware processes that use AOS. How authentication, communication, and agent transport are implemented internally in AOS is described later in this section.

MAM components (distinct processes from an architectural point of view, see Fig. 1), are responsible for the agent runtime environment. Agents are executed by a MAM component, which provides a runtime environment (API) to them with MAM specific functionality. The MAM components use AOS internally when they communicate with each other, when an agent's state needs to be altered, or when it needs to be migrated to another AOS kernel. Note that services do not host agents (see Section 3.2).

AOS has a specification that clearly describes the methods (API) provided by AOS to higher-level middleware systems, including arguments and semantics. The AOS kernel effectively hides differences in the underlying operating system with regard to communication interfaces and file system access from the processes that use AOS. MAM

components invoke AOS API methods using RPC calls. Different RPC interfaces have been implemented, for Sun RPC, Java-RMI and XML/RPC. This allows the MAM processes to be implemented in a different language than AOS, even within the same MAM. AOS allows for multiple so-called dispatchers that implement different RPC implementations, to be used simultaneously such that any component can use an RPC interface of its choice.[1]

3.2 AOS Concepts and Primitives

The AOS API provides primitives for agent transport (shipping/receiving ACs), and communication (creating endpoints and connecting/accepting connections). In addition, the API contains calls allowing for safe sharing of AOS between different MAM components. The agent transport mechanism provides integrity protection of agent code and (meta-) data, and the communication methods include a simple but highly effective authentication model (see Section 3.3). Note that the API contains no primitives for process management. This is because different mobile agent middleware systems have very different methods for managing agent processes. For example, some agent systems use a thread-based model, where agents run as a thread in an agent server process, whereas in other systems each agent runs as an independent process. As a result, it is hard to attain a single, simple model for agent process management, and process management is therefore left to the mobile agent middleware.

Agent Containers Agent code, data, and meta-data (e.g., owner information, time of creation, permissions, etc.) are stored in AOS in a data structure called the *Agent Container (AC)*. The AC is, in fact, an archive with a table of contents, immutable segments (e.g., code), and mutable segments (e.g., state and auxiliary data). Primitives for creation of AC and segments, and reading/writing segments are part of the API.

In addition, a *finalize* call is used to synchronize any new content of the AC to disk (to allow for recovery if AOS crashes and is restarted[2]), and to generate checksums in a secure *Table of Contents (ToC)* of the AC's segments. The secure ToC is used for integrity verification of an AC when it is received by another AOS kernel. Finalize must be called before an agent can be migrated to another AOS.

Communication Endpoints Communication-related calls include creation and deletion of communication endpoints (similar to Unix sockets). Connect, accept, send, receive, and select calls exist which allow for setting up and using secure, reliable, ordered communication channels to these endpoints. Connections between the same pair of AOS kernels, with the same security properties (i.e., cryptographic *cipher suites*), are multiplexed over a single AOS "base channel" internally to AOS. This allows for

[1] In particular, it is straightforward for a Java-based middleware to use Java-RMI, whereas a C-based middleware implementation can more straightforwardly use Sun RPC.

[2] The API defines error codes that allow for detection of an AOS restart, and contains a call for re-initializing AOS resources after such an event was detected by the middleware.

a reduction of connection setup times by amortizing expensive initial secure connection setup times over multiple connections, compared to setting up new secure connections for each connection. The method for setting up secure connections is discussed in Section 3.3. Agent transport makes use of the same internal AOS connection implementation, allowing for safe (integrity and confidentiality protected) transport of agent containers.

Secure Sharing of AOS Secure sharing of a single AOS instance and its resources by different unrelated agent middleware on the same host is enabled by the concept of a *role*. A role is a set of resources associated with a cryptographically protected authentication token (called a *cookie*), which is used by an agent middleware to invoke methods on AOS. AOS creates this cookie securely, and also creates an internal data structure that describes the resources associated with this cookie. We refer to this cookie and its associated resources as a role.

During start-up of AOS, an *init role* is generated, that is allowed to invoke any method on the AOS API. The init role is used by an *init process* that controls usage of AOS (think of *init* process in Unix). Given the init role, the init process can generate other roles, which are called *child roles*. Generally, these roles allow users to create additional subroles for components within the same agent middleware.[3] For example, a service may only be allowed to use communication related calls, and agent servers may only be allowed to access the ACs of the agents they manage. Assigning a new role to each MAM component allows for compartmentalization of an agent middleware system. This avoids that a single compromised MAM component can compromise the state of another MAM component that uses the same AOS instance, e.g., by destroying AOS kernel objects such as ACs.

Roles determine ownership of resources in AOS. Resources include agent containers, communication channels, and child roles. When a role is deleted (using an AOS call), all resources associated with this role are deleted, including subroles and their resources. In principle, roles are persistent such that role information and resources (in particular, ACs) can be recovered after an AOS crash or system reboot. AOS implements resource protection by imposing limits on the number of resources (and subroles) owned by a role.

3.3 Authentication Model

AOS comes with a simple but highly effective authentication model based on public key cryptography, that is used when connections are set up and agents are shipped to other AOS kernels. The authentication model is based on the concept of *Self-certifying Identifiers (ScIDs)* [5]. A ScID is a SHA-1 hash of the public key of an AOS kernel, where the AOS kernel has access to the associated private key.

Each endpoint created by an AOS kernel, both for AC transport and for communication, is described by a data structure that contains a ScID in addition to the AOS kernel's endpoint information (i.e., IPv4/v6 address and port). This data structure is used by a

[3] Each user may obtain its own initial role by requesting it from the init process, or through some other way; the way in which an init process functions precisely is not currently specified.

MAM component to set up a connection or to ship an AC to another MAM component using AOS, where AOS internally verifies that its peer AOS kernel has the private key corresponding to the ScID in the AOS contact record. A standard authentication protocol (i.e., TLS/SSL) is used for authentication and key-exchange, as part of setting up an efficient, secure, encrypted channel to the peer AOS. The MAM component can specify the cryptographic cipher suite for the channel at connection setup time. The advantage of ScIDs over, e.g., X.509 based authentication models, is that no PKI infrastructures are required to bind keys to identities, as ScIDs are used as the name of the entity (AOS kernel), and are coupled directly to its key as described above.

3.4 End-to-End Authentication and Secure Communication

When AOS is shared between multiple MAM systems, each MAM should be able to authenticate its peer MAM end-to-end, to ensure it is not connected to another MAM that uses AOS at the same time. To this purpose, the AOS endpoint data structure explained above can be included in a *Middleware Contact Record (MCR)*, in addition to information that can be used to securely identify (authenticate) the middleware running on top of AOS. This MCR can be used by the MAM to set up a secure, end-to-end authenticated connection to another MAM, using an encrypted transport set up using the AOS contact information in the MCR. An MCR can contain a Self-certifying ID (ScID) of the peer middleware process, so that it can be authenticated using the mechanism outlined in Section 3.3. Another approach is to use a simple name in the MCR which can be combined with a PKI to securely authenticate a remote middleware process. AOS does not force any particular authentication model upon the middleware.

To securely use AOS channels as the basis for an end-to-end authenticated channel, both middleware processes must, after authenticating each other, exchange authenticated messages to each other that contain the AOS endpoint information of their own (trusted) AOS kernel. This is required, because if this check does not take place, an impostor AOS kernel may sit between the endpoint AOS kernels as a man-in-the-middle, which could decrypt and read all information passed over the channel—as generally only MAM authentication information is known at the time when a connection is made. After such information is exchanged, both parties are certain that they communicate through the remote AOS kernel that is actually used by their peer. Only in this case, can the confidentiality of the underlying AOS channel be trusted, allowing the middleware processes to let AOS take care of encryption of the connection itself without the middleware processes having to encrypt the connection end-to-end.

3.5 Secure Agent Migration

AOS provides mechanisms for shipping an AC to another AOS kernel. AOS signs a secure *Table of Contents (ToC)* of the AC, which is used by the receiving AOS kernel for integrity protection. The ToC data structure is available to the MAM layer. The MAM layer has to implement a secure agent transport protocol (ATP) on top of the AOS mechanism for shipping an AC to allow for additional, middleware specific authentication and control over agent transport. For example, specific code segments may have to be present in an AC as a prerequisite for starting it up in a specific MAM process. Other

extensions to the basic ATP provided by AOS are the construction of secure audit trails, describing all changes made to the AC during the agent's itinerary [8, 10]. Different security mechanisms at the MAM layer can be conceived (see, e.g., [10]); AOS provides all the necessary hooks to construct such mechanisms effectively.

4 Performance

For a central component such as AOS, which is intended to be used for all interprocess communication, mobile agent code/data management and migration operations, performance is highly important. This section presents the approach in which two AOS kernels were implemented and used in our department, and discusses performance of these AOS kernels. Measurements of performance of agent middleware that runs on top of AOS is not provided in this section; since these aspects are middleware implementation specific, it is chosen to focus on the overhead of functionality of AOS itself, which is most likely to impact the performance and scalability of the agent middleware system that uses AOS. In particular, communication throughput and scalability have been tested, and AC shipment related overhead and scalability in terms of concurrently shipped ACs.

Independently, two versions of AOS have been implemented, one in Java and one in C++, based on a precise specification of the AOS interface and the internal protocols used by AOS.[4] These AOS kernels have been thoroughly tested on interoperability. Both the Java and the C++ kernel are used to construct two different agent middleware systems in our group, Mansion [10] (written in C) and AgentScape [6] (written in Java). These two mobile agent systems are quite different in their design and implementation decisions; even so, AOS has shown to be a solid basis for their construction.

This section evaluates the performance of the Java and C++ AOS kernels. Although the tests are limited to AOS operations, the aspects measured are expected to most influence the performance of a mobile agent middleware system built using AOS.

All tests were run on a dedicated 1 GHz dual Pentium-III machine with 1GB memory, running Linux on an ext3 filesystem and using a Fast Ethernet (100 Mbit/s) local area network. Tests with the Java kernel used the Sun Java 1.5 standard compiler and Java HotSpot server virtual machine version 1.5. The cryptographic libraries used in the Java AOS kernel are from Bouncy Castle.[5]

The tests were run with modified AOS kernels that included microsecond timers, and were executed 5 to 10 times in a row, with averages shown in this section. For all tests that use AOS-to-AOS communication, the connection was configured to use 128 bits AES encryption with SHA-1 message authentication.[6]

4.1 AOS-to-AOS Communication Cost

AOS uses an internal protocol to multiplex communication channels over a single internal encrypted "base channel." Figure 2 shows the performance and scalability of AOS

[4] The AOS specification can be requested from the authors.
[5] `http://www.bouncycastle.org`
[6] AES provides a reasonable trade-off between security and efficiency, compared to e.g., 3DES.

for communication, for 1 to 16 threads communicating concurrently over AOS. In this experiment, each thread sends 25 MB over an AES encrypted base channel to a server process running on a different AOS kernel.

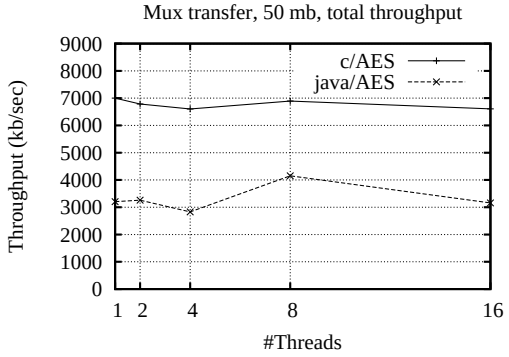


Fig. 2. Total throughput for multiplexed communication over a shared AOS-to-AOS connection.

As shown in Fig. 2, the C++ kernel has a substantially higher throughput than the Java kernel, which is due to the fact that the OpenSSL library implemented in C is faster than the pure Java Bouncy Castle SSL implementation used in the Java kernel.[7] Both kernels apply locking strategies to make sure that only a single thread can write payload on the base channel. The figure shows that the total throughput stays roughly the same for both kernels, irrespective of the number of threads that simultaneously send payload over the wire, although some variation exists which remains unexplained. Although the per-thread throughput decreases linearly with the number of threads for obvious reasons (i.e., sharing and overall saturation of the underlying connection), the AOS kernels and the internal protocol used for multiplexing do not adversely influence scalability.

4.2 Finalize Costs

Prior to shipping an AC, an AC must be finalized to ensure that the AC's table of content is generated, and that all segments are stored safely in a zip file synchronized to disk. Finalize is a call that constructs a secure Table of Content of the AC and signs it, prior to shipping it to another AOS kernel. In addition, finalize syncs the AC to disk for crash recovery reasons.

Table 1 shows a microbenchmark of the finalize costs of agent containers of 500 KB, 1 MB and 5 MB containing random data. These sizes are typical for many agents used in our own agent middleware system. ToC checksumming and signing cause little overhead, even for large ACs, and this increases linearly with the size of the AC. This is because the checksum (SHA-1 hash) generation has to take place over every byte of all

[7] Performance measures with an unencrypted (NULL) SSL channel show that Java performance in the unencrypted case comes close to the performance of the C++ kernel in the same scenario.

Table 1. Finalize micro benchmarks (in milliseconds) for resp. the C++ kernel and the Java kernel.

	C++			Java		
	500kb	1mb	5mb	500kb	1mb	5mb
checksum	9	19	98	36	74	70
sign	51	52	70	5	16	51
zip	133	248	1356	145	303	1449
sync	166	238	922	179	401	1623
total	359	558	2446	442	878	3854

segments. Creating a zip file and synchronizing it to disk cause substantial overhead, because zipping requires that each segment is copied into the zip file, possibly after compression. Synchronizing the resulting zip file to disk is also rather expensive. Finalize times scale roughly linearly with the AC sizes for both the Java and the C kernel, although finalize takes substantially longer on the Java kernel than on the C kernel.

As mobile agents may migrate often during their lifetime, AC finalize and transfer cost can increase the time for an agent to achieve its task considerably, and influences scalability of the mobile agent middleware as a whole, which is confirmed by experiences with our own agent middleware systems. A straightforward optimization for performance, is to have AOS ship segment files to another AOS kernel directly, without zipping the files first, in an FTP-like manner. Another possible optimization is to let go of the crash recovery assurance by means of the fsync system call.

4.3 AC Shipment Cost

AC shipment is composed of a `ship_ac` primitive combined with a `wait_ac` primitive at the receiving end, which returns after shipment is completed. Ship_ac takes a finalized agent container, and ships it over an SSL connection as described above. After receipt of the AC, the receiving AOS kernel extracts the agent's zip file containing the agent's segments, and verifies the checksums in and the signature over its ToC. Only after this verification, wait_ac returns. After an acknowledgement is received for all shipped ACs, the timer is stopped. The ship_ac cost measured thus includes both the on-the-wire time and the extract/verify cost at the receiving end.

The total ship_ac costs including AC extraction and verification were measured for AC sizes of 500 KB, 1 MB, and 5 MB containing identical segments of 5 KB random binary data. The cost of extracting and verifying an AC after it is received depends primarily on the size of the AC. The times for ZIP extraction (default compression ratio) and signature and checksum verification in the C++ kernel are 0.064, 0.127, and 0.734 sec. for 500 KB, 1 MB, and 5 MB, respectively. Extraction and verification in the Java kernel takes substantially longer, namely 0.597, 1.033, and 3.472 sec., respectively. Of these times, about 80–90% is spent on unzipping the AC.

Figure 3 shows the results for both the C++ kernel and the Java kernel for 1, 2, 4, 8 and 16 ship_ac calls at the same time. The figures show that AOS ship_ac calls scale roughly linearly with concurrent use. The figures also show that the time needed to ship an AC is more for the Java kernel than for the C++ kernel. This can be attributed in

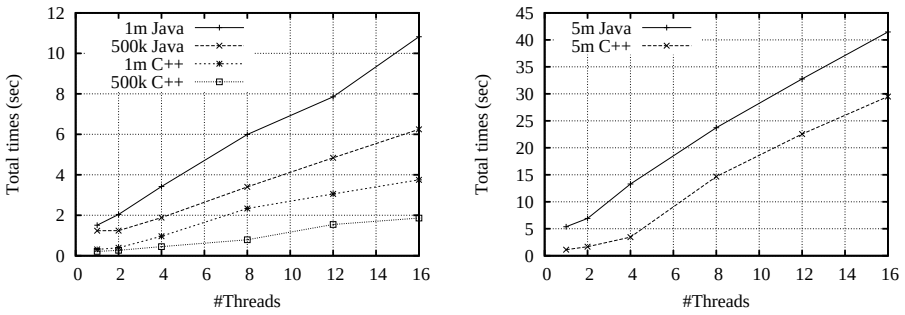


Fig. 3. Elapsed time to ship 1–16 Agent Containers of 500 KB resp. 1 MB (left), and 5 MB for both the Java and the C++ kernel.

part to the fact that cryptography (for encrypting the connection) and AC extraction and verification take longer in Java than in C++.

5 Related Work

Comparison of AOS with other related work should be done with the design requirements of AOS in mind. As AOS is not an agent middleware itself, but rather a middleware building block, comparing AOS with full functional middleware can only be done partially by considering the leading design requirements.

The FIPA standard specification includes a series of documents describing the functionality and operation of agent middleware. FIPA compliant agent middleware can interoperate which each other, e.g., agents can exchange messages, interact with, and reason about agents on other middleware. One of the most widely used FIPA compliant agent middleware is JADE [1]. The latest middleware design (version 3.5 as of today) is modular in design and many parties (universities and companies) have contributed to JADE. The middleware is implemented in Java and supports a Java API for agent development. It is a complete self-relying system, with integrated location and yellow pages services. This is different from the AOS perspective to agent middleware, where services can be arbitrary location or yellow pages services such as DNS or LDAP servers.

Ajanta [4] is designed to include a number of security primitives and architecture features to protect both the host and the agent from malicious actions. It includes amongst others a similar concept as the agent container in AOS, allowing for an audit trail mechanism resembling the one outlined in this paper and in [10]. However, Ajanta is completely Java-based and is not designed to incorporate or interact with other software components or services.

The Tacoma [3] project focuses on operating system support for mobile agents. In that respect, it has many similar design goals as AOS by providing abstractions for, in particular, data storage and agent mobility. Although it also provides a simple container abstraction, called a “briefcase”, only very simple protection mechanisms were implemented. Tacoma supports multiple programming languages for agents, in particular C and Tcl/Tk.

The MadKit agent platform architecture [2] aims to provide a generic multi-agent platform. The architecture is based on a minimalist agent kernel decoupled from specific agency models. Although there are similarities with the design goals of the architecture model with AOS, the design and implementation is quite different. The aim of MadKit is to allow a developer to implement its own agent architectures. Basic services like message passing, migration, monitoring, or management, are provided by platform agents. MadKit comes with a set of "containers", realizing different execution environments for running an application. Alternatively, AOS aims to provide a layer for constructing different agent middleware, and is not directly used by agents.

6 Summary

This paper discusses the design requirements, implementation, and performance of the AOS kernel. AOS is a portable middleware building block specifically aimed at constructing mobile agent middleware systems. It can be used by different MAM processes, possibly of different users, independently, where each such process may be implemented in a different language. Programming language flexibility is facilitated by the use of different RPC dispatchers, each providing a method invocation interface suitable for a specific language. The AOS design allows for secure sharing of a single AOS kernel between different middleware processes: it provides effective software fault isolation and safety by separating resources created by different middleware processes.

AOS provides a minimal set of primitives that are general to mobile agent systems, in particular for agent code and data storage, agent transport, and for communication between mobile agent middleware components. AOS provides basic security services which can be used by higher-level middleware layers to construct more elaborate security, such as authentication mechanisms and secure agent transport and auditing of mobile agents. AOS does not impose a specific model on the agent middleware.

Two implementations of AOS have been built and tested for interoperability. Performance measurements of the AOS kernel were shown in this paper, which show that AOS performs reasonably well, with noticeable differences between the Java and the C++ kernel. The C++ kernel outperforms the Java kernel for most tests, primarily due to the fact that C is more efficient than Java for tasks such as cryptography, which is used throughout the AOS kernel. On the other hand, Java provides better portability, and the Java kernel has been used to run the AgentScape mobile agent platform on Linux, Solaris, Mac OS X, and Windows systems. The C++ kernel is currently only available for Linux and Solaris platforms. Both implementations of AOS were shown to scale well with respect to concurrent usage by middleware systems for communication and transport of Agent Containers, which is important when using AOS to construct large-scale distributed mobile agent systems.

Acknowledgements

A number of colleagues have contributed ideas and helped to code parts of the AOS kernel. The authors would like to acknowledge Etienne Posthumus, Patrick Verkaik, Arno Bakker, David Mobach, and Michel Oey. Maarten van Steen and Niek Wijngaards

are acknowledged for early contributions to this work. This research is supported by the NLnet Foundation, `http://www.nlnet.nl`

References

1. F. Bellifemine, A. Poggi, and G. Rimassa. Developing multi-agent systems with a FIPA-compliant agent framework. *Software – Practice and Experience*, 31(2):103–128, 2001.
2. Olivier Gutknecht and Jacques Ferber. The MADKIT agent platform architecture. In *Proceedings of the International Workshop on Infrastructure for Multi-Agent Systems*, pages 48–55, Montreal, Canada, June 2000.
3. D. Johansen, R. van Renesse, and F.B. Schneider. Operating systems support for mobile agents. In *Proceedings of the 5th Workshop on Hot Topics in Operating Systems*, pages 42–45, Orcas Island, WA, May 1995.
4. N. Karnik and A. Tripathi. Security in the Ajanta mobile agent system. *Software – Practice and Experience*, 31(4):301–329, April 2001.
5. D. Mazières, M. Kaminsky, M. F. Kaashoek, and E. Witchel. Separating key management from file system security. In *Proceedings of the 17th ACM Symposium on Operating Systems Principles*, pages 124–139, 1999.
6. N. J. E. Wijngaards; B. J. Overeinder; M. van Steen; F.M.T. Brazier. Supporting Internet-Scale Multi-Agent Systems. *Data and Knowledge Engineering 41(2-3)*, 2002. pp. 229-245.
7. N. Suri, J. M. Bradshaw, M. R. Breedy, P. T. Groth, G. A. Hill, R. Jeffers, T. S. Mitrovich, B. R. Pouliot, and D. S. Smith. NOMADS: Toward a strong and safe mobile agent system. In *Proceedings of the Fourth International Conference on Autonomous Agents*, pages 163–164, Barcelona, Spain, June 2000.
8. Anand R. Tripathi, Tanvir Ahmed, and Neeran M. Karnik. Experiences and future challenges in mobile agent programming. *Microprocessor and Microsystems*, 25(2):121–129, April 2001.
9. G. J. van 't Noordende, A. Balogh, R. Hofman, F. M. T. Brazier, and A. S. Tanenbaum. A secure jailing system for confining untrusted applications. *International Conference on Security and Cryptography (SECRYPT), Barcelona, Spain*, July 28-31 2007.
10. Guido J. van 't Noordende, Frances M. T. Brazier, and Andrew S. Tanenbaum. Security in a mobile agent system. In *Proceedings of the First IEEE Symposium on Multi-Agent Security and Survivability*, pages 35–45, Philadelphia, PA, August 2004.

Proceedings of the International Multiconference on
Computer Science and Information Technology pp. 27–37
ISSN 1896-7094
© 2007 PIPS

Abstract software migration architecture towards agent middleware interoperability

Jordi Cucurull[1], Benno J. Overeinder[2], Michel A. Oey[2], Joan Borrell[1], and Frances M.T. Brazier[2]

[1] Department of Information and Communications Engineering,
Universitat Autònoma de Barcelona,
08193 Bellaterra, Spain
{jcucurull,jborrell}@deic.uab.cat
[2] Department of Computer Science, Vrije Universiteit Amsterdam,
Amsterdam, The Netherlands
{bjo,michel,frances}@cs.vu.nl

Abstract. Agent mobility is the ability of an agent to migrate from one location to another. So far, there are several difficulties with agents' migration due to the lack of interoperability among agent middleware that is distributed over the net.
In this paper, an abstract software migration architecture is presented, which is the first step towards full agent middleware interoperability. With this architecture, the process of migrating an agent is uniformly defined for multiple middleware, leaving the agent execution environment' standards as a future research.
To validate the suggested abstract migration, the architecture has been successfully implemented over two different agent middleware: JADE and AgentScape.

Keywords: Mobile Agents, Interoperability, IEEE-FIPA, migration, abstract implementation, JADE, AgentScape.

1 Introduction

The ability of an agent to migrate from one location to another provides a new dimension to multi-agent systems. An agent can decide to which location it chooses to migrate on the basis of the tasks it has at hand and the resources required. Different locations provide different services and different resources at different points in time.

Mobility between locations, however, requires interoperability between locations. Interoperability can be realised at different levels of a system: from operation systems level to application level. This paper proposes interoperability at application level. In this paper, an abstract software architecture is proposed to implement the Cucurull *et al.* [4] migration model based on a number of IEEE FIPA specifications. The model has been designed to support multiple migration protocols, providing developers the flexibility needed at application level.

To demonstrate the feasibility of the migration model, and the architecture design and implementation, two reference implementations of the architecture for the JADE [2] and AgentScape [10] agent middleware have been realised. Given the implementation

of the migration architecture on both middleware, agents can be sent between the two heterogeneous middleware types, but executing the agent is not possible (yet). The software migration architecture presented in this paper is, then, the first step towards a full agent middleware interoperability.

The paper is organised in seven sections: Section 2 summarises the migration architecture proposal; Section 3 presents the software migration architecture and its requirements; Section 4 presents the implementation of the architecture for the JADE platform; Section 5 describes the implementation of the architecture for the AgentScape platform; Section 6 states the related work; and Section 7 discusses the results and conclusions.

2 Migration proposal

2.1 Overview

The main concepts of the migration model proposed by Cucurull *et al.* [4] are presented in this section.

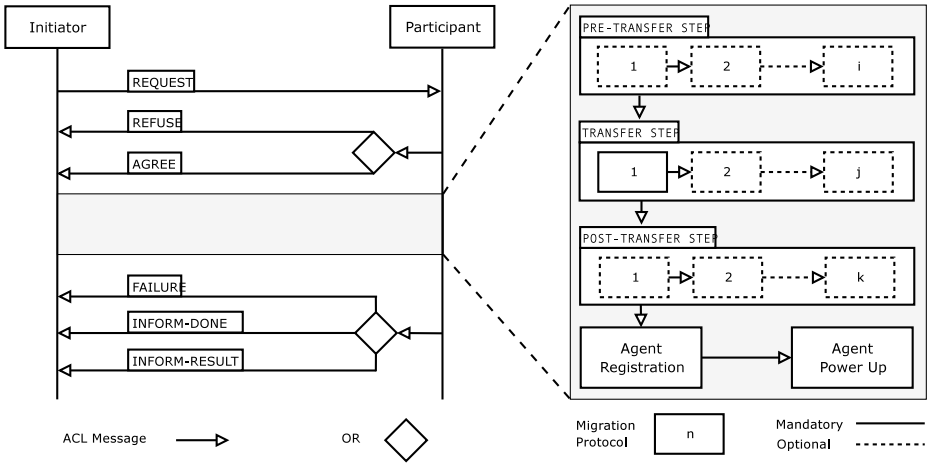


Fig. 1: Migration model.

The agent migration process is composed of a *Main Protocol* (left side of Figure 1), based on an IEEE-FIPA Request Interaction Protocol [6], and five steps within it (right side of Figure 1):

- **Pre-Transfer step**: Operations that need to run before agent transfer, e.g., authorisation, resource agreement, etc.
- **Transfer step**: Transfering of the agent code, data, and state. Different migration strategies (push, on demand, etc.) can be implemented.
- **Post-Transfer step**: Operations that need to run after agent transfer, e.g., authorisation, agent results transfer, etc.

- **Agent Registration step**: The agent is registered at the destination platform and, then, deleted at the origin (this prevents two agents running simultaneously and possibly get into conflict).
- **Agent Power Up step**: The agent is started in the destination platform.

The individual steps in the migration process are defined independently of each other. Each step in the protocol has its own (set of) protocols to perform its task. These protocols are well-defined in their functionality and operation, related to the step in the protocol; are independent of the other ones; can be composed of many stages; should use one or more ontologies and interaction protocols; and must have a unique name.

The first request–reply pair sent in the migration process follows an specific ontology specified in Cucurull *et al.* [4], and includes agent information related to its identity, its interoperability, and the protocols to use within the process. These protocols are the ones selected for the first three steps (Pre-Transfer, Transfer and Post-Transfer), which can be chosen by the agent before starting the migration process. The last two steps are fixed and always use the Agent Registration Protocol and the Agent Power Up Protocol respectively. If the two parts do not agree which protocol to use, or if some error happens during the process, the whole migration process is cancelled (a failure message is sent within the *Main Protocol*), and the agent execution is resumed at the local platform.

To have a minimum functional migration system, at least one protocol has to be available to the Transfer step. Three protocols are provided to illustrate the functionality of the model: the Push Transfer Protocol, the Agent Registration Protocol, and the Power Up Protocol. Note, however, that the Push Transfer Protocol is only an example. The choice of protocols for the first three steps is completely open. Developers are free to support the protocols of their choice—no default protocols are provided except for the last two steps.

2.2 Protocols

Push Transfer Protocol is the protocol used in the Transfer step to send all the agent code along with agent data and/or state within one message. Note that agent code is only sent if it is not already present in the remote middleware platform. The protocol is based on the use of the IEEE-FIPA Request and Proposal interaction protocols. The code, data and state sent are packed and coded according to the middleware platforms' requirements specified in the first message sent by the *Main Protocol*.

Agent Registration Protocol is the protocol used in the Agent Registration step. The protocol first sends a request to the remote middleware platform to rebuild and register the transmitted agent to the remote AMS. The AMS is a FIPA agent that maintains the directory of agents of each platform. If the remote middleware platforms acknowledges receipt, the agent's code, data, and state on the source middleware platform are deleted. This protocol uses an IEEE-FIPA Request Interaction protocol [6].

Agent Power Up Protocol is the protocol used in the Agent Power Up step. It requests the remote middleware platform to initiate the agent registered in the previous protocol. This protocol also uses an IEEE-FIPA Request Interaction protocol [6].

3 Proposed architecture

This section proposes an abstract software architecture to support the implementation of the migration model described in the previous section on the basis of requirements imposed to it. As stated earlier this architecture targets implementation of a migration protocol at application level, providing flexibility for developers.

3.1 Middleware requirements

The migration proposal outlined in Section 2 relies on the use of IEEE-FIPA specifications for the implementation of the different protocols in the five steps of the migration process. The agent middleware platform for which the implementation is targeted, should be FIPA-compliant or support at least:

- FIPA Agent naming scheme [7] or have a method to translate names.
- Agent directory service to register incoming agents.
- ACL messages since the whole migration architecture is based on exchanging them.
- Agent Communication Channel (the FIPA messaging service).
- FIPA SL content language to encode data included in the ACL messages.
- FIPA Request Interaction Protocol and other used within the migration protocols.

In addition to the requirements introduced above, certain rights to perform specific actions on the agent middleware platform are needed to perform migration of agents. For example, to send an agent to another agent middleware platform, the protocol needs access to the source middleware platform to access agent code, data, and state. Rights are needed to perform agent creation, to recreate and reinstantiate the migrated agent at the destination middleware.

3.2 Software architecture

Figure 2 shows the general idea of the application-level migration architecture, where each agent middleware platform has its own migration service. This service is provided by a dedicated agent, called the Agent Mobility Manager (AMM). The AMM agent implements the proposed migration protocol and communicates with the remote AMMs using ACL messages. As can be seen in Figure 2, the migration process is initiated by a message sent to the AMM (not defined by the proposal of Section 2 yet). This model supports migration initiated by the user and by the platform. Furthermore the AMM agent can host other future mobility related facilities, like acting as a message proxy for migrated agents, etc. Finally, the migration process supported by the architecture is:

1. Agent contacts local AMM, indicating it wants to migrate.
2. Local AMM suspends agent and gets the agent's code, data, and state.
3. Local AMM contacts remote AMM and starts migration:
   - pre-transfer step;
   - transfer step (local AMM sends agent code+data+state to remote AMM);
   - post-transfer step.

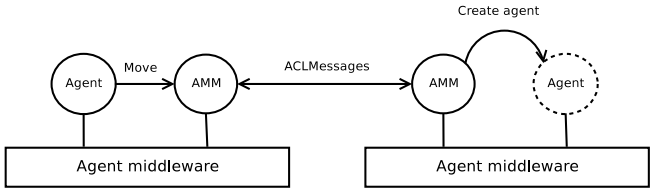


Fig. 2: Application level agent migration architecture.

4. Remote AMM creates and registers the new agent using the code+data+state (agent registration step).
5. Local AMM shuts down the local agent.
6. Remote AMM starts the new remote agent (agent power-up step).

The AMM should be supported by an agent middleware platform with the functionality described in the FIPA Agent Management specification [7]. The AMM is an extension of the FIPA architecture model. The internals of the AMM are left open to developers for each implementation.

4 JADE implementation

Most of the middleware requirements defined in Section 3.1 relate to the IEEE-FIPA specification. In this respect, JADE [2] provides all required functionality for the implementation of the migration architecture because it is FIPA compliant (see also [4]). This section presents the implementation of the migration protocol in JADE, making use of the service architecture of the JADE middleware. The section starts by describing this middleware, followed by the migration architecture implementation.

4.1 JADE middleware

Java Agent DEvelopment Framework (JADE) middleware [2] is a Java agent platform which provides a fully distributed system, where agents can reside on different hosts, an efficient transport mechanism with asynchronous messages, and internal (within a single JADE middleware platform instance) agent mobility support. An important feature of JADE is support for ontologies and content languages used in the agent messages. Each JADE agent runs as an independent *thread* and executes *behaviours*. Behaviours are tasks that can be dynamically inserted into and scheduled by an agent.

The JADE middleware is organised according to the FIPA specifications [7]. It is composed of the Agent Management Service (AMS), the Directory Facilitator (DF) and the Message Transport Service (MTS). The distributed nature of the middleware is achieved by using what are called agent containers. This is a specific solution of JADE, which is not part of the FIPA specifications, that allows several instances of the same middleware spread over one or more hosts. The agent mobility provided by JADE, on the contrary of the mobility implementation presented in this section, is between these containers. The JADE middleware architecture is depicted below in Figure 3.

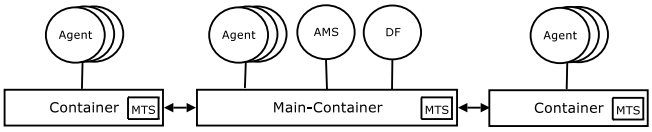


Fig. 3: The JADE agent middleware architecture.

Since JADE v3.2, a new design known as "distributed coordinated filters architecture" is followed. A small kernel and a set of extensible services provide middleware support to agents. The agent software migration architecture presented in Section 3 is implemented as one of these services.

4.2 Implementation of the software architecture

The architecture proposed in Section 3 extends standard JADE migration. Agents can migrate not only within a JADE middleware platform instantiation (between its agent containers), but also between different middleware platform instantiations.

Service level migration When an agent wants to move to another agent middleware platform instantiation, it calls its `doMove()` method with the agent's destination. This call is intercepted by our migration service which notifies the local Agent Mobility Manager (AMM) to start the migration process. Notice that, because of the service architecture of JADE and for compatibility reasons with its own mobility service, the way the migration process is initiated differs from the Figure 2, in which it is initiated by a message issued from the migrating agent to the AMM. This does not represent a problem and does not exclude a future addition of the method shown in the figure. Then, the process described on Section 3.2 is followed by the two involved AMMs. Notice the agent will keep its JADE name in the destination since it is a FIPA compliant name.

Organisation The proposed migration architecture is implemented by the AMM and by a number of additional components: protocols, ontologies, mediators and schedulers.

The AMM is a generic JADE agent which runs several JADE behaviours that offer the proposed architecture functionality. This agent serves migrating agents until the middleware is stopped.

Migration protocols are implemented using FIPA interaction protocols, which in JADE are provided by sets of two behaviours (one acting as an initiator and the other as a participant). *Ontologies* are used to exchange ACL message complex content information by aforementioned protocols. They take advantage of the JADE support for content languages and ontologies. *Mediators* are components that ease the dialog between the AMM and the middleware internal services to support the agent migration process with operations like: serialise, register, start agent, etc. And, finally, *schedulers*, are components that schedule agents that have requested to be migrated to start the migration process.

5 AgentScape implementation

This section describes the second implementation of the migration architecture—the implementation of the migration architecture in AgentScape. The AgentScape middleware, the FIPA messaging service addition and, finally, the migration architecture implementation are described in turn.

5.1 AgentScape middleware

AgentScape middleware [10] (see Figure 4) is a multi-language mobile agent middleware platform designed to support scalable, secure, distributed multi-agent applications. Agents migrate between virtual domains called *locations*. An AgentScape location consists of one or more hosts running the AgentScape middleware, typically within a single administrative domain. Each AgentScape location runs a *location manager* process on one of the hosts. Each host has its own *host manager*.

The AgentScape *kernel* [12] provides low-level secure communication between the higher level middleware processes, between agents and secure agent mobility. *Agent servers* provide language dependent run-time environments for agents, *Web service gateways* support Web service access and monitoring using the SOAP/XML protocol. The kernel is implemented in both Java and C.

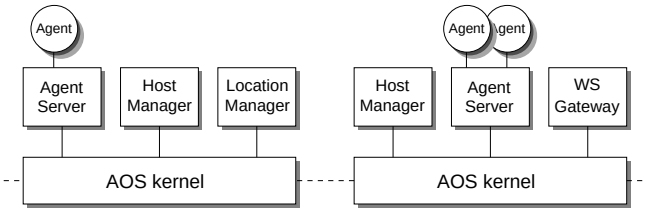


Fig. 4: The AgentScape middleware architecture.

5.2 FIPA Workaround

Due to differences in design objectives, FIPA compliance has never been an AgentScape design objective. AgentScape's main design objective is to support large scale distributed multi-agent applications efficiently, leaving the choice of interaction protocols, languages and naming up to the developer. The option to support FIPA based agents, however, is a design objective. To this end, an implementation of an application level migration architecture is proposed (in addition to the system level protocol AgentScape currently supports) to support the FIPA based migration architecture of Section 2, fulfilling the requirements listed in Section 3.1.

To support ACL messages and the SL content language several classes with this functionality have been ported from the JADE agent platform. Meanwhile, the Agent

Communication Channel (ACC) has been implemented from scratch. Since AgentScape agents does not support FIPA messages by default, a specific component called MessageHandler has been created to allow each agent to send and receive these messages. A directory of agents with their FIPA names has also been included.

5.3 Migration implementation

The following paragraphs describe the implementation details at the application level and the resulting final structure of the architecture.

Application level migration The migration scheme for AgentScape follows the application level procedure as described in Section 3.2 (see also Figure 2).

In the AgentScape migration process, an agent that wants to move to a remote platform, serialises itself, searches for the local Agent Mobility Manager (AMM), comparable to the AMM defined for JADE, and sends a message to it requesting to migrate. The local and remote AMMs take care of the rest of the migration process. Then, the remote AMM creates a new agent and reinstantiates the state of the agent from just before the migration. Since the platform internals have not been modified, it is important to note that the new agent will have a different AgentScape identity, but it will keep the original FIPA name.

Organisation As in JADE the migration architecture is implemented by the AMM. It is composed of several components: protocols, ontologies, mediators, and schedulers.

The AMM is implemented and registered as a standard AgentScape agent. Furthermore, it is registered in the Agent Communication Channel (ACC) to allow communication with other remote AMM agents. Once started, it remains waiting for non-ACL agent messages asking to migrate, and for ACL messages of other AMMs to start the reception of agents.

Migration protocols are implemented by FIPA interaction protocols, which in AgentScape are provided by sets of two classes. One class acts as an initiator and the other as a participant, this last running in background. *Ontologies*, *mediators*, and *schedulers* operate in the same way as its counterparts in the JADE implementation (refer to Section 4.2).

5.4 Open issues

The proposed software migration architecture described in Section 3 leaves some open issues when it is used with middleware that is not compliant with the FIPA specifications. The issues appeared in AgentScape middleware migration implementation and its proposed solutions are presented in the following list:

– **FIPA compliance workaround**: The limited set of FIPA standards needed for the migration architecture has to be added to the middleware. For simplicity reasons, they have been integrated at the middleware application level. This option is the least complex because it can be implemented using static libraries and agents without modifying middleware internals.

- **FIPA agent naming**: Agent local naming scheme should be mapped to the FIPA agent naming scheme and vice versa. In case a direct mapping between the two schemes were not possible, a method based on constant tables linking the two agent names should be developed to do the name translations. The latter method is used in AgentScape.
- **Middleware local agent naming**: Local agent name (the one following the middleware local name scheme specification) will not be maintained in the destination platform, unless it can be mapped directly from the FIPA name. Since this is a FIPA based migration architecture, the only agent information sent is the agent AID (Agent Identification) defined within the FIPA Agent Management specification [7]. In fact, if agents use the FIPA name to refer to them, this is not an issue, because these names are kept. In AgentScape, since the agent identification cannot be directly mapped to a FIPA name, the local agent name cannot be kept.
- **Agent life-cycle**: The agent life-cycle shows the operational state of an agent in a specific moment. By using it, for example, the messaging service can decide to deliver a message to an agent or wait to do this later. When an agent migrates, its life-cycle should be in a specific state (in FIPA the transit state). Then, the agent life-cycle provided by the middleware, using the migration proposed, should be in a state equivalent to the life-cycle defined by FIPA. This might be specially important when, in a nearby future, agents would be able to migrate between different kinds of middleware. In the present implementation, agent local life-cycle is not changed when it is migrating.

These are the issues that have been found implementing the software migration architecture described in Section 2. Some of the solutions have not been implemented as they are not critical, but have to be taken into account for a complete future agent middleware interoperability.

6 Related work

For mobile agents to be deployed on a larger scale, interoperability between platforms needs to be ensured. Ametller *et al.* [1], distinguish several areas of interoperability between agent platforms: executable code, platform, and communication protocols. Milojicic [9] describes process migration issues (not limited to mobile agents), with respect to the executable code area. The Kalong architecture [3], a mobility module used in the Tracy middleware focused on the optimization of migration, has also been used in other middleware [11] but without relaying on agent standards. A number of organisations have initiated the development of standards for mobility at the levels of platform and communication in an attempt to deal with the problem of incompatibility and interoperability with varying degrees of success. One example is the OMG-MASIF [8] specification by the OMG group, which deals with the standardisation of mobile agent APIs. At the level of agent communication, IEEE-FIPA standards have been defined [7].

An early attempt to define standards for mobility [5], proposed in the IEEE-FIPA context, was deprecated due to lack of implementations. The proposal in Section 2 shares the same general aim, supporting multiple migration protocols.

7 Conclusions

This paper presents a software migration architecture based on the model proposed by Cucurull *et al.* [4] for inter-platform agent migration. A list of minimal requirements, incorporating a number of FIPA standards, has shown to be sufficient to implement this architecture on two agent platforms: one fully FIPA compliant, one not.

The JADE middleware, FIPA-compliant, satisfies, by definition, all the architecture requirements. The migration service for JADE extends JADE's standard functionality.[3] The AgentScape middleware, the non-FIPA compliant agent middleware platform, has been extended to fulfill the requirements.[4]

Although the presented work makes it possible to migrate agents between different platforms, the shared ontology for execution interoperability has not yet been defined. This is topic of current research. Once this has been completed, interoperability between different agent platforms will be a fact.

Acknowledgements

Work partially funded by the Spanish project TSI2006-03481, the Catalan project 2005-FI-00752 and the European Social Fund (ESF). The authors are grateful for the support provided by Stichting NLnet `www.nlnet.nl`, Martijn Warnier, and Reinier Timmer.

References

1. J. Ametller, J. Cucurull, R. Martí, G. Navarro, and S. Robles. Enabling mobile agents interoperability through fipa standards. In M. Klusch, M. Rovatsos, and T.R. Payne, editors, *Cooperative Information Agents X*, volume 4149 of *Lecture Notes in Artificial Intelligence*, pages 388–401, Edinburgh, UK, September 2006. CIA 2006, Springer Verlag.
2. F. L. Bellifemine, G. Caire, and D. Greenwood. *Developing Multi-Agent Systems with JADE*. Wiley, Chisester, England, 2007.
3. P. Braun and W.R. Rossak. *Mobile Agents: Basic Concepts, Mobility Models, and the Tracy Toolkit*. Morgan Kaufmann, San Francisco, CA, 2004.
4. J. Cucurull, R. Martí, S. Robles, J. Borrell, and G. Navarro. FIPA-based interoperable agent mobility. In *Multi-Agent Systems and Applications V*, LNAI. Springer, 2007.
5. FIPA. FIPA agent management support for mobility specification, 2000.
6. FIPA. FIPA request interaction protocol specification, 2002.
7. FIPA. FIPA agent management specification, 2004.
8. OMG Mobile Agent Systems Interoperability Facilities Specification (MASIF).
9. D. S. Milojicic, F. Douglis, Y. Paindaveine, R. Wheeler, and S. Zhou. Process migration. *ACM Computing Surveys*, 32(3):241–299, September 2000.
10. B. J. Overeinder and F. M. T. Brazier. Scalable middleware environment for agent-based Internet applications. In *Proceedings of the Workshop on State-of-the-Art in Scientific Computing (PARA'04)*, pages 675–679, Copenhagen, Denmark, June 2004. Published in Applied Parallel Computing, LNCS 3732, Springer, Berlin, 2006.
11. D. Trinh P. Braun and R. Kowalczyk. Integrating a new mobility service into the JADE agent toolkit. In *MATA*, volume 3744 of *LNCS*, pages 354–363. Springer, 2005.

[3] Available at `http://jipms.sourceforge.net` as development release 1.98.
[4] Available at `www.agentscape.org`.

12. Guido J. van 't Noordende, Benno J. Overeinder, Reinier J. Timmer, Frances M. T. Brazier, and Andrew S. Tanenbaum, *A Common Base for Building Secure Mobile Agent Middleware Systems*, Proceedings of the Workshop on Agent Based Computing IV (ABC'07), pp. 13–25, Wisla, Poland, October 2007.

Proceedings of the International Multiconference on
Computer Science and Information Technology, pp. 39 – 45
ISSN 1896-7094
© 2007 PIPS

An approach for using the Web as a Mobile Agent infrastructure

Marius Feldmann

Department of Computer Science
Dresden University of Technology
Dresden 01062, Germany
feldmann@rn.inf.tu-dresden.de

Abstract. The paper describes an approach for establishing a mobile agent architecture in the Web. This architecture makes use of currently much discussed interaction principles and the document centric constitution of the Web. It builds out an abstraction layer useful for mobile agent execution and allows realizing various feasible Web applications following the mobile agent principle.

1 Introduction

After the idea of mobile software agents had spread in the middle of the 1990s it was assumed that they would be one of the major design principles for future distributed systems. Due to their characteristics several advantages over client-server and further architectures have been pointed out [1]. This optimistic opinion changed at the beginning of the new century rapidly. There were even statements that "the only widespread incarnation of mobile software agents is malware" [2]. Beside the lack of common implementations the main reasons for the pessimistic view on mobile agents especially regarding their success in the Internet are their technological complexity, their need for special security considerations and the absence for a widespread execution infrastructure. [3]
Despite all these arguments against mobile agent systems nowadays they have assured their position in various dedicated application areas. Even though their success regarding the Internet is still missing, though there are various imaginable application scenarios realizable by the mobile agent approach in a very straight forward way. Areas of these applications include especially information retrieval and aggregation, monitoring information sources or proxy functionalities.
The approaches that have been tried to establish a mobile agent infrastructure in the Internet (e.g. Java Aglets) were mainly designed by top down approaches. After analyzing the technical needs and creating the isolated, often monolithic system it was intended to be installed on various servers in the Internet to bring the vision of mobile agents to reality. This approach failed due to the non-appearance of spreading of the created infrastructure and its too high complexity.
A more promising approach is the use of as much already existing infrastructure as possible. For this the World Wide Web offers an ideal starting basis. It provides well-known principles, protocols and languages. They can be used as the building block for a mobile agent based architecture. By this it is possible to avoid reinventing the wheel and use various available tools. Due to its document-centric nature the Web

enables a very abstract view on technical details and thus reduces complexity of the overall agent infrastructre.

2 General idea and Related Work

The general idea behind using the Web as an execution environment for mobile agents results from the powerful tool set provided by it. The Web comes with easily understandable document description formats (mainly based on XML), a global addressing scheme (URI) [6] and the application level request-reply protocol HTTP.
The potentials of Web documents open a wide range of application areas. They are enrichable with logic that can be interpreted and executed on Web servers and on Web browsers. For this purpose XSLT [9] might be considered, but generally ECMAScript (JavaScript) is used. Furthermore Web documents are addressable by URIs leading to the fact, that code contained in a document is natively accessible through URI fragment identifiers. The third advantage of Web documents is their interpretation within the scope of the Document Object Model (DOM). This model has been defined by the W3C for dynamic manipulation and information extraction of tree-structured XML based Web documents.
In nutshell Web documents can be considered as

- logic and data encapsulating,
- dynamically modifiable,
- migrateable,
- reactive or even proactive

entities. With these characteristics they fulfill all requirements for mobile agents e.g. defined in [4]. Everything what is required is based on generally available standards and widespread mechanisms. By using well-known Web standards the system gains a high degree of interoperability with all types of technologies in the Web and makes an integration of Web accessible information possible. The separation of code and data is disposed. Both are components of documents and thus are part of the same domain of technology.
The idea of using documents for representing software agents is commonly known. It has already been described in various approaches. Especially XML has been used to define languages for agent description. The available approaches are often summarized under the term Active Documents. They for example occur in systems like Adlets [7] and have been extended for utilization in combination with the World Wide Web [8]. These systems define additional semantics of own code constructs included into the documents and interpreted by a specialized runtime environment. By this specialized runtime environment they follow rather the above mentioned top down approach than first checking for reusability of principals and techniques.
Apart from using documents as full-featured mobile agents they have also been utilized for distributing separated binary represented code. The most famous technology in this category is the Java Applets mechanism. Though Applets are considerably widespread, they underachieved. They have been designed as separate technology that did not evolve out of regular Web description formats, which might be a major reason for failure. Beside other mechanisms, Applets are a very good example that due to the slightly innovation resistant character of the World Wide Web new technologies have to evolve by stepwise modification of existing ones. By trying

to establish a document centric mobile agent system that is based on JavaScript, the DOM and HTTP this evolutionary condition is fulfilled. The currently often used HTTP interaction mechanisms Asynchronous JavaScript and XML (Ajax, [5]) and HTTP streaming enlarge the possibilities of such a system.
By using HTTP as communication protocol many predefined and widespread security mechanisms defined for this protocol can be used. These include especially authentication mechanisms and the extension HTTPS [11] for additional communication encryption. Based on these technologies a system model representing the mobile agent's execution environment has been developed that can be implemented in arbitrary languages. The model might be understood as an extension to the Representational State Transfer (REST) model [10].

3 Fundamental System Model

The system model is based on two components:

1. the Pneuma: the document that encapsulates data together with a execution state and logic (= the mobile agent)
2. the Soma: the execution environment that is available on Web servers and Web browsers

Document based mobile agents are intended to run both on Web servers and on Web browsers. Due to the different abilities of these two host types different requirements for the two resulting types of Somata have been formulated. For instance Web browsers are not natively addressable leading to the need for a workaround to place a mobile agent on one specific browser.

3.1 Considerations for the Pneuma

The Pneuma is based on regular Web standards, especially on JavaScript. Beside this, logic might even been formulated in XSLT. The essential fact is the wide range of possibilities that avoids fixation on one language.
A crucial extension that has to be introduced is a mechanism for distinguishing regular Web documents from those conforming to this model. To achieve this specifiability a HTTP header extension has been introduced. By using the header's field-name “Agent-Host” the origin of the agent can be indicated and implicitly the distinctiveness of the HTTP request is pointed out. The extension can be seen in .
After a Pneuma has moved to the place of execution it is able to use functionality offered by the Soma. Some of the fundamental functionalities are described below. Before a document based agent migrates to one host it can check if the target host offers all necessary functionalities needed for the execution. This is done by a HTTP GET request to a predefined URI. The reply to this request contains an XML description of the host's properties including all offered abilities. It especially covers the language and method support. When a Pneuma has executed it is not able to change the state of the execution environment but only to change itself or further accessible Web resources.
After checking for necessary runtime support the document agent can migrate by using standard HTTP methods – especially the PUT method for placing a resource at

a given URI is used for that purpose. It can be used at arbitrary points of execution for relocating itself. The new location can be communicated back to the server of origin by a POST request.

```
PUT /cgi/agentinterface#321 HTTP/1.1\r\n
Host: www.someexample.ws\r\n
Connection: Keep-Alive\r\n
Accept-Encoding: gzip\r\n
Accept: text/xml,application/xml,text/html,*/*\r\n
Accept-Language: en-gb\r\n
User-Agent: Mozilla/5.0\r\n
Content-type: text/javascript\r\n
Content-length: 541\r\n
Agent-Host: www.examplehost.ws\r\n
\r\n
function aggregateDocuments(){
...
}
```

Fig. 1. Example for an HTTP request containing a mobile agent

3.2 Considerations for the Soma

The Soma is used on a host to bring the code inside the Pneuma to execution. There are two different types of these execution environments:

1. Permanent available Soma
2. Dynamical loaded temporary Soma

The permanent available Soma is located on Web servers. Beside representing the execution environment for mobile agents it manages communication with Web browser that have been bound to the server through its permanent Soma. It is accessible by a constant URI. If this URI is concatenated with the predefined fragment identifier “soma” an associated downloadable temporary Soma is addressed. For example if the Web server's execution environment is available at http://example.com/place a Web browser can request the temporary Soma through http://example.com/place#soma. After requesting the temporary Soma it is deployed on the Web browser and can realize execution of mobile agents.

A very important mechanism is the binding process between a permanent Soma and various temporary ones. When a temporary execution environment is requested by a Web browser the associated original Soma on the Web server generates a so-called temporary URI (tURI) before the request is answered. A tURI consists out of its associated URI plus a unique ID attached as fragment identifier. The unique ID can be generated by an arbitrary algorithm.

For communicating the generated tURI to the associated Web browser a further HTTP header extension has been introduced: “T-Uri”. When the request for a Soma is answered, this header-field is included to the reply message indicating the particular

tURI. All requests that are addressed to this tURI are forwarded by the Web server to the bound Web browser. Furthermore the Web server operates as proxy for all bound Web browsers. These two requirements – the addressability by tURI and the proxy functionality - enable a direct browser to browser communication and therefore the exchange of mobile agents between them. It is possible to generate or modify a Pneuma on one Web browser and transmit it for execution on a further Web browser.
Out of a technical point of view the interaction between client and server is based on the non-blocking interaction mechanism Ajax or on HTTP streaming. Due to this, data that has been transmitted to the tURI by a third instance, which is requested in fixed intervals by polling or is directly forwarded to the Web browser if it is attached to the server via HTTP streaming. The overall mechanism of fetching a temporary Soma and starting communication is shown in Fig. 2.

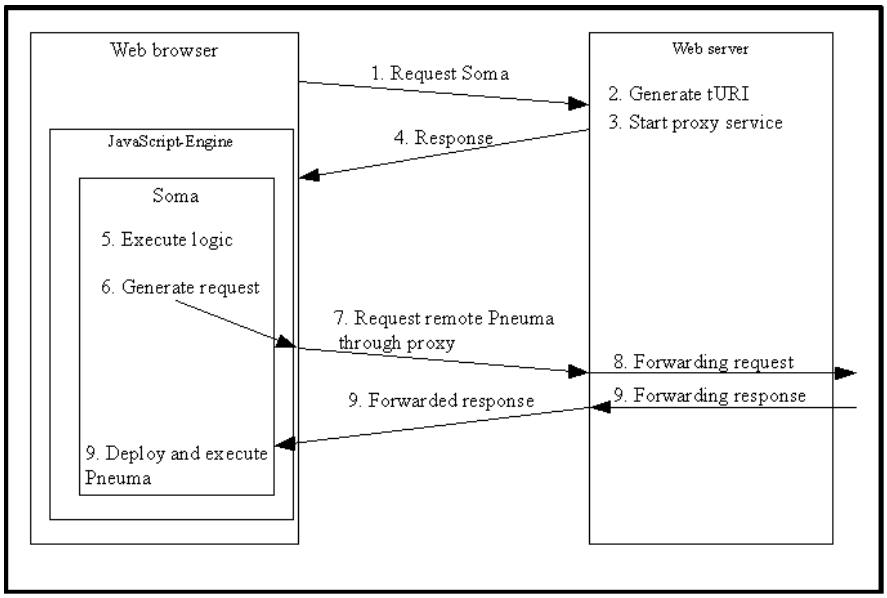


Fig. 2. Fetching a temporary Soma and request a Pneuma

When the functionalities offered by the Soma are discussed, it has to be differentiated between the permanent and the temporary one. The temporary execution environment has to be formulated in JavaScript thus making it deployable on a JavaScript-enabled Web browser. Hence the temporary Soma offers all general purpose features available in regular JavaScript programs run on Web browsers. This includes DOM manipulation and initiating HTTP based communication with remote resources.
The permanent execution environment is more complex and has to fulfill a broader range of functionalities. It does not only have the task of executing Pneumas transmitted to it, it also has to manage the temporary Somata and attached tURIs. Moreover it is responsible for realizing the proxy functionality. The Soma needs to make available a special XML description for indicating its capabilities. It is published under the URI of the execution environment plus the fragment identifier “capability”. Furthermore it has to offer the possibility for realizing HTTP interaction

with further Web servers. The permanent execution environment can be implemented in arbitrary languages and on basis of various technologies that support the realization of requirements. These include Common Gateway Interfaces, Java Server Pages or even Enterprise Java Beans.

3 Application Examples

The model has been validated and proven by various application scenarios based on its implementation. Direct browser to browser communication and exchange of data and logic between them makes a remote DOM manipulation possible which may be used in applications for remote control of Web browsers. By this a user can control the view and page presented to another user. This is suitable for support and educational purposes .
Beside direct Web browser interaction important application areas are Web proxy functionalities and document aggregation. In the case of proxy functionality a Pneuma can be located to a Soma where it functions as proxy for forwarding requests to its destination and delivering the replies to the requesting instance. By this it is feasible to first transmit such a Pneuma to a Web browser where fundamental parameters such as Quality of Service (QoS) parameters are specified. After this it migrates to an appropriate host and fulfills its work. It permanently checks the compliance of QoS parameters and if they are not accomplished the Pneuma migrates to an other host and fulfills its work from there. This dynamic proxy functionality might be useful to bypass location dependent adaptations of Web applications such as returning a set of pages of a Web search depending if the requesting instance is located in Poland or in the USA.
The last considered application scenario is the aggregation of Web documents on remote hosts. A Pneuma can be instructed by a Web browser to migrate to a Web server or another Web browser and to request information from different sources. These information are aggregated to one document and delivered to the initiating Web browser. Furthermore it is possible that the Pneuma does some filtering of content or is instructed to search for and monitor information sources by itself. This application scenario is especially useful if the Web browser is running on a host with only low bandwidth available. It is particularly feasible when requesting different Web feeds for example in Atom format. This shifts the information and feed aggregation away from the edges of the Web (the Web browsers) to its center (the Web servers).

4 Conclusion and Future Work

The work has shown that it is possible to develop a mobile agent infrastructure based on Web technology. For this minimal extensions have been defined. They all can be realized with standard Web principles and mechanisms. Particularly the expendable protocol HTTP is a powerful basis for mobile agent interaction and migration.
The model that has been developed relies on a high degree of interoperability with other Web applications and thus allows integration of them. The use of document based mobile agents allows the implementation of various feasible applications. These applications can be used by regular JavaScript enabled browser. Due to the

document centric view a high level of abstraction is achieved. Furthermore the user profits from a high degree of transparency. Especially the mentioned remote control and the proxy functionality can be realized without additional software installation and without expensive configuration steps. The extension of mobile agents' principles to the document centric World Wide Web may be one of the building blocks for realization the vision described by the buzzword 'the Web as platform'.
Future work will focus more intensively on security mechanisms. Especially validating an agent's origin has to be considered in more detail. Additionally it should be possible to confine the URIs which a particular agent might access. Furthermore an XML format for indicating the capabilities of a Soma has to be specified. After completing this work an efficiency analysis for different application examples will be provided.

References

1. Lange D. B., Oshima M.: *Seven good reasons for mobile agents,* Commun. ACM, Vol. 42. ACM Press (1999) 88-89.
2. Roth V.: *Obstacles to the adoption of mobile agents,* 5th IEEE International Conference on Mobile Data Management (MDM 2004), Berkeley, CA, USA (2004) 296-297.
3. Vigna G.: *Mobile agents: Ten reasons for failure,* In Proceedings of MDM Berkeley, CA (2004) 298–299.
4. Gray R. S.: *Agent Tcl: A flexible and secure mobile-agent system,* Fourth Annual {Tcl/Tk} Workshop (1996).
5. Garrett J. J.: Ajax: A New Approach to Web Applications. Adaptive Path. Online article (2005)
`http://www.adaptivepath.com/publications/essays/archives/000385.php`
6. Berners-Lee T., Fielding R., Masinter L.: *Uniform Resource Identifier (URI): Generic Syntax,* RFC 3986 (2005).
7. Chang S.-K., Znati T.: *Adlet: An Active Document Abstraction for Multimedia Information Fusion,* IEEE Transactions on Knowledge and Data Engineering, Vol. 13, no. 1 (2001) 112-123.
8. Bompani L., Ciancarini P., Vitali F.: *Active Documents in XML,* ACM SigWeb Newsletter 8 (1) (1999) 27-32.
9. Novatchev D.: *Functional programming in XSLT using the FXSL library,* Extreme Markup Languages 2003 (Montréal, Québec)
10. Fielding R. T.: *Architectural Styles and the Design of Network-based Software Architectures,* Doctoral dissertation, University of California, Irvine (2000).
11. Rescola E.: *HTTP Over TLS,* RFC 2818 (2000).

Proceedings of the International Multiconference on
Computer Science and Information Technology, pp. 47 – 57
ISSN 1896-7094
© 2007 PIPS

Utilization of Software Agents and Web Services as Transducers for Legacy Software; Case Study Based on an SMTP Server

Michal Oglodek
Faculty of Mathematics and Information Sciences
Warsaw University of Technology
Pl. Politechniki 1
00-661 Warsaw, Poland
moglodek@o2.pl
Maciej Gawinecki, Marcin Paprzycki
Systems Research Institute
Polish Academy of Science
ul. Newelska 6
01-447 Warsaw, Poland

Abstract. There exists a number of ways in which legacy software can be "wrapped" to become interoperable. Two of currently more popular of them are utilization of Web Services and software agents. The aim of this paper is to experimentally compare efficiency of JADE implemented agents, with Web Services, when used as transducers for an SMTP server.

Keywords: Web Services, Agent-based Computing, Transducers, Legacy Software, Simple Mail Transfer Protocol, Efficiency.

1 Introduction

One of interesting problems in software development is how to deal with legacy software developed using different technologies. Even if it sometimes good to keep and application "isolated," there is also a need for applications to communicate with each other (be *interoperable*). For instance, while companies may opt to develop their own software standards, there are at least two situations where communicating with "other" applications is necessary. First, in the case of company-to-company communication (e. g. when a wholesaler establishes direct link with a retailer). While this case may be relatively easy to solve by generating a few interfaces, the second scenario is more complicated. Nowadays, companies merge on regular basis, resulting in merging their independently created IT support systems, as most likely the IT system for the "new" company will not be created from scratch.

Let us also note that the question is not only how different (sub)systems can "talk to each other," but it is also important to consider effect that the attempted integration will have on exposing directly internal or external access to a given application. It is easy to envision, that security concerns (and resulting practical matters) need to be taken into consideration (e.g. an open port needed to access an application is likely to result in strict firewall policies). To illustrate this, in Figure 1 we present a problem

brought about by an external access to a Java Application. If the company firewall policy is very strict about open ports, the RMI (which is necessary for the external client to communicate with the application) is not easily accessible from the outside.

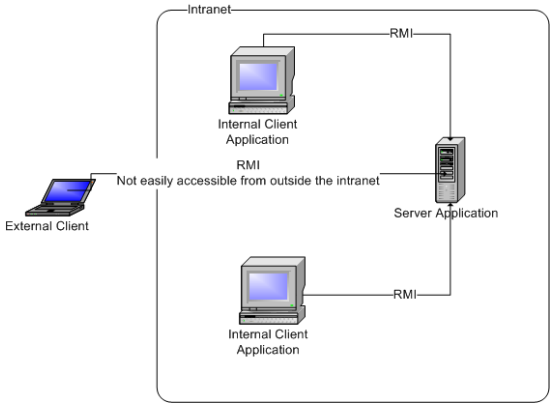


Fig. 1. External access to the application using Java RMI

Here, at least the following concerns regarding direct RMI connection arise:

- firewalls blocking the RMI (company policy, users using public free Internet, or Internet c afés could have an access problem),
- securing data transmission (sending data / man-in-the-middle attacks),
- no language / platform independence (while a Web Service can be used by clients written in many different languages).

Nowadays, *wrappers* and *transducers* are the most popular ways of approaching these problems. Here, a *transducer* is component pattern, which accepts messages from a requesting system, translates them into the program's native communication protocol, and passes to an application [2]. It also accepts responses, translates them back and forwards to the requester. A *wrapper*, on the other hand, can "inject" code into an application to allow it to communicate with the requester. In other words, it can directly invoke API methods, and examine and modify data structures of the program [2]. Both wrappers and transducers can secure (limit, authorize, etc.) access to the remote software; extend its functionality and improve performance [4].

In this paper we explore possibilities brought about by utilization of Web Services and software agents as transducers. Both approaches fit into this role as they use message-oriented communication and are platform independent. Specifically, we show how they can be used to expose access to an SMTP server. Note that transducers are used as we do not have direct access to the server (code injection and method invocation), and we can relay only on messaging utilizing SMTP.

We proceed as follows. In the next section we briefly introduce Web Services and software Agents as well as discuss the scenario used in our experiments. We follow with a head-to-head comparison of both approaches.

2 The scenario

The aim of this paper is to compare Web Services and software agents used as transducers to expose an SMTP server. Here we follow the general approach proposed in [14, 15], where we have experimented with performance of software agents in selected scenarios. Our goal is perform an initial assessment which of the two approaches is likely to be better to "glue applications." Obviously, our aim is only to establish some initial guidelines as to pros and cons of both approaches. We start our description from introducing the SMTP protocol and its weaknesses.

2.1 Simple Mail Transfer Protocol – weak points

The Simple Mail Transfer Protocol (SMTP) is a standardized protocol for e-mail transmission [5, 6]. It uses relatively simple, text-based communication. Its simplicity is advantageous but results in lack of *flexibility*, and in *security* and *performance* concerns. Let us look into them in some detail.

Security concerns. Sendmail is one of the first mail transfer agents utilizing the SMTP protocol. It listens for connections using port 25. History of sendmail shows how exposing port 25 can lead to serious vulnerabilities allowing hackers access to the shell of the machine running the sendmail/SMTP Server [8]. Therefore, standard security policies mandate closing port 25. Note that the SMTP-AUTH extension [7], attempts at assuring that only authorized users are able to send messages (reducing the spamming problem). While hosting companies require authentication, typical client applications use simple Base64 algorithm for encrypting authentication data. This does not protect against eavesdroppers and man-in-the middle attacks

```
S: 220 www.example.com ESMTP Postfix
C: HELO mydomain.com
S: 250 Hello mydomain.com
C: MAIL FROM:<sender@mydomain.com>
S: 250 Ok
C: RCPT TO:<friend@example.com>
S: 250 Ok
C: DATA
S: 354 End data with <CR><LF>.<CR><LF>
C: Subject: test message
C: From: sender@mydomain.com
C: To: friend@example.com
C:
C: Hello,
C: This is a test.
C: Goodbye.
C: .
S: 250 Ok: queued as 12345
C: QUIT
S: 221 Bye
```

Fig. 2. Example of SMTP communication: C is a client, S is server.

Lack of flexibility and performance issues. Let us use Figure 2, an example of communication between of a client and the SMTP server. Here, the client wants to send an e-mail from the `sender@mydomain.com` address to the `friend@example.com`. During this session client sends 182 characters to the SMPT server and receives 132 characters back. Real content (sender and recipient addresses, and the message body) contains only 90 characters, which means that the overhead is 204 characters (92 characters of commands, and 132 of response). While this is a relatively small cost in the case of sending a single message to a single recipient, it becomes substantial in the case of 1000+ recipients and large attachment, as in the following scenario:

> *An accountant in a corporation needs to send salary information to all employees. Company uses an external provider for e-mail accounts. The message contains a long legal text as well as a few big attachments – overall, its size is about 500kB. There are 1000 employees and each should receive customized information.*

Usage of the SMTP protocol would require sending 1000 separate e-mails and would transferr approximately 1000*500 kB = 500000 kB (around 500 MB of data).

2.2 Extending SMTP functionality with a transducer

Using a transducer it is possible to extend the SMTP server by allowing for *MailTemplate* messaging. The idea is to allow client to send (to the SMTP server) message body (in our scenario 500kB) in a form of a template (with parameter placeholders) and a set of parameters to be inserted before the mail is actually sent. This allows for e-mail customization as well as limits use of the network (between the mail client and the SMTP server). In this work we compare performance of four possible approaches to communicating with the SMTP server:

- (A1) direct SMTP communication with an external SMTP server,
- (A2) direct SMTP communication with an internal SMTP server,
- (A3) communication using the *MailTemplate* through a transducer realized by Web Service deployed on the remote machine with the SMTP Server,
- (A4) communication using the *MailTemplate* through a transducer realized by a software agent deployed on the remote machine with SMTP Server.

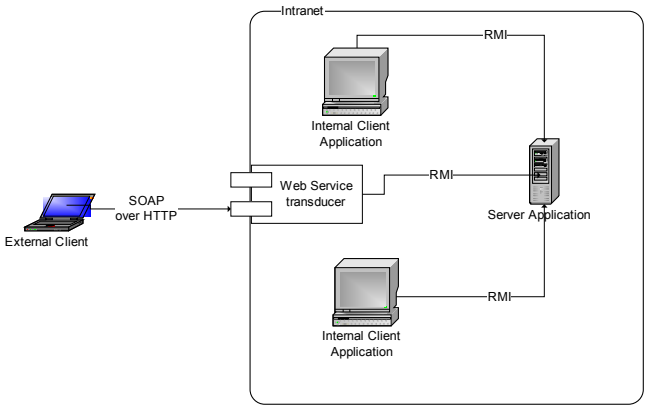


Fig. 3. Secure external access to the application using Web Service over HTTP

The Web Service transducer is deployed on the machine running the SMTP Server, or within the local network where the SMTP Server is located. Web Service utilizes standard functionality of the SMTP Server through direct communication and exposes the extended functionality to the external clients (see Figure 3). Note that the main advantage of Web Services is conformity with most firewall policies and platform independence. At the same time, their basic drawback is overhead introduced by use of SOAP data structures. We will return to these issues in what follows.

The agent-based transducer (*AgentReceiver*), located at the SMTP server, exposes its functionality to other agents. It uses FIPA ACL messaging to communicate with *AgentSender,* which forwards user requests. In our scenario *AgentSender* prepares the template, set of parameters and list of receivers; serializes them and sends it, as an ACL message, to the *AgentReceiver*. *AgentReceiver* deserializes message, acts as a mail-merger that prepares messages (for the SMTP server), authenticates the user and sends messages out. Furthermore, *AgentReceiver* informs *AgentSender* whether sending e-mails was successful. The other direction of communication mirrors this scenario. Here, we assume that both agents reside within different platforms, and thus utilize an agnostic communication protocol (e.g. HTTPS). Finally, note that in our work we use JADE (Java Agent Development Framework) [9] to implement agents.

3 Comparing Web Services and software agents as transducers

When comparing the four possible approaches we consider their following features:

- broadly understood security (including authentication and authorization),
- message overhead (in the “accountant scenario”),
- flexibility (in the “accountant scenario”),
- performance,
- easiness of implementation and deployment.

3.1 Comparing security

As specified in Section 2.1, the approaches (A1) and (A2) are not really secure. More precisely, direct access to the SMTP server results in:

a) lack of protection against the man-in-the-middle attack,
b) lack of protection against retrieving the content of the message by sniffers,
c) lack of protection against retrieving and decrypting user name and password (used for authentication with the SMTP server),
d) lack of conformity with most company security policies,
e) access to the local network of the remote machine.

The latter problem can be solved by closing all ports, but the port for the text-based HTTP protocol (as most company firewall policies do). A transducer can prevent exposing the application and the local network to direct attacks. Other problems can be addressed by use of the secure HTTPS protocol. Note that both JADE agents and Web Services can use HTTP and HTTPS protocols [12, 13]. The potential drawback is that, for JADE agents, message transfer based on HTTPS is about 15% slower than the HTTP MTP [12]. Similarly, the SSL for Web Services will generate overhead [16]. However, one could resign from using SSL to secure the entire transport, and use a solution that provides security on the message level: the WS-Security [13].

3.2 Comparing message overhead

Both the SMTP and the Web Service generate overhead related to headers. As shown in Figure 2, sending a single mail results in about 204 characters of overhead. In order to establish overhead of Web Services, let us consider the following simple SMTP transducer used for mail sending (in Figure 4). The approximate overhead of the SOAP header request is 480 bytes.

Performance and flexibility gains come from utilization of a Web Service transducer that implements the *MailTemplate* solution and is used in the scenario of an accountant sending e-mails to employees. Here, instead of sending 500 MB of data (+ the SMTP overhead) in the standard SMTP approach, only about 500 kB (+parameters +SOAP overhead) of data will be sent.

Let us now consider an ACL message sent by the *SenderAgent* to the *ReceiverAgent* (see Figure 5). As we can see, the message content is serialized and encoded using the Base64 algorithm (used as default by JADE). The ACL Message header contains 336 characters (non-bold text in Figure 5). Here, (the same as above) 90 characters are contained within a Java object send as a content slot of the ACL message. Serialization of this object generates additional overhead of 415 characters. Next, JADE applies Base64 encoding to the serialized object and as a result output contains 756 characters. Therefore, the total overhead is 1002 characters (more than in the case of a Web Service request).

```
POST /SmtpTransducer.asmx HTTP/1.1
Host: www.example.com
Content-Type: application/soap+xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
  <soap12:Body>
    <SendE-mail xmlns="http://tempuri.org/">
      <from>sender@mydomain.com</from>
      <to>friend@mydomain.com</to>
      <subject>test message</subject>
      <body>Hello,
            This is a test.
            Goodbye.
      </body>
    </SendE-mail>
  </soap12:Body>
</soap12:Envelope>
```

Fig. 4. Example of SOAP Communication for *MailTemplate* request

```
(REQUEST
  :sender  (agent-identifier
    :name SmtpSender@abc123:1099/JADE
    :addresses (sequence http://192.168.1.100:7778/acc )
    :X-JADE-agent-classname oglodek.project.SmtpSenderAgent )
  :receiver  (set ( agent-identifier
     :name SmtpReceiver@corn:1999/JADE
     :addresses (sequence http://corn.bunge.pl:7778/acc )) )
  :X-JADE-Encoding Base64
  :content  " .... Base64 ENCODED VALUE ..... "
```

Fig. 5. Example of ACL Message for *MailTemplate* request sent from the *SenderAgent* to the *ReceiverAgent*

Table below summarizes the overhead of each considered approach:

	Direct access (A1 and A2)	Access via Web Service (A3)	Access via JADE agent (A4)
C haracters sent by the client	182	570	1092
Size of actual content	90	90	90
Overhead	204	480	1002

3.3 Note on (im)possible further flexibility

Note that the *MailTemplate* approach could benefits from an assumption that the *MailTemplate* is not be a data structure, but a mobile executable code, generating a set of messages to be sent. Such mobile code would be carried by the *AgentReceiver* , migrating from a client machine to the SMTP server. This would allow configuring mechanism for generating templates at runtime. However, this solution has a set of problems of its own:

- mobile agent could be dangerous for a target hosting platform (see concept of malicious code, summarized in [10]),
- overhead of sending template and an agent itself higher than in cases A3 and A4,
- required homogeneity of origin and target platforms; as a result (for the time being) *AgentReceiver* would be able to migrate only within JADE platform [11].

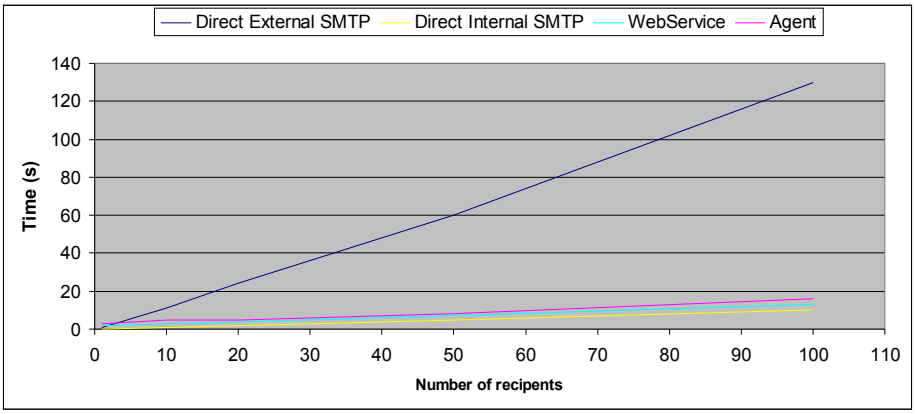
Therefore, we have decided to not to pursue this possible solution further.

3.4 Comparing performance

We examined performance of all four approaches by comparing time of sending e-mail messages of size *s* to *n* recipients (one e-mail per recipient). To be precise, by the *size of a message* we mean:

a) in approaches with direct access to the SMTP server (A1 and A2): the average size of an e-mail message body (without header)
b) in approaches with the *MailTemplate* (A3 and A4): size of the message body (with parameter placeholders), plus vector of parameters, without header.

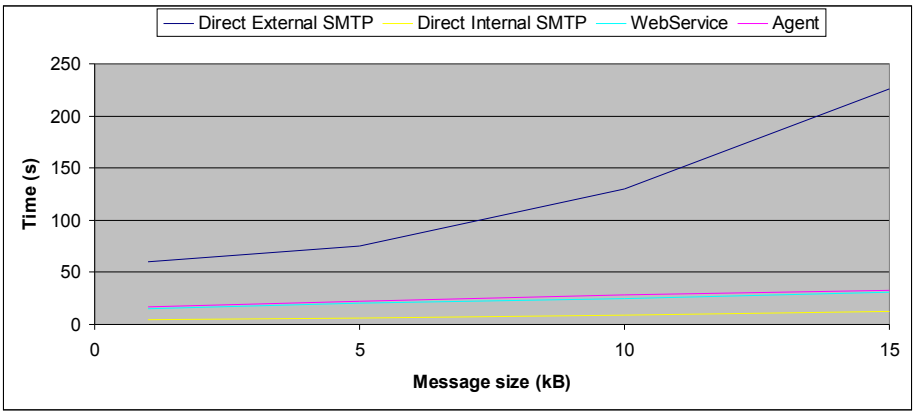
In approaches A1 and A2, *n* messages were send to the SMTP server, whereas in approaches A3 and A4, a single *MessageTemplate* was send to the transducer, which translated it into *n* requests to the SMTP server. Under these assumptions we have performed two tests. First, we studied the relation between completion time and number of recipients (n), assuming constant message size (s ~10 kB). Second, the number of recipients was fixed ($n = 1000$), while the relation between sending time and message size (s) was evaluated. Both tests were performed on the system consisting of two PC's: (1) PC with 2 GHz Intel Core Duo; 2.0 GB RAM; running Microsoft Windows XP; it hosted client applications (in approaches A1-A3) or the *AgentSender* ; (2) 4200+ AMD Athlon 64 X2 Dual; 4.0 GB RAM; running Microsoft Windows Server 2003 R2 64 bit; it hosted the ESMPT Mail Service (version 6.0.3790.3959) and transducers (in A3 and A4): the *AgentReceiver* or the WebService. The WebService was implemented in C# .NET under Microsoft .NET Framework 2.0 and was hosted by the IIS 7 web server. Agents have been implemented in JDK 1.6, and run within JADE 3.5 framework. Finally, both machines were located in a LAN with 1 Mbit/s download and 512 kbit/s upload bandwidths. The results of the tests are presented on Figures 6 and 7, respectively.



Number of recipients	Direct External SMTP	Direct Internal SMTP	Web Service	Agent
1	1.04	0.14	2.13	3.02
10	11.21	1.18	3.21	4.35
20	24.38	2.23	4.42	5.53
50	60.92	5.56	7.58	8.77
100	128.89	10.98	13.64	15.21

Fig. 6. Relation between time (seconds) and number of recipients

We can observe a very large difference between results obtained with the external SMTP server (A1) and the remaining approaches. This was to be expected as the external SMTP server requires sending *n* messages across the network. The situation changes with the internal SMTP server, where messages are generated "internally" and no extra time is used sending them (over the network) to the SMTP server. As a result this solution is the fastest. It is however similar to utilization of transducers, where the message template is send once to the remote server and the transducer communicates with the SMTP server locally. The only difference is time used by SOAP / ACL messaging, and overhead of transducer processing messages. We can also notice that the Web Service (A3) is slightly more efficient than using software agents (A4). This can be explained by the bigger overhead of an ACL message request comparing to a SOAP request. Technology used for implementation of Web Service (.NET on IIS 7.0) versus Agents and HTTP MTP (Java with internal engine acting as the HTTP server) can also have some impact on performance.



Message size (kB)	Direct External SMTP	Direct Internal SMTP	Web Service	Agent
1	59.42	4.67	15.12	17.43
5	76.38	6.53	20.41	22.21
10	127.65	9.28	25.32	27.15
15	226.21	12.43	31.26	33.13

Fig. 7. Relation between time (seconds) and message size.

3.5 Implementation and deployment issues

Most of modern development tools allow easy use of Web Services. For example Visual Studio can generate proxy class for a given URL of a Web Service so that one does not need to deal with SOAP message creation. Similarly, all serialization / deserialization of objects in a SOAP message will be done automatically by the proxy class. Hence the developer that has no knowledge about SOAP headers or SMTP commands can start working with them without any additional learning required. In the case of the SMTP server one needs to know only the specification of the SMTP protocol – commands, syntax, order of commands, etc. At the same time implementing software agents with JADE, especially behavioral programming, implementing interaction protocols and developing communication ontologies, can be somewhat more difficult for users that are working with agents for the first time.

4. Conclusions

In this paper we have compared two transducer-based approaches to wrapping legacy software. One utilized Web Services, while the other was based on software agents.

We have fund that (a) utilization of either of the transducers can have clear positive effects on performance, flexibility and security of legacy software; and (b) Web Services seem to be able to handle messaging somewhat more efficiently, by introducing less overhead. The next step that should be undertaken is extending the breadth of scenarios under which the performance comparison is undertaken.

References

1. M. Gawinecki, M. Kruszyk, M.Paprzycki, M.Ganzha (2007) *Pitfalls of agent system development o the basis of a Travel Support System,* In Proceedings of the BIS 2007 Conference (to appear), http://agentlab.swps.edu.pl/agent_papers/BIS_2007.pdf
2. Micheal R. Genesereth, Steven P. Ketchpel (1994), *Software Agents,* In Communication of the ACM, Vol. 37, No. 7 July. http://citeseer.ist.psu.edu/genesereth94software.html
3. S.Srivatsa Sivan and R. Venkatavaradan (2005), *Design Guidelines: Building Web Service Wrappers for an XML-based System,* http://www.devx.com/enterprise/Article/27882
4. Design Patterns, *Elements of Reusable Object-Oriented Software,* Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides , Addison Wesley.
5. RFC 821, http://www.faqs.org/ftp/rfc/pdf/rfc821.txt.pdf
6. RFC 1123, http://www.faqs.org/ftp/rfc/pdf/rfc1123.txt.pdf
7. RFC 2554, http://www.faqs.org/ftp/rfc/pdf/rfc2554.txt.pdf
8. *Hacking: The Art of Exploitation,* 2nd Edition, Jon Erickson.
9. JADE (Java Agent DEvelopment Framework), http://jade.tilab.com/
10. Łukasz Nitschke, Marcin Paprzycki, Michał Ren, *Mobile Agent Security,* http://agentlab.swps.edu.pl/agent_papers/NATO_2006.pdf
11. Inter-Platform Mobility Project, https://tao.uab.cat/ipmp/
12. Jose A. Exposito, Joan Ametller, Sergi Robles , *How to use the new HTTP MTP with JADE,* http://jade.tilab.com/doc/tutorials/HTTP_UAB.html
13. Mike Lehman Securing Web Services, http://www.oracle.com/technology/oramag/oracle/05-jan/o15web.html
14. K. Chmiel, M. Gawinecki, P. Kaczmarek, M. Szymczak, M. Paprzycki: *Efficiency of JADE Agent Platform,* Scientific Programming, vol. 13, no. 2, 2005, 159-172.
15. K. Chmiel, D. Tomiak, M. Gawinecki, P. Kaczmarek, M. Szymczak, M. Paprzycki: *Testing the Efficiency of JADE Agent Platform,* In: *Proceedings of the ISPDC 2004 Conference,* IEEE Computer Society Press, Los Alamitos, CA, 49-57.
16. Use of SSL Creates Performance Overhead for Browsers, http://support.microsoft.com/kb/15003

Proceedings of the International Multiconference on
Computer Science and Information Technology pp. 59–69
ISSN 1896-7094
© 2007 PIPS

An Architecture for Resource Bounded Agents

Sławomir Nowaczyk and Jacek Malec

Department of Computer Science, Lund University
Slawomir.Nowaczyk@cs.lth.se
Jacek.Malec@cs.lth.se

Abstract. We study agents situated in partially observable environments, who do not have sufficient resources to create conformant (complete) plans. Instead, they create plans which are conditional and partial, execute or simulate them, and learn from experience to evaluate their quality. Our agents employ an incomplete symbolic deduction system based on Active Logic and Situation Calculus for reasoning about actions and their consequences. An Inductive Logic Programming algorithm generalises observations and deduced knowledge so that the agents can choose the best plan for execution.
We describe an architecture which allows ideas and solutions from several subfields of Artificial Intelligence to be joined together in a controlled and manageable way. In our opinion, no situated agent can achieve true rationality without using at least logical reasoning and learning. In practice, it is clear that pure logic is not able to cope with all the requirements put on reasoning, thus more domain-specific solutions, like planners, are also necessary. Finally, any realistic agent needs a reactive module to meet demands of dynamic environments.
Our architecture is designed in such a way that those three elements interact in order to complement each other's weaknesses and reinforce each other's strengths.

1 Introduction

Rational, autonomous agents able to survive and achieve their goals in dynamic, only partially observable environments are the ultimate dream of AI research since its beginning. Quite a lot has already been done towards achieving that dream, but dynamic domains still remain a major challenge for autonomous systems. In particular, nontrivial environments that are only partially observable pose demands which are beyond the current state of the art, possibly except when dedicated, hand-crafted solutions are developed for specific domains.

One of the major ways of coping with uncertainty and lack of knowledge about current situation is to exploit previous experience. In our research we are interested in developing rational, situated agents that are aware of their own limitations and can take them into account, as brilliantly presented by Chong and others in [1]. To facilitate this, we use Active Logic [2] for knowledge representation, which characterises reasoning as an ongoing process, instead of focusing on a fixed point of entailment relation.

Due to limited resources and to the necessity to stay responsive in a dynamic world, situated agents cannot be expected to create a complete plan for achieving their goals. In theoretical AI a common approach is to create a *conformant plan*, i.e. a plan which contains provisions for any possible sequence of external events and observations, and

which is guaranteed to reach the goal in all scenarios. For situated agents, however, not only the task of *creating*, but even a requirement to simply *store* such plan could easily exceed available resources.

The lack of resources is partly addressed by considering "information-providing" actions and interleaving their execution with planning activity. In particular, executing them at the right time allows the agent to greatly simplify its subsequent planning process—it no longer needs to take into account the vast number of possible situations which would be inconsistent with newly observed state of the world. Thus, it can proceed further in a more effective way, by devoting its computational resources to more relevant tasks.

Therefore, situated agents need to consciously alternate between reasoning, acting and observing their environment, or even do all those things in parallel. We aim to achieve this by making the agents create short partial plans and execute them, learning more about their surroundings throughout the process. They create several partial plans and reason about usefulness of each one, including what knowledge can it provide. They generalise their past experience to evaluate the likelihood of any particular plan leading to the goal. The plans are conditional (i.e. actions to be taken depend on observations made during execution), which makes them more generic and means that their quality can be estimated more meaningfully. We also intend for the agent to judge by itself whether it is more beneficial to begin executing one of those plans immediately or rather to continue deliberation and, possibly, develop longer and more complete plans, in order to avoid making an unrecoverable mistake.

We expect the agent to live significantly longer than the duration of any single planning episode, so it should generalise solutions it finds. In particular, the agent needs to extract domain-dependent control knowledge and use it when solving subsequent, similar problem instances. It is the authors' belief that deductive knowledge, at least in many of the domains we are interested in, may contain more details and be more accurate than other forms of representation (such as numerical or probabilistic), therefore our agent learns deductively using a symbolic representation in Active Logic. To this end we introduce an architecture consisting of four modules, which allow us to combine state-of-the-art solutions from several fields of Artificial Intelligence, in order to provide the synergy our agent needs to achieve the desired functionality.

Ultimately, the agent will need to be able to handle non-stationary, adversary environment, to cooperate with others in multi-agent setting and to plan for goals more complex than simple reachability properties, such as temporally extended goals and restoration goals. It is our belief that symbolic knowledge representations are the only way of achieving such versatility.

The goal of this paper is to present and justify the architecture we use for our agents, as well as suggest possible ways to extend it. In the next section we introduce example domains which we exploit to present and test our ideas. In section *Architecture* we provide an overview of the organisation of our agent. The following four sections introduce each of agent's functional modules in more detail: *Deductor*, *Planner*, *Actor* and *Learner*. After that, we describe the module interaction in some more detail, briefly present some of the *Results* of our experiments with the architecture, discuss some of the *Related Work* and finish with some *Conclusions*.

2 Experimental Domains

Throughout this paper we will be using two simple domains, to better illustrate our ideas. The first one is a simple game called Wumpus [13]. The game is easy to understand and people have no problems playing it effectively as soon as they learn the rules. For artificial agents, however, this game—and other similar applications, including many of practical importance—remain a serious challenge.

The game is played on a square board. There are two characters, the player and the monster called Wumpus. The player can, in each turn, move to any neighbouring square, while the Wumpus does not move at all. Position of the monster is not known to the player, he only knows that it hides somewhere on the board. Luckily, Wumpus is a smelly beast, so whenever the player enters some square, he can immediately notice if the creature is in the vicinity. The goal of the game is to find out the exact location of the monster, by moving throughout the board and observing on which squares does it smell. At the same time, if the player enters the square occupied by the beast, he gets eaten and loses the game.

For learning experiments we also use a second domain, a modified version of "king and rook vs king and knight" chess ending. Since we are interested in partially unknown environments we assume, for the sake of experimentations, that the agent does not know how the opponent's king is allowed to move—*a priori*, any move may be legal. The agent will need to use learning and discover what kinds of moves are actually possible.

Those domains are, obviously, only examples and the architecture presented here does not dependent on them. In order to better understand the goal of our research, it can be helpful to imagine the setting similar to the *General Game Playing Competition* [3]: our agent is given some declarative knowledge about the domain and is supposed to act rationally from the very beginning, while becoming more and more proficient as it gathers more experience.

3 Architecture

The architecture we propose consists of four main modules (shown in Figure 1). *Deductor* corresponds to a typical "core" of logically reasoning agent, except that in our case it does not use classical logic, but rather Active Logic formalism in order to better interact with other modules. *Planner* module is mainly responsible for creating potentially interesting plans, although any kind of domain-specific reasoning could just as well be performed in it. Its main purpose is to increase the amount of agent's beliefs (including plans) that provide Deductor with knowledge to reason about.

Actor is an overseer of the interactions between an agent and its environment. It analyses and interprets agent's sensor data, in order to react whenever something interesting happens in the external world. It also watches over agent's reasoning process and makes decisions about when their results are sufficiently well developed to begin acting based on them.

Finally, *Learner* analyses the performance of the agent as a whole and generalises its past experience in order to improve the work of each module. In particular, the most important part is to evaluate conditional partial plans and to learn which ones are most likely to lead to good results in future situations.

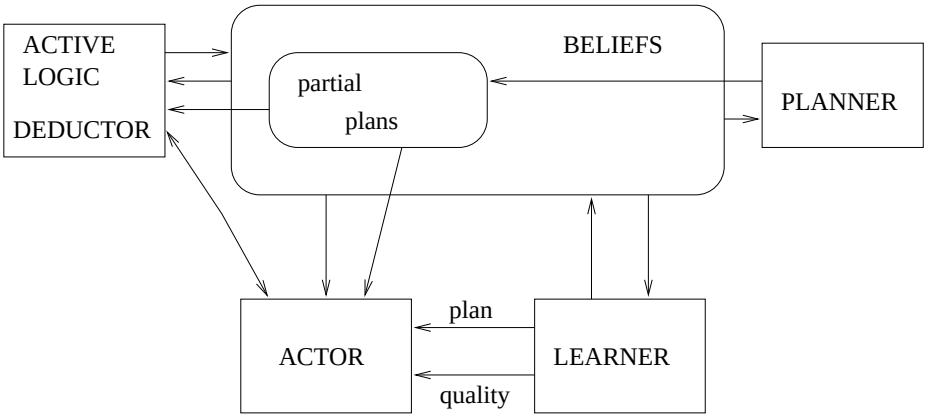


Fig. 1. Architecture of an agent.

In the following sections, we describe each module in more detail, as well as point out the most important interactions among them.

4 Deductor

This module analyses both current state of the world and how it will change as a result of performing a particular action. Our agent uses a variant of Active Logic [2], augmented with some ideas from Situation Calculus [12].

Active Logic is a reasoning formalism which, unlike classical logic, is concerned with the *process* of performing inferences, not just the final outcome (fixed point) of the entailment relation. In particular, instead of classical notion of theoremhood, AL has i*-theorems*, i.e. formulae which can be proven *in* i *steps*. This allows an agent to reason, among other things, about *difficulty* of proving something, to retract knowledge found inappropriate and to resolve contradictions in a meaningful way, as well as makes it aware of the passage time and its own non-omniscience.

To this end, each formula in AL is labelled with the step number when it was derived. E.g., the *modus ponens* inference rule looks like this: $\frac{i:\alpha, \alpha \Rightarrow \beta}{i+1:\beta}$ and means "if at step i formulae α and $\alpha \Rightarrow \beta$ are present in the belief set, then at step $i+1$ formula β will also be present." Moreover, there is a special inference rule $\frac{i:Now(i)}{i+1:Now(i+1)}$, which allows an agent to refer to the current moment and to explicitly follow the passage of time. A more in-depth description of Active Logic can be found in [11].

Following ideas of [8] we have decided to augment Active Logic with some concepts from Situation Calculus. In particular, in order to have the agent reason about changing world, every formula is indexed with current situation. Furthermore, since the agent needs to reason about effects of executing various plans, we additionally index formulae with the plan the agent is considering. Therefore, a typical formula our agent reasons about looks like this: $Knows(s, p, Neighbour(a2, b2))$ and means "an

agent knows that after executing plan p in situation s, squares $a2$ and $b2$ will be adjacent[1]." This formula is only mildly interesting, as its validity depends on neither s nor p, at least in the Wumpus domain. But: $Knows(s, p, \neg Wumpus(b2))$ which means "an agent knows that after executing plan p in situation s, Wumpus will not be on $b2$," *does*, obviously, depend on s, since agent's knowledge changes as it acts in the world. It still does not, however, depend on p itself. Clearly, no "new" knowledge can be obtained by simply *considering* some plan (without actually executing it). If an agent $Knows(s, p, \neg Wumpus(b2))$, then it must also $Know(s, \emptyset, \neg Wumpus(b2))$, where $\emptyset$ stands for an empty plan[2].

In contrast, an example of some really interesting formulae that can be formulated could be

$$Knows(s, p, Wumpus(b3)) \vee Knows(s, p, Wumpus(c2))$$

which means "an agent knows that after executing plan p in situation s, it will *either* know that there is Wumpus on $b3$ or that there is Wumpus on $c2$". As an example of reasoning by cases and predicting action results, this is exactly the kind of knowledge that we want the agent to infer—it *does* tell important things about quality of the plan being considered. If all the agent knew before was: $Knows(s, \emptyset, Wumpus(b3) \vee Wumpus(c2))$ than clearly executing p is useful. For a human "expert," such p looks like a good plan. The goal of our research is to make *an agent* be able to reason about plans in exactly this way. It is our intuition, supported by preliminary experiments reported later in this paper, that creating a *domain independent* Actor module which would efficiently select good plans by learning from experience using formulae like the one above is possible.

5 Planner

As we stated earlier, our agent creates conditional, partial plans. The plans are partial because limited resources do not allow our agent to consider all the possibilities and come up with a good conformant plan. The plans are conditional since we intend the agent to learn that some of them are generally good and some of them are generally bad. By the virtue of being conditional, the plans remain concise but also strictly more expressive than in the unconditional case. In particular, their applicability is significantly enhanced and the agent is able to reason about their usefulness.

Let us take an example from the Wumpus domain. With an agent on square $a1$, one simple plan is "$a2$", meaning "go to $a2$", while another is "$a2a3$", meaning "go to $a2$ and then go to $a3$". A conditional plan could be "$a2\ ?\ a1 : b2$", meaning "go to $a2$ and if it smells there go back to $a1$, else go forward to $b2$". Actually, in Wumpus domain it is difficult to find a simple plan longer than one step which would be good, while there exists a number of conditional plans which can be easily classified as good.

[1] We use chess-like notation for naming squares, with letters designing columns and numbers designing rows.

[2] Since in our game the player has no way of changing Wumpus' position, the actual validity of "$Wumpus(b2)$" remains constant, only agent's knowledge is changing

In a sense, our treatment of plans is related to the notion of hierarchical planning, since the conditional partial plans we consider are very similar to macro-operators [4]. Our goal is to let the agent learn which conditional partial plans are *good* to later use them as building blocks for finding complete solutions.

6 Actor

The Actor module acts as a controller of the agent as a whole. It has three kinds of duties. First, it observes the environment and forces the agent to react to interesting events that take place there. Second, it decides when enough time has been spent on deliberation and no further interesting results are likely to be obtained. Finally, it makes decisions to execute a particular plan from Deductor's repertoire.

The typical scenario consists of Actor continuously monitoring the sensor input, analysing it and transforming it into symbolic representation whenever needed. In Active Logic, there exists a special provision for that, called *observation function*, with exactly the same status as domain axioms, but with different temporal extent. Such input can then be used by the Deductor, allowing an agent to respond to the changes in the world. It is also possible for Actor to react more "violently" if needs arises, either by directly and immediately performing some physical actions (e.g. obstacle avoidance if the agent is about to hit something) or by straightforwardly influencing agent's internal reasoning process (for example, if Actor notices that the change of the environment reaches some threshold, it can "reset" Deductor, under the assumption that its previous results are no longer applicable anyway). In this sense, the Actor implements the reactive behaviour of the agent.

In addition, Actor continuously monitors the progress of agent's internal reasoning and can react to certain conditions there. For example, as soon as Deductor finds a plan which looks sufficiently good (in the extreme case, one which directly leads to the goal), the reasoning can be interrupted and this plan executed immediately. Similarly, if it turns out that the deduction cannot proceed further before some observation is made, Actor can decide to pause the reasoning and perform the necessary sensing actions.

Finally, Actor chooses which plan to execute. This decision may be based on plan ranking done earlier by either Deductor or Learner, or may simply be a random pick, if a situation is sufficiently new and such knowledge is not yet available.

7 Learner

The goal of the learning module is to provide Actor with knowledge necessary to choose the best plan for execution and to decide when to stop deliberation—i.e. when too much time has been spent on it without reaching any new, interesting insights.

In order to do that, Learner finds out recipes for evaluating plans and to decide which ones are most likely to lead to the goal. Since the plans are partial, it is very difficult to predict, in general, whether a given plan is a step in the right direction. There is an active research on heuristics in planning, focusing on exactly this problem.

Using learning techniques developed in the field of Inductive Logic Programming, however, is another possible way to approach the problem. Based on experience and

on deductive reasoning performed by Deductor module, it is possible to analyse how the world (and the agent's knowledge about it) will change after executing a particular plan. It is then possible to learn rules for determining the class of plans which have been successful in the past, and use that to choose the one to be executed next.

In the simplest case, it could consist of just distinguishing "dangerous" plans, i.e. ones that can lead to agent's immediate death. This is not enough in general, but there is a large class of domains which are "safe" in a sense that it is possible to win from every position. Wumpus domain belongs to this class, while chess does not.

Moreover, an important question is one of credit assignment, since the agent typically executes several partial plans before it reaches a terminal state of the game. Only the complete sequence of actions is then rewarded or punished. It can very well happen that one of the plans in a bad sequence was, in fact, good. There are numerous techniques, each with own advantages and disadvantages, being developed for dealing with this problem, and it is not clear at this point which one would be best suited for our particular case.

8 Module Interactions

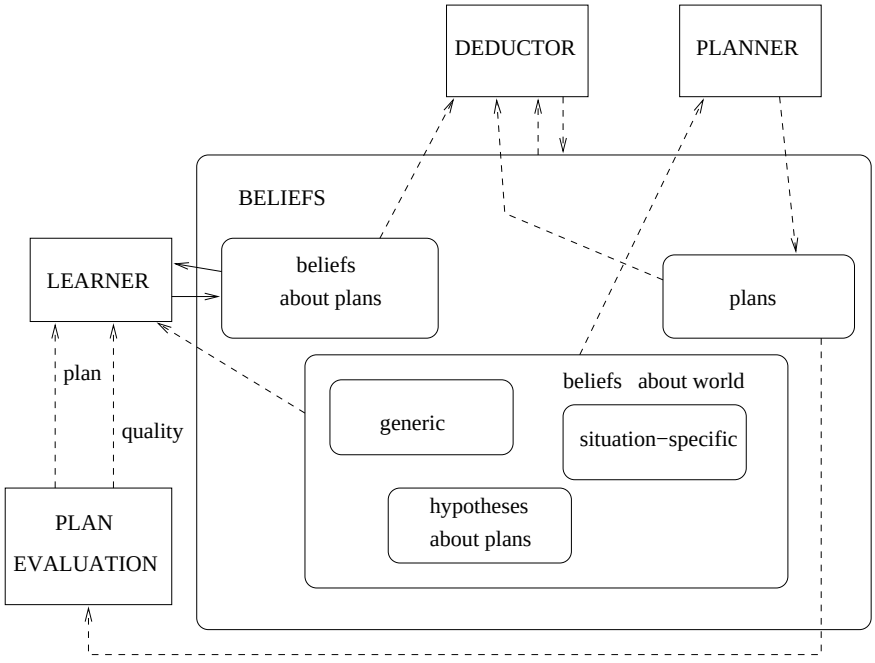


Fig. 2. Some details of the architecture of the agent.

The main idea of our architecture is to ensure fruitful interactions between the modules introduced above. Each of them roughly corresponds to one of the major subfields of Artificial Intelligence and is designed to employ state of the art solutions from it. Each module can provide a very good performance in specific situations, but neither is quite sufficient to achieve real intelligent behaviour of a generic situated agent.

The major contribution of our architecture is the idea to use multiple conditional partial plans, together with expected results of their application to a particular situation, as the way to exchange information between modules. In this way, Planner comes up with plans which are potentially interesting, but it does not need to commit to a single one. Deductor reasons about each of those plans separately, but is also able to explore interactions between them, as well as to make use of any similarities it can find in order to extrapolate results concerning one plan to the others. Learner induces generic knowledge about what types of plans have been successful in the past. Actor predicts, based on that, which should be executed in the future, as well as oversees the reasoning and chooses the best course of action as soon as enough knowledge becomes available.

Using the above architecture, agent is able to reason about current state of the world, both about the details of current situation and about the generic laws governing the application domain. Moreover, it can also reason about the state of the world as it expects it to be after execution of each plan under consideration. Finally, it reasons bout *plans* themselves, how successful were they in the past, both in general and in situations similar to the current one, and how good they are expected to be right now.

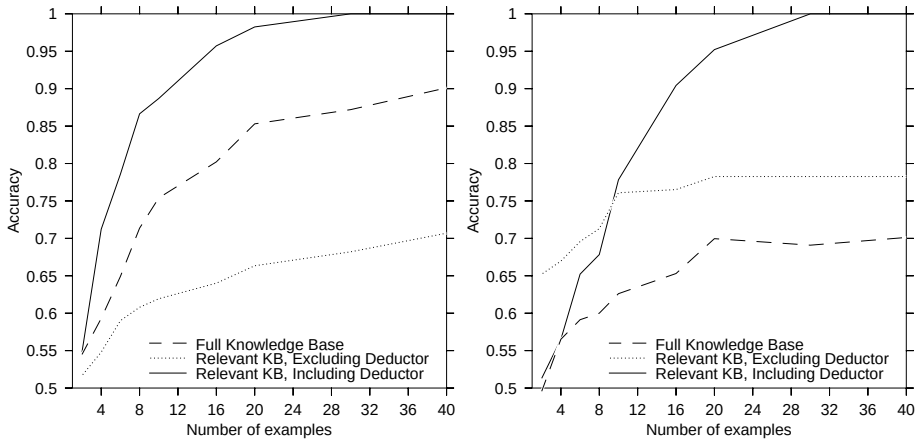
9 Results of Initial Experiments

We have performed some initial experiments in order to evaluate the feasibility of our ideas and to check how well do they work in practice. Our focus was on interactions between modules and on showing that different approaches we combine do indeed complement each other.

We have used an ILP algorithm PROGOL [7] for learning, since it is among the best known ones and its author has provided a fully-functional, publicly available implementation. In a previous paper [10] we have presented results of learning to distinguish “bad” plans early. We have shown that PROGOL is able to find the correct hypothesis from as few as 30 randomly-chosen examples. Such a hypothesis allows the agent to save up to 70% of its reasoning time, since it does not need to waste resources analysing plans which are known to be useless. Those results required providing additional domain knowledge specifically for the purpose of learning, but we have also analysed (in [9]) how the PROGOL learning algorithm can be adapted to extract it automatically.

The results we have obtained can be seen in figure 3. The first curve (middle one, marked “Full Knowledge Base”) corresponds to the case where we used no additional domain-specific knowledge, except from PROGOL *mode declarations*, and where the whole knowledge the agent has gathered has been provided to the learning algorithm.

For the remaining two curves we identified the parts of agent’s knowledge which are most relevant to learning the concept of bad plans and only presented those to the learning algorithm. In the lowest one, marked “Relevant KB, Excluding Deductor”, we have provided only domain description and observations, having removed all the results



(a) In Wumpus domain. (b) In Chess domain.

Fig. 3. Results of learning.

of Deductor's work. In the highest one, we have left them intact. Only in this latest case the Learner has been able to correctly learn the concept.

Those results show that Deductor provides knowledge which significantly improves quality of learning. At the same time, as mentioned above, learning the right hypothesis allows the agent to save a lot of its reasoning effort. We plan further experiments which shall reveal even more synergy between modules in the architecture.

10 Related Work

The amount of work on agent architectures in general is too large to summarise comprehensively here, therefore we will only discuss a small fraction of this field here.

The work presented in this paper is related to studies of architectures for general intelligence. There has been a lot of attempts to define such an architecture, with SOAR being the most prominent and most successful of them ([6] provides a general introduction). In contrast to SOAR, we do not claim any cognitive plausibility of our architecture, focusing our interests on mainly on the task of achieving intelligent behaviour of an artificial agent.

Our approach is also quite distinct from the layered architectures which are popular nowadays (see, for example, [5] for a collection of papers on this topic). In particular, we do not focus specifically on the reactive part of the system, hiding it one part of the Actor module. This does not mean that we diminish the necessity or importance of the reactivity, but rather that we simply decided to concentrate on the higher-level reasoning aspect of our agent as it is less understood and requires more attention. We make sure, however, that higher levels of our architecture remain sufficiently flexible to be able to handle the requirements of reactive part.

Although we have been stressing the need of rationality and the importance of reasoning throughout this paper, our approach also differs from the one exhibited by Beliefs-Desires-Intentions systems and BDI architectures (see, among others, [14] for a formalised approach to this topic). We do not explicitly distinguish intentions, while both plans and goals are treated in a very similar manner. This is due to the fact, however, that BDI approaches usually lack learning, which is the central issue in our approach, and which allows an agent to account for its intentions in a different, but also sufficiently effective, way.

11 Conclusions

In this paper we present an architecture for rational agents we are currently developing. It allows agents to combine planning, deductive reasoning, inductive learning and time-awareness in order to operate successfully in dynamic environments. Agents create conditional partial plans, reason about their consequences using an extension of Active Logic with Situation Calculus features, and employ ILP learning to generalise past experience in an attempt to distinguish good plans from bad ones.

The basic framework of the architecture is already implemented and initial experiments show that the synergy effects we aimed for do, indeed, appear. The modules themselves still need more work before they contain the functionality needed to tackle practical problems, since we have only been working on toy ones up to now. State of the art solutions, however, are available and many of them can be integrated into our architecture with minimal changes. Nevertheless, the ones needing further attention are efficient Deductor for Active Logic, and Learner using modified ILP algorithm. Our current research focuses on those topics.

References

1. W. Chong, M. O'Donovan-Anderson, Y. Okamoto, and D. Perlis. Seven days in the life of a robotic agent. In *GSFC/JPL Workshop on Radical Agent Concepts*, 2002.
2. Jennifer Elgot-Drapkin, Sarit Kraus, Michael Miller, Madhura Nirkhe, and Donald Perlis. Active logics: A unified formal approach to episodic reasoning. Technical report, University of Maryland, 1999.
3. Michael Genesereth, Nathaniel Love, and Barney Pell. General game playing: Overview of the aaai competition. *AI Magazine*, 26(2):62–72, 2005.
4. Malik Ghallab, Dana Nau, and Paolo Traverso. *Automated Planning: Theory and Practice*. Morgan Kaufmann, 2004.
5. David Kortenkamp, R. Peter Bonasso, and Robin Murphy, editors. *Artificial Intelligence and Mobile Robots*. AAAI Press / MIT Press, 1998.
6. R. L. Lewis. Cognitive theory, SOAR. *International Encylopedia of the Social and Behavioral Sciences. Amsterdam*, 2001.
7. Stephen Muggleton. Inverse entailment and Progol. *New Generation Computing, Special issue on Inductive Logic Programming*, 13(3-4):245–286, 1995.
8. Sławomir Nowaczyk. Partial planning for situated agents based on active logic. In *Workshop on Logics for Resource Bounded Agents, ESSLLI 2006*, 2006.

9. Sławomir Nowaczyk and Jacek Malec. Inductive logic programming algorithm for estimating quality of partial plans. In *Mexican International Conference on Artificial Intelligence*, 2007.
10. Sławomir Nowaczyk and Jacek Malec. Learning to evaluate conditional partial plans. In *Sixth International Conference on Machine Learning and Applications*, 2007.
11. Khemdut Purang, Darsana Purushothaman, David Traum, Carl Andersen, and Donald Perlis. Practical reasoning and plan execution with active logic. In John Bell, editor, *Proceedings of the IJCAI-99 Workshop on Practical Reasoning and Rationality*, pages 30–38, 1999.
12. Raymond Reiter. *Knowledge in Action: Logical Foundations for Specifying and Implementing Dynamical Systems*. The MIT Press, 2001.
13. Stuart Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall Series in AI, 2nd edition, 2003.
14. Michael Wooldridge. *Reasoning about Rational Agents*. MIT Press, 2000.

Proceedings of the International Multiconference on
Computer Science and Information Technology, pp. 71 – 81
ISSN 1896-7094
© 2007 PIPS

Employment of mobile multiagent systems for attendance of transaction of distributed database

Zofia Kruczkiewicz[1]

[1] The Institute of Computer Engineering,
Control and Robotics,
Wroclaw University of Technology,
Wrocław, Poland
zofia.kruczkiewicz@pwr.wroc.pl

Abstract. It is aimed at the limitation of the network traffic which is generated during the realization of *two-phase-commit* protocol of distributed transactions in distributed database systems. This paper presents the way of the decrease of the number and the way of the diminution of the size of messages sent during distributed transactions as the manner of the limitation of the network traffic. Multiagent system is employed for attendance of confirmation of two-phase distributed transactions. The paper presents the manner of the limitation of the network traffic in multiagent systems as the use of mobile agents as resources managers of databases, static agents as coordinators of transactions and static agents as managers of transactions. These agents communicate via *two-phase-commit* protocols. The paper presents the performance experiments enabling the comparison of *two-phase-commit* protocols, the protocol elaborated in the paper and the protocol described in the literature.

1 Introduction

General applicable distributed computerized information systems store information in distributed heterogeneous databases. It is aimed at such efficient work of these systems, which generate the network traffic as small as possible. This system should have the high performance. Therefore mechanisms of the mobility and the intelligent behavior of employed software are used in these systems. The base of the maintenance of the indivisibility of realized distributed transactions of distributed heterogeneous databases, the immunity from states of the disconnection and movements of the part of systems, is the *two-phase-commit* protocol. It is described multiagent systems for attendance of confirmation of two-phase validation of distributed transactions in the paper. The paper presents the *two-phase-commit* protocol based on mobile agents, which is projected during the analyses of the MAS. This protocol includes the limited number and the least size of messages and it generates the smaller traffic than the protocol presented in the literature [13].

The concept and the implementation of activities and communication between agents play the basic role in the creation of multiagent systems (MASs). It results from the basic paradigm of MAS, which is the social character of a system i.e. the

autonomy, the adaptation and interactions. The transmission of information in MASs is based on conversations [11], which include interactions and activities. These interactions and activities affect on the performance of the system and the network traffic. Performance engineering is engaged on the problems of the performance of software, during all phases of its creation. It is possible to value the performance of the system and the size of sent information in the network as time parameters of the system. Therefore it is possible to perform the choice of arbitrated solutions due to the time parameters of the model of MAS, which are matched to degree of the development of the software [1, 2].

It puts the stress on limitation of the size and the number of messages of protocols of created MASs. The mobility of software is one of the manners of this limitation, which relies on transmission of mobile agent as autonomous unit of software [6]. The mobile agent is relocated in the network of type of WAN or LAN, it decides about its activities, it executes services of the sending system. It is differed following patterns of interaction protocols: the static protocol (behaviors of agents and interactions are not changed) and the mobile protocol (behaviors of agents and interactions are changed). The choice of the mobility results from the type of an application and it leads to the minimization of the network traffic, the cost of CPU, generalizations of services, the autonomy of components of software, the higher immunity from errors of components moved in the network than from errors of components of the client–server architecture, the elasticity of management of data and the encapsulation of the protocol. Several defined patterns of communications based on the mobility are compared in work [6]: client-server (CS*, Client-Server*), remote start-up of code - (REV*, Remote Evaluation*), transmission of code (COD*, Code Demand*) and mobile agent (MA*, Mobile Agent*). The work [4] presents methods of the improvement of the performance and the network traffic of MAS ACORN, which rely on the dynamic grouping of mobile agents in cafes due to their information and the introduction of the idea of the agent architecture as the *Thick/Thin* agents. The *Thin* agent is the mobile agent, which collects information during movement in the network and sends this information to the static agent of the type of the *Thick* agent. The works [3], [8] compare the performance of patterns of interaction protocols of CS and MA. It defines the sequence and parallel architecture for these patterns [8] and researches their performance in the respect of the performance of *Voyager, Aglets* and *Concordia*. The work [11] compares static and mobile MASs, which are equivalent in the respect of realized tasks. This paper describes the research of the performance of the serialization of agents as the mechanism of the mobility.

The intelligent and efficient behavior of MAS enables limitation of the network traffic generated during distributed transactions of heterogeneous databases in the distributed environment. The review of systems of attendance of mobile transaction is presented in the work [12]. These systems are *Clustering, Two-Tier Replication, Pro-motion, Reporting, Semi-based, Prewrite, Kangaroo Transaction, Multidatabase Transaction Processing Manager.* The work [12] performs the comparison of operation models, ownership ACID, cohesions, transparency, stability, reactions on disconnect and mobility of elements of the system and it serves the proposal of the system for the attendance of the mobile transaction (MTS, *Mobile Transaction Service*). The work [13] presents the role of agents in improvement of the safety of distributed transactions of the system based on the technology CORBA. The work

[10] describes the architecture of *Distributed Database Management Systems* (DBMSs), which employs the tool named CaseLP in specifying elements and testing of behavior of the system. The work [9] presents pro-active MAS, which is based on static and mobile agents which support distributed transactions of the system. These mobile agents improve the performance in comparison with classic solutions applying mobile agents. The system is realized as the platform based on *IBM Aglets* and the virtual machine of java (JVM) in the environment of *Aglets Software Development Kit* (ASDK).

The second chapter of work introduces MAS JADE. It presents protocols of validation of two-phase transactions in the fourth chapter. The fifth chapter serves the results of performance experiments concerning on comparison of two protocols i.e. the elaborated *two-phase-commit* protocol and the protocol presented in the work [13]. Work is ended with conclusions.

2 System MAS JADE

It chooses Java MAS JADE (*Java Agent Development Framework*) system [7] based on FIPA specification (*The Foundation for Intelligent Agents*) [5] for the implementation of *two-phase-commit* protocols. JADE includes packages for the creation of agents and the environment of their operations as the platform of agents (AP, *Agent Platform*) (fig. 1). AP platform consists of following agent components: *Agent Management System* (AMS), which fulfils the control over the access and the utilization of agent platform; *Directory Facilitator* (DF), which catalogues services of applications; *Message Transport System (MTS)*, which controls the sent messages; *Agent*, which represents applications. AP can be divided onto several machines as containers (fig. 1). Each container has the virtual machines of Java (JVM) and includes one or many interacting agents. The main container includes AMS and DF agents and the RMI register (*Remote Method Invocation*) which controls the mobility of messages and the optional mobility of agents of one AP. Containers are connected with the main container and they build the environment of operations of set of agents. According to FIPA specification, DF and AMS communicate by use of ACL (*Agent Content Language*) language, *Fipa-Agent-Management* ontology and *Fipa-Request* interaction protocol.

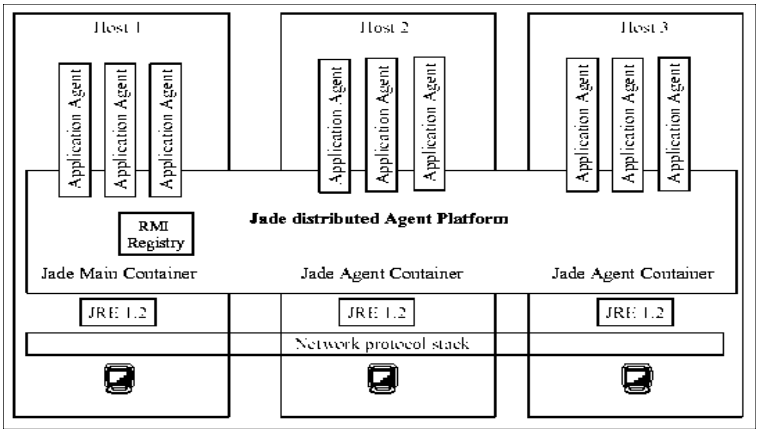


Fig. 1. Distributed Agent platform and JADE containers [7]

3. *Two-phase-commit* protocols based on static and mobile agents

Communication between agents is defined by the use of the interaction protocol (IP) [5]. System JADE offers the set of patterns of interaction protocols based on FIPA specification. These protocols are dedicated to different typical applications. JADE supports implementation of user-defined protocols. Patterns of protocols are placed in *FIPA Interaction Protocol Library Maintenance* [5]. It introduces new diagrams to design of interactions between agents, which are called protocol diagrams (*PD, Protocol Diagrams*). PD is based on UML sequences diagrams. It expresses new elements of PD diagrams in meta-UML language, named AUML (Agent UML) [5]. It presents IP protocols of exemplary applications at diagrams of fig.2-3.

The design system consists of following elements: environments of MAS JADE, the static agent as *Transaction Manager,* which administers transactions, static agents as *Coordinator Manager_1* and *Coordinator Manager_2,* which gather information of execution states of the fragments of distributed transaction and static or mobile agents as *Resources Manager_1*, *Resources Manager_2, Resources Manager_3* and *Resources Manager_4*, which oversee execution of fragments of distributed transaction realized in allocated resources. Agents act according to concept of Thick /Thin agent [4], i.e.:

- the static thick agent as *Resources Manager* owns the full knowledge of its environment, it is able to acquire, to process and to analyze information therefore it has own code of a big size
- the mobile thin agent as *Resources Manager* owns the limited knowledge of its environment and the limited range of operations, but it is able to migrate in network and it has own code of a small size.

The comparison of interaction protocols bases on the assumptions that they enable realization of the same task on the same data placed on same hosts and this protocol is the best in the point of generated network traffic, which allows to executing task of the smallest joint size of sent messages.

The IP1 protocol as *two-phase-commit* protocol (fig. 2) is used in the MASs presented in the work [13]. The thick agent as *Manager Transaction* sends a *begin_transaction* messages about beginning of transaction to the set of agents, which includes at least two agents i.e. *Coordinator Manager_1* and *Coordinator Manager_2*. Agents as *Coordinator Manager_1* and *Coordinator Manager_2* send messages about beginning of transaction (*request_1, request_2, request_3, request_4*) to static thick agents as *Resources Manager_1, Resources Manager_2, Resources Manager_3* and *Resources Manager_4*. Then agents as *Coordinator Manager_1* and *Coordinator Manager_2* send *can-commit* message about query of the fragment of the distributed transaction to agents as *Resources Manager_1, Resources Manager_2, Resources Manager_3* and *Resources Manager_4*. Each agent as *Resources Manager* confirms by sending *ok-commit* message about the correct result of the part of the transaction or *failure* message about the failure to the agent as *Coordinator Manager*. After receiving all answers, agents as *Coordinator Manager_1* and *Coordinator Manager_2* send *ok-commit* message to the agent as *Transaction Manager* if agents as *Resources Manager* confirm all correct parts of transaction or *failure* message, if at least the one agent as *Resources Manager* reports the failure. If the agent as *Transaction Manager* receives *ok-commit* messages from all agents as *Coordinator Manager*, it sends *do-commit* messages to them about the commit of the distributed transaction, in the opposite case if it receives at least one *failure* message, it sends *rollback* messages to all agents as *Coordinator Manager* about the come-back of the state of resources before the beginning of the transaction. In the first case agents as *Coordinator Managers* send *do-commit* messages to all agents as *Resources Manager*. Next agents as *Resources Manager* confirm by sending *have-commit* messages to agents as *Coordinators Manager* which send *have-commit* messages to the agent as *Transaction Manager*. In the second case agents as *Coordinator Manager* send *rollback* messages to agents as *Resources Manager.* Agents as *Resource Manager* confirm the failure of the transaction by sending *have-rollback* messages to agents as *Coordinator Manager.* Next agents as *Coordinators Manager* send *have-rollback* messages to the agent as *Transaction Manager* about confirmation of the failure of the transaction.

The protocol IP2 at the figure 3 differs from protocol IP1 at the figure 2 where two mobile thin agents are employed in overseeing resources of transaction. Instead of *request* and *can-commit* messages it is sent *request-to-move* messages with information of network migrations which are necessary for the execution of fragments of the transaction. Protocol IP2 does not include four *can-commit* messages, because mobile agents as *Resources Manager_1* and *Resources Manager_2* must not receive them however the network traffic appears when the mobile agents as *Resources Managers* migrate through network and execute all fragments of the transaction. The time of the realization of *request-to-move (request_1, request_2)* message consists of the time of the receiving this message by mobile agent as *Resources Manager_2* and two migrations to databases placed on computers of net addresses given by parameters *request_1* and *request _2*. The agent as *Resources Manager_2* sends *ok-commit* or *failure* message after realization of fragments of transactions in two databases to the agent as *Coordinator Manager_2* which sends *ok-commit* or *failure* message to the agent as *Transaction Manager*. During the second phase of transaction the agent as

Coordinator Manager_2 receives *do_commit* or *rollback* message and it sends one of these messages to the agent as *Resources Manager_2*. It migrates through two databases and commits or cancels the fragments of the transaction. Then it sends *have-commit* or *have-rollback* message to the agent as *Coordinator Manager_2*. The agent as *Coordinator Manager_2* sends *have-commit* or *have-rollback* message to the agent as *Transaction Manager.* The same conditions concern the agent as *Resources Manager_1.* The smaller network traffic is generated when the executing of the fragment of transaction fails during the first migration. The agent as *Resources Manager_1* and *Resources Manager_2* finishes the executing of the transaction without another migration and sends *failure* message to own agent as *Coordinator Manager*. During the second phase of the transaction it migrates to the first database and cancels the state of its resources. It decreases the number of migrations of agents from eight to four.

In general, the protocol with mobile agents includes n agents as *Coordinator Managers* and each agent as *Coordinator Manager* coordinates p agents as *Resources Manager* and each agent as *Resources Manager* migrates through q databases. The number q is greater than 1. The expression $p*q$ is equal to the number of agents as Resources Manager of IP1 protocol. It is taken the assumption that the size of all messages and the size of code of the each mobile agent are equal to a. Therefore the IP1 protocol includes the number of messages which is equal to the value of the following expression: $(4*n + 5*n*p*q)* a$. The IP2 protocol includes the number of messages which is equal to the following expression: $(4*n + 4*n*p+2*n*p*q)* a$.

The difference between these expressions is equals to value: $n*p*(3*q - 4)*a$. It means that the IP1 protocol has the greater joint size of messages than IP2 protocol. If the transaction of IP2 protocols fails during the first migration of the each mobile agent, its expression is equal to the following one: $(4*n + 6*n*p)* a$. Now the difference between two expressions is equal to $n*p*(5*q-6)$ and it is greater than the first difference. The IP2 protocol can generate the smaller network traffic than the IP1 one.

4 Performance experiments

It constructs equivalent static and mobile MASs due to the number of databases. MASs are based on IP1 and IP2 interaction protocols (fig. 2, fig. 3). Researching of the time of the answer is purpose of the experiment in comparison with the generated network traffic of the static and the mobile MASs i.e. the time that expires from the moment of the beginning of the task of searching out to the time of receiving the result of searching. The system based on thick agents of IP1 interaction protocol is presented on fig. 4 and system based on mobility thin agents based of IP2 interaction

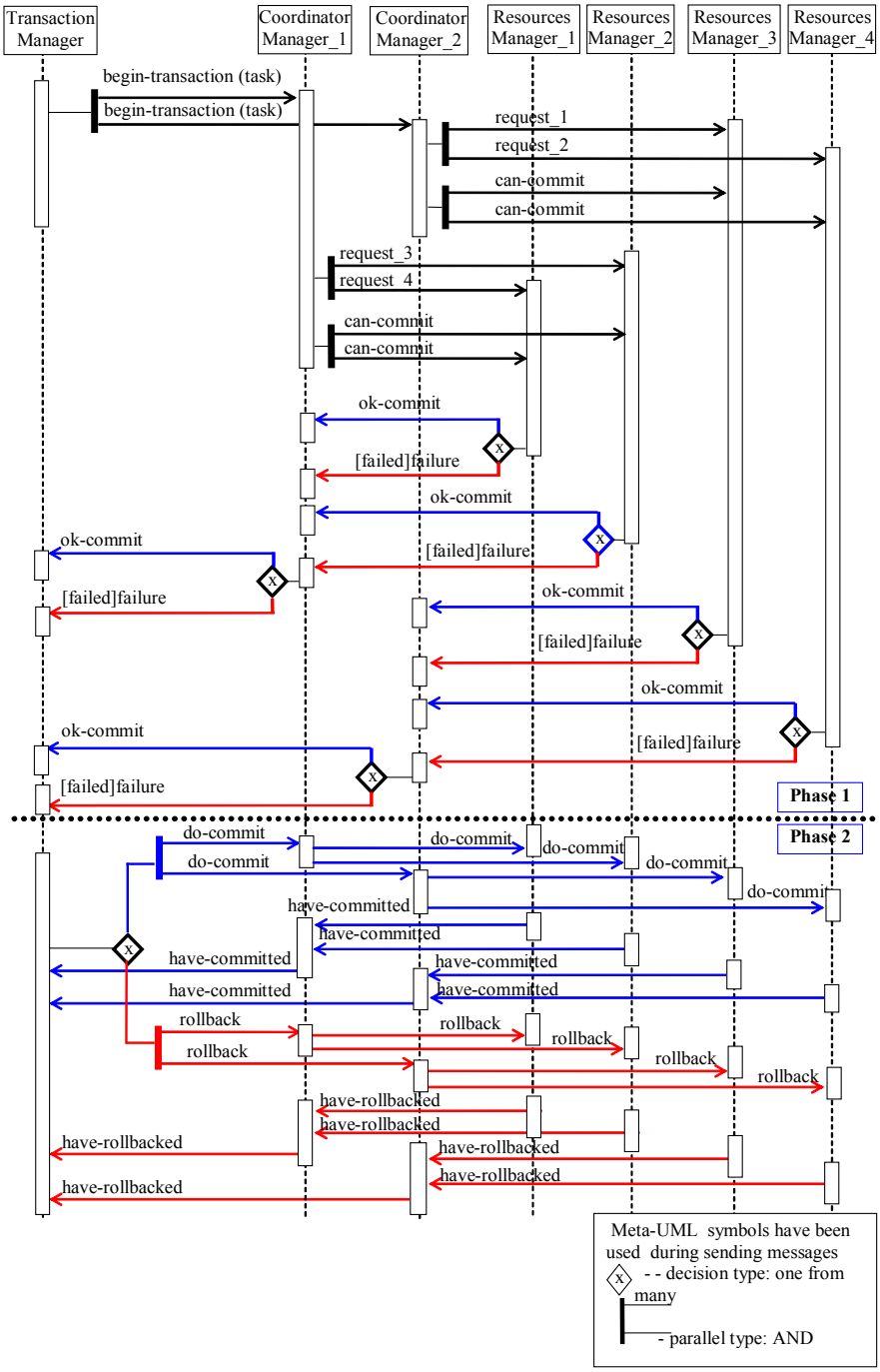


Fig. 2. IP1 protocol of validation of two-phase distributed transaction of MAS based on static agents [13]

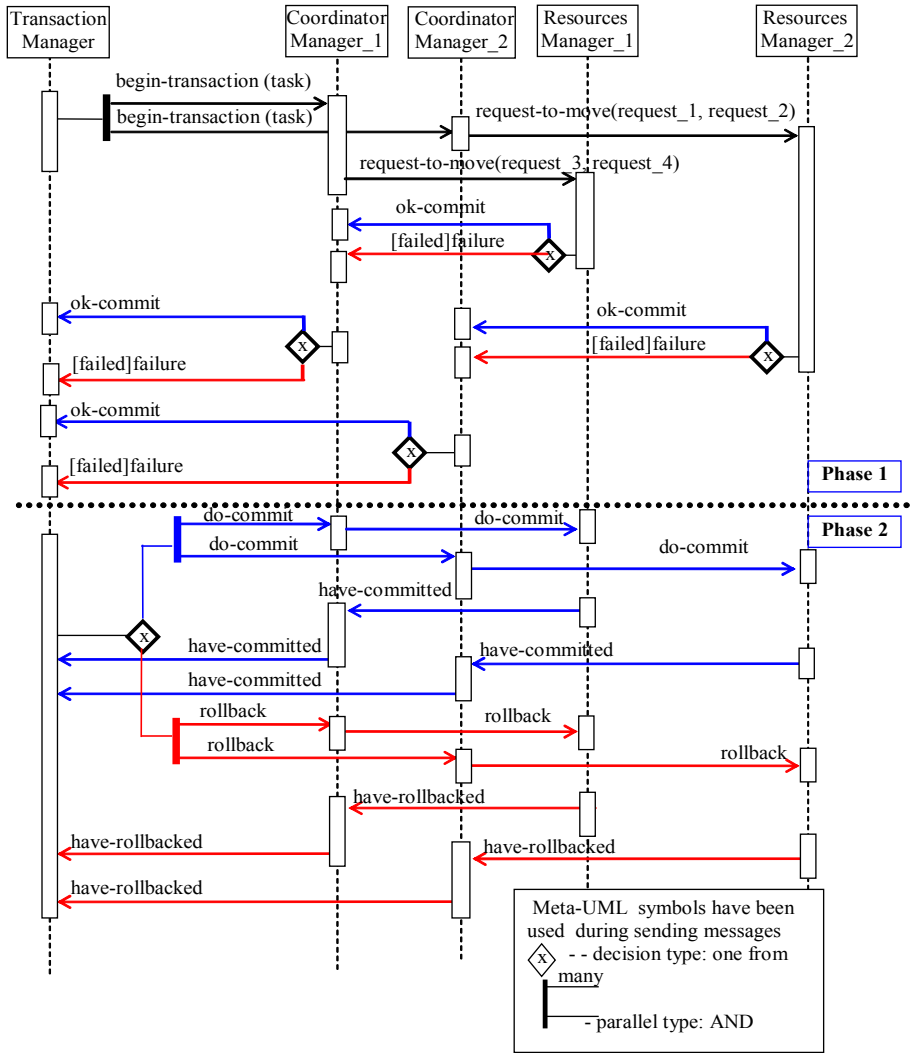


Fig. 3. IP2 protocol of validation of two-phase distributed transaction of MAS based on static and mobile agents

protocol is presented on fig. 5. Technical data of MASs are presented in tab.1. It carries research on computers of the local network acting on 100BASE-TX.

The agent is independent thread of virtual java machine (JVM) in system acting on JADE i.e. the independent thread of operating system, which works under control of JADE. Each agent can execute several independent tasks simultaneously. In order to build detached threads for the each agent, it introduces the concept of the behavior. This solution is efficient, because the transfer of the steering from the thread to the thread is hundred times slower than the calling method – each agent behavior is called

by the agent method [7]. Communication and mobility of agents are based on RMI of JAVA, which employs the migration of the code and the state of agents.

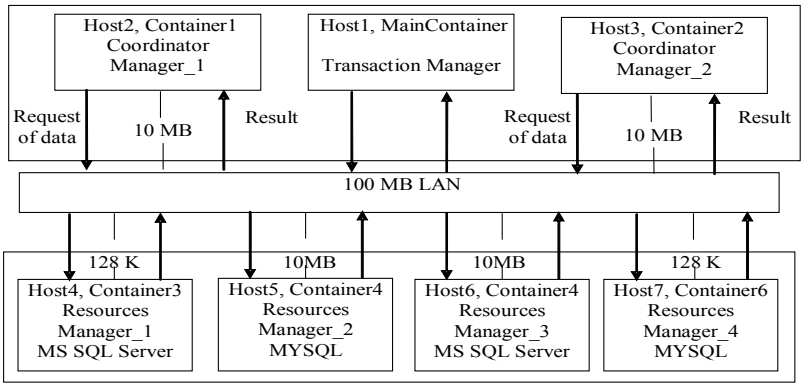


Fig. 4. Analysis of communication infrastructure system in the following environment: seven hosts, static solution based on IP1

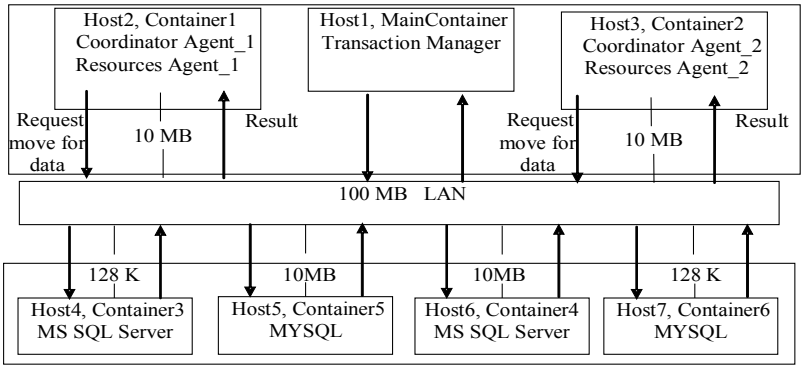


Fig. 5. Interaction infrastructure of MAS in the following environment: seven hosts, system IP2

Evaluated systems are researched without taking into consideration influence of the graphic interface of user. Agents as *Resources Manager* execute task of searching out on servers of databases systems MySQL 4.0.17 and MSQL Server 2000, with database of bookstore. Each of table stores 1000 records of data of clients of bookstores. Static and mobile agents as *Resources Manager* execute the SQL query: SELECT * FROM klienci WHERE nazwisko LIKE 'X%'. Realized query is not a transaction but this solution enables to evaluate the influence of IP1 and IP2 protocols on performance of two-phase distributed transactions. The time of each searching is gauged by the use of the java class method i.e. *System.currentTimeMillis*.

Tab. 1. Technical test machines data

	Hosts 1, 2	**Hosts 3, 4, 5, 7**
OS	MS Windows 2000	MS Windows XP
CPU	Celeron 1GHz	Athlon XP 2200+
RAM	120 MB	256 MB
JVM	Sun SDK 1.5	Sun JRE 1.5
JADE	3.4	3.4

Tab. 2. Time [ms] of searching of MASs after 1000 tests on 7 computers

Time [ms]	MAS based on IP1	MAS based on IP2
Average time	36751	29583
Max time	38587	32228
Min time	35670	28520

In the static solution each query is repeated for 250 times by each *Resources Manager*. In the mobile solution each *Resources Manager* migrates between two hosts for 250 times per the one host. The operation of the static agent is equivalent to the one migration of the mobile agent. It presents results of research in table 2.The group of mobile agents is faster than the group of static agents. This improvement of the performance of IP2 protocols results from the limited number of mobile agents and the smaller size of its code than the size of message sending to static agents.

5 Conclusions

After the employment of the proper pattern of interaction protocol based on mobility it is possible to ascertain the mobility solution on the base of carried performance tests of MASs. This solution can exceed stationary solution in respect of:

- higher performance - MAS based on IP2 interaction protocol with the mobility acts faster than the equivalent static solutions based on IP1 interaction protocol
- small capacity of software – the capacity of the mobile test configuration of MAS based on IP2 protocol is less than for the static solution based on IP1 protocol
- small network traffic - it is possible to employ the simplest interaction protocol at the configuration of the agent according to the concept of Thick /Thin agent, where the thin agent is mobile.

References

1 Babczyński T., Kruczkiewicz Z., Magott J.: *Performance Evaluation of Multiagent Information Retrieval System,* Foundations of Computing and Decision Sciences, Vol. 29, No. 1-2 (2004) 7-23.

2 Babczyński T., Kruczkiewicz Z., Magott J.: *Performance Evaluation of Multiagent Personalized Information System,* In: Proc. of the 7th Int. Conf. on Artificial Intelligence and Soft Computing 2004, Lecture Notes in Computer Science-LNAI, Springer-Verlag (2004) 810-815.

3 Buse D. P., Sun P., Wu Q. H., Fitch J.: *Mobile agents for industrial information management, monitoring and control,* In: Proc.of International Conference on Intelligent Agents, Web Technology and Internet Commerce-IAWTIC'2003.12-14, February 2003, Vienna-Austria.

4 Carter J., Ghorbani A. A., Marsh S.: *Just-in time information architectures in multiagent systems,* In: Proc.of the First Int. Joint Conference on Autonomous Agents and Multiagent Systems: part 2 2002, Bologna, Italy.

5 FIPA, `http://www.fipa.org/specs/`

6 Fugetta A., Picco G. P., Vigna G.: *Understanding Code Mobility,* IEEE Transaction On Software Engineering. Vol. 24, No.5, 1998.

7 JADE, `http://sharon.cselt.it/projects/jade/doc/`

8 Jha R., Iyer S.: *Performance evaluation of mobile agents for e-commerce applications,* In: Proc. of International Conference on High Performance Computing, Hyderabad, India, Dec 2001.

9 Kelash H. M., Faheem H. M., Amoon M.: A Multiagent System for Distributed Systems Management, Transactions on engineering, computing and technology V11 February 2006 ISSN 1305-5313.

10 Mascardi V., Merelli E.: *Agent-Oriented and Constraint Technologies for Distributed Transaction Management,* DISI – Universita di Genova via Dodecaneso, 35-16146 Genova (Italy) e_mail: *mascardi@disi.unige.it*, Istituto di Informatica - Universita di Ancona via Brecce Bianche - 601301, Ancona (Italy) e-mail: *merelli@inform.unian.it*

11 O'Malley S. A., Self A. L., Deloach S. A.: *Comparing performance of static versus mobile multiagent systems,* In: Proc. of National Aerospace and Electronics Conference (NAECON) Dayton OH, October 10-12 2000.

12 P. Serrano-Alvarado, Roncancio C., Adiba M.: *Mobile Transaction Supports for DBMS,*. An Overview, LSR-IMAGLaboratory BP72, 38402 StMartin d'Heres.

13 Vogler H., Kunkelmann Th., Moschgath Marie-Louise: *An Approach for Mobile Agent Security and Fault Tolerance using Distributed Transactions,* Hartmut Darmstadt University of Technology, Information Technology Transfer Office Alexanderstr. 10, D-64283 Darmstadt, Germany `{vogler, kunkel, malu}@ito.th-darmstadt.de`

2nd International Symposium
Advances in Artificial Intelligence and Applications (AAIA'07)

International Symposium on Advances in Artificial Intelligence and Application (AAIA) will be held within the framework of the *International Multiconference on Computer Science and Information Systems*, and will be co-located with the XXIII Fall Meeting of Polish Information Processing Society. Each year during AAIA we plan to celebrate one outstanding Polish scientist who contributed to the field of Artificial Intelligence. This year we will celebrate 60th birthday of Professor Janusz Kacprzyk.

The AAIA'07 will bring researchers, developers, practitioners, and users to present their latest research, results, and ideas in all areas of artificial intelligence. We hope that theory and successful applications presented at the AAIA'07 will be of interest to researchers and practitioners who want to know about both theoretical advances and latest applied developments in Artificial Intelligence. As such AAIA'07 will provide a forum for the exchange of ideas between theoreticians and practitioners to address the important issues.

Papers related to theories, methodologies, and applications in science and technology in this theme are especially solicited. Topics covering industrial issues/applications and academic research are included, but not limited to:

- Data mining
- Knowledge management
- Neural Networks
- Evolutionary Computation
- Machine learning
- Fuzzy modeling and control
- Architectures of intelligent systems
- Image processing and understanding (interpretation)
- Natural Language processing
- Applications in Bioinformatics

Special Tracks

Besides regular sessions this year we plan to organize Special Tracks:

Computational Biology :	chair Ken Revett
Heuristics in Continuous Global Optimization:	chair Josef Tvrdik
Computer Linguistics - Applications :	chair Krzysztof Jassem

All papers submitted to Special Tracks have been undergo the same review process as regular submissions.

International Programme Committee

- János Abonyi , Pannon University, Hungary
- Ajith Abraham, Yonsej University, South Korea
- Troels Andreasen, Roskilde University, Denmark
- Irma Becerra-Fernandez, Florida International University, USA
- Janez Brest, University of Maribor, Slovenia
- David Wolfe Come, Heriot-Watt University, UK
- Wlodzislaw Duch, NCU Poland and NTU, Singapore
- Bogdan Gabrys, Bournemouth University, UK
- Krzysztof Goczyla, Gdansk University of Technology, Poland
- Mirsad Hadzikadic, UNC at Charlotte, USA
- Jerzy W. Jaromczyk, University of Kentucky, USA
- Krzysztof Jassem, Adam Mickiewicz University, Poland
- Piotr Jedrzejowicz, Gdynia Maritime University, Poland
- Yaochu Jin, Honda Research Center, Germany
- Etienne E. Kerre, Ghent University, Belgium
- Grzegorz Kersten, Concordia University, Canada
- Waldemar Koczkodaj, Laurentian University, Canada
- Mieczyslaw Kokar, Northeastern University, USA
- Jozef Korbicz, University of Zielona Gora, Poland
- Jerzy Korczak, University of Louis Pasteur, France
- Witold Kosinski, PJWSTK, Poland
- Juliusz Lech Kulikowski, Polish Academy of Science, Poland
- Jacek Mandziuk, Warsaw University of Technology, Poland
- Victor W. Marek, University of Kentucky, USA
- Zbigniew Michalewicz, University of Adelaide, Australia
- Ryszard S. Michalski, George Mason University, USA
- Daniel Neagu, University of Bradford, UK
- Andrew P. Paplinski, Monash University, Australia
- Witold Pedrycz, University of Alberta, Canada
- James F. Peters, University of Manitoba, Canada
- Hartmut Pohlheim, DaimlerChrysler AG Research and Technology, Germany
- Zbigniew Ras, University of North Carolina, USA
- Ken Revett, University of Westminster, UK
- Danuta Rutkowska, Technical University of Czenstochowa, Poland
- Leszek Rutkowski, Technical University of Czenstochowa, Poland
- Khalid Saeed, Bialystok Technical University, Poland
- Nahid Shahmehri, Linkopingn University, Sweden
- Andrzej Skowron, Warsaw University, Poland
- Roman Słowiński, Poznań University of Technology, Poland
- Marek Andrzej Suchenek, California State University, USA
- Stan Szpakowicz, University of Ottava, Canada and PAS, Poland
- Ryszard Tadeusiewicz, AGH University of Science and Technology, Poland
- Miroslaw Truszczynski, University of Kentucky, USA
- Jozef Tvrdik, University of Ostrava, Czech Republik
- Athanasios Vasilakos, University of Western Macedonia,Greece

- Anita Wasilewska, Stony Brook University, U.S.A
- Wlodek Zadrozny, IBM, USA
- Daniela Zaharie, West University of Timisoara, Romania
- Ivan Zelinka, Tomas Bata University, Czech Republic
- Wojciech Ziarko, University of Regina, Canada
- Jacek M. Zurada, University of Louisville, USA

Organizing Committee

Halina Kwaśnicka and Urszula Markowska-Kaczmar, Institute of Applied Informatics, Wrocław University of Technology, Wrocław, Poland

Proceedings of the International Multiconference on
Computer Science and Information Technology, pp. 87 – 92
ISSN 1896-7094
© 2007 PIPS

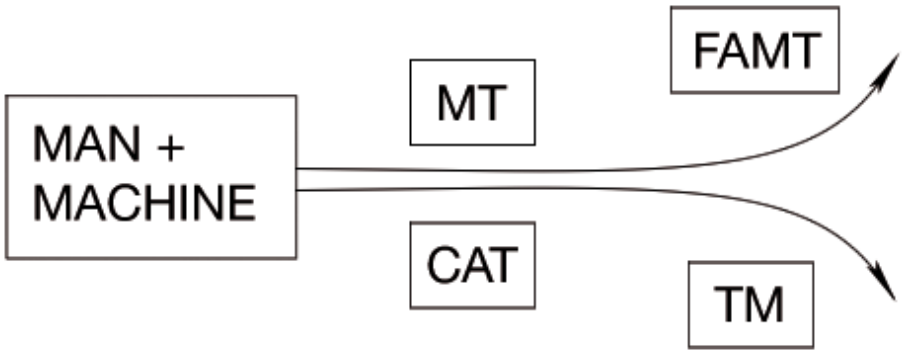
Using Regular Expressions in Translation Memories

Jacek Gintrowicz, Krzysztof Jassem

Faculty of Mathematics and Computer Science, Adam Mickiewicz University, Poznań
gintowicz@gmail.com, jassem@amu.edu.pl

Abstract. The paper presents the idea of using regular expressions in translation systems based on translation memories. Regular expressions are used in: search for a match in the instance sentence, search for an appropriate example in the translation memory and in the transfer of the instance sentence into its equivalent. The application of transfer rules to translation memories support the thesis, put forward in the paper, that Machine Translation and Computer-Aided Translation converge into the same direction.

"While the original aim of the MT pioneers was fully automatic MT systems, there has also been, at least since the 1966 ALPAC report and possibly even earlier, the view that computers could be used to help humans in their translation task rather than replace them". This citation comes from Harold Somers (Somers, 2003). The below figure may illustrate this view:

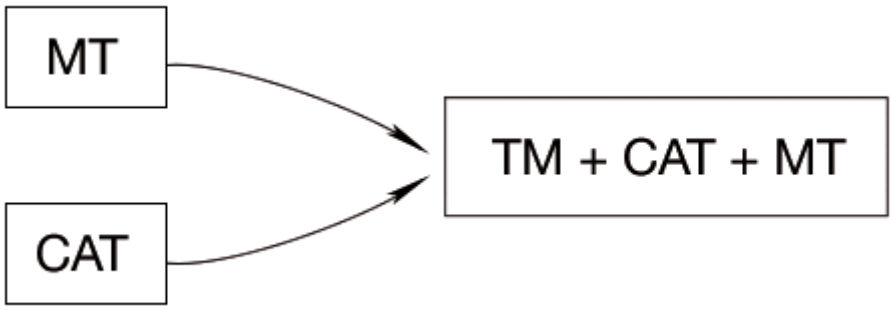


The idea of co-operation between man and machine goes in two directions: one direction is via Machine Translation (MT), possibly pre-edited or/and post-edited, towards Fully Automatic Machine Translation (FAMT) where there is nothing to do for a human. The other direction is via Computer-Aided Translation (CAT), where the user – human translator – is handily equipped with necessary glossaries and dictionaries and has the possibility to re-use his/her own translation, towards the use of large Translation Memories where there is almost nothing to do for a computer (except for just matching an instance text to an example that has once been translated).

It appears that the idea of Computer-Aided Translation is nowadays beating the idea of Machine Translation. The reason is mainly economical: The creation of a good-quality MT system is much more costly than that of a good-quality CAT system

and the demand for the two types is comparable. It does not mean that there is no future for MT. Firstly, modern approaches, based on large corpora, such as: Statistical Machine Translation (SMT) [Brown, 1990, Ney, 2005] and Example-Based Machine Translation (EBMT) [Nagao, 1984, Hutchins, 2005] aim at building MT systems at minimum cost. Secondly, once the quality of MT becomes reliable, the demand for it will rapidly grow.

Still, the near future belongs to CAT. But we believe in the following scenario:



The producers of MT systems will incorporate more and more ideas and tools from CAT systems. On the other hand, CAT systems will include more and more machine work (some CAT systems can already be linked to MT procedures) .Eventually, there will be no difference between CAT and MT systems. There will be no translator's workbench or machine translation system – just one translation tool.

This paper shows a step in this direction. Given a translation memory and a CAT system that uses the memory, can we enhance the process of matching a sentence under translation to a memory unit? The enhancement should be executed fully automatically.

The idea presented in the paper is to enhance the procedure of matching by means of regular expressions.

The idea of using regular expressions for natural language processing is widely known. Regular expressions are most frequently applied to searching for regularly structured fragments, like numbers or dates. Kartunnen (1996) suggests applying finite automata and transducers that represent regular expressions, for natural language texts. Oflazer (2004) shows the use of regexps for tokenization, shallow parsing or morphology analysis. Hasan (2005) describes how regexps may be applied in SMT for sentence clustering.

Here, we present the use of regular expression in a translation system based on the translation memory. Regular expressions are used in: search for a match in the instance sentence, search for an appropriate example in the translation memory and finally in the transfer of the instance sentence into its equivalent.

Below we present the formalism that describes search-and-transfer rules developed for a CAT system, called TranslAide.

Each rule consists of:

A) Regular expressions defining search patterns in: Instance, Source TM and Target TM.

B) Transfer of the instance expression into an output expression.

Below, we show an example of a rule for translation from Polish into English. The rule converts one of Polish date formats into its English equivalent.

Suppose the instance sentence is:

Instance: *Egzamin jest zaplanowany na 18.02.2008.*

Suppose the translation memory contains the following example and its translation:

TM Source: *Egzamin jest zaplanowany na 15.01.2007.*

TM Target: *The exam is planned on 15/01/2007.*

We expect the algorithm to generate the following output sentence based on TM Target:

Output: *The exam is planned on 18/02/2008.*

The appropriate rule looks like this (only selected tags are listed for clarity):

```
Rule                                                                    1:
1.    <instance>([0-9]{1,2})[\.]([0-9]{1,2})[\.]([0-9]{2,4})</instance>
2.    <source>([0-9]{1,2})[\.]([0-9]{1,2})[\.]([0-9]{2,4})</source>
3.    <target>([0-9]{1,2})[\/]([0-9]{1,2})[\/]([0-9]{2,4})</target>
4.    <orders>
5.          <order      sourceGroup="1"      suffix="/"      />
6.          <order      sourceGroup="2"      suffix="/"      />
7.          <order      sourceGroup="3"      suffix=""       />
8.    </orders>
```

Line 1 defines the search pattern for an input sentence (here, the pattern matches one of Polish date notations). The Instance must contain a matching string to become applicable to the rule.

Line 2 defines the search pattern for the source part of an example. (Here, the pattern is identical to that for the Instance). The source part of an example must contain a matching string to be considered as the source for translation

Line 3 defines the search pattern for the target part of an example. The target part of an example must contain a matching string to be considered as the target for translation

Lines 4. to 8. define the transfer of the regular expression: Line 5 refers to the first group of the instance surrounded by braces, i.e. *[0-9]{1,2}*. It says that in the output sentence the matching string should be added a suffix "/".

Lines 6 and 7 refer to the second and third group of the instance respectively.

The <orders> tag makes it possible to permutate the order of the groups. The groups are listed inside the tag in the order expected in the target sentence (here, the order of the target is identical to that of the source).

Algorithm

The consecutive steps of the algorithm will be illustrated by applying Rule 2 and Rule 3 to the below sentences:

```
Rule 2:
<instance>([0-9]{1,2})[\.]([0-9]{1,2})[\.]([0-9]{2,4})</instance>
<source>([0-9]{1,2})[\.]([0-9]{1,2})[\.]([0-9]{2,4})</source>
<target>[0-9{1,2})(st|nd|rd|th)?\s ((January)|(February)|(March)|(April)|(May)|
(June)|(July)|(August)|(September)|(October)|(November)|(December))\s([0-
9]{2,4})</target>
<orders>
        <order sourceGroup="1" sufix=" " />
        <order sourceGroup="2" sufix=" " />
        <order sourceGroup="3" sufix="" />
</orders>
<rules className="ToEnglishDateRules" />
```

Rule 2 is another rule for date conversion.

```
Rule                                                                    3:
<instance>([0-2]?[0-9])(?:\:([0-5]?[0-9])(?:[\:\.]([0-5]?[0-9]))?)?</instance>
<source>([0-2]?[0-9])(\:([0-5]?[0-9])([\:\.]([0-5]?[0-9]))?)?</source>
<target>((([0-1]?[0-9])([\:\.]([0-5]?[0-9])([\:\.]([0-5]?[0-
9]))?)?\s?[AaPp]\.?[Mm]\.?)|(([0-2]?[0-9])(\:([0-5]?[0-9])([\:\.]([0-5]?[0-
9]))?)))</target>
<orders>
        <order          sourceGroup="1"          sufix=":"          />
        <order          sourceGroup="2"          sufix=":"          />
        <order          sourceGroup="3"          sufix=""           />
</orders>
<rules className="ToEnglishTimeRules" />
```

Rule 3 searches for a Polish time format and converts it into an English time format while translating.

Instance: Egzamin jest zaplanowany na 12.05.2008, na godzinę 15:15.
TM Source : Egzamin jest zaplanowany na 02.03.2007, na godzinę 13:00.
TM Target: The exam is planned on 2 nd March 2007, at 1 PM.
Output: The exam is planned on 12 May 2008, at 3:15 p.m.

*0**Input:** Instance.
*1

For each rule in turn
if regexp described in <instance> tag matches a fragment of Instance
extract the fragment, and name it **RegExpInstance**
//Here, Rule 1 extracts the RegExpInstance: 12.05.2008

For each TM Unit in turn //TM Unit is a pair <TMSource; TM Target>
If regexp described in <target> matches a fragment of TM Target
extract the fragment, and name it **RegExpTarget**
*//Here, the above TM Target is found and **2 nd March 2007** is extracted*

Separate RegExpInstance into groups according to positions of braces in the <instance> pattern
*//Here the **12.05.2008** is divided into 3 groups: **12** , **05** , **2008** .*

For each group use the conversion declared in <rules className>
*//Here, the rule named ToEnglishDateRules is invoked. The rule converts the 3 groups into **12, May, 2008** respectively*

Merge the converted groups according to the <orders> element.
//Here, the merged expression is: 12 May 2008, as the order of the groups is preserved

Place the merged expression into the TM Target surrounded by the tag **antra-changed**

//The following is obtained:
The exam is planned on <antra-changed> 12 May 2008 </antra-changed> , at 1 PM.
Still, the hour needs adjustment

Go back to the beginning. In the next loop do not touch texts surrounded by the **antra-changed** tag.
//Another loop yields:
*The exam is planned on **<antra-changed> 12 May 2008 </antra-changed> ,** at **<antra-changed>** 3:15 p.m **</antra-changed> .***

Clear up all <antra-changed> tags
So far, transfer rules have been developed for:

- various formats of date
- various formats of time
- currency expressions
- metric expressions
- numbers
- e-mail addresses.

Future Work

One direction of the development is to expand the set of rules. The sample ideas for the Polish language are: zip-codes, telephone numbers and URL addresses.

Another aspect is the transfer of formatting tags. The most obvious fields of interest seem to be XML and HTML formatting languages. The structure of tags is strictly defined, which allows for using the mechanism of regular expressions to recognize and transfer tags between texts in different languages.

Another possible direction is to expand the formalism beyond regular expressions. The idea is to replace regular expressions in the rules by lexical expressions based on bilingual dictionaries prepared beforehand. The authors have the access to multiple bilingual dictionaries developed for the MT system, Translatica (`www.translatica.pl`, `www.poleng.pl`). The first experiments with this idea look promising.

Summary

The transfer approach in Machine Translation has been widely criticized for high cost of rule preparation. The idea of using transfer rules in a CAT system, presented here, overcomes this problem: On the one hand a rule is not required to cover the whole sentence – only a part must match the pattern – which expands the application of a rule. On the other, thanks to using a general formalism of regular expressions, one rule may be matched to many fragments of texts.

A successful application of an MT idea (transfer rules) to a CAT system supports the thesis of the convergence between MT and CAT systems.

Bibliography

1. Brown P. F., Cocke J., Della Pietra S. A., Della Pietra V. J., Jelinek F., Lafferty J. D., Mercer R. L., Roossin P. S. (1990), *A statistical approach to machine translation* . Computational Linguistics 16 (2), pp. 79-85.
2. Hasan S., Ney H.(2005), *Clustered language models based on regular expressions for SMT*, 10th EAMT conference *Practical applications of machine translation*, 30-31 May 2005, Budapest; pp. 119-125.
3. Hodász G., Gröbler T., Kis B. (2004), *Translation memory as a robust example-based translation system* , 9th EAMT Workshop, Broadening horizons of machine translation and its applications, 26-27 April 2004, Malta; pp. 82-89.
4. Hutchins J. (2005), *Towards a definition of example-based machine translation* , MT Summit X, Phuket, Thailand, September 16, 2005, Proceedings of Second Workshop on Example-Based Machine Translation; pp.63-70.
5. Karttunen L., Chanod J-P., Grefenstette G., Schiller A.(1996), *Regular Expressions for Language Engineering*, Natural Language Engineering, Cambridge University Press, pp. 305-328.
6. Nagao M. (1984), *A framework of a mechanical translation between Japanese and English by analogy principle* , Artificial and human intelligence: edited review papers presented at the international NATO Symposium, October 1981, Lyons, France; ed. A. Elithorn and R. Banerji. Amsterdam: North Holland, pp. 173-180.
7. Ney H. (2005), *One decade of statistical machine translation: 1996-2005*, MT Summit X, Phuket, Thailand, September 13-15, 2005, Conference Proceedings: the tenth Machine Translation Summit: invited paper; pp. 12-17.
8. Oflazer K., Yilmaz Y, *Vi-xfst* (2004), *A Visual Regular Expression Development Environment for Xerox Finite State Tool*, In Proceedings of the Seventh Meeting of the ACL Special Interest Group in Computational Phonology, Association for Computational Linguistics, Barcelona, Spain, July 2004, pp 86-93.
9. Somers H. (ed.) (2003), *Computer and translation: a translator's guide* , John Benjamins Publishing, Amsterdam, page 11.

Proceedings of the International Multiconference on
Computer Science and Information Technology pp. 93–102
ISSN 1896-7094
© 2007 PIPS

A Rule-Based Characterization of Clusters of Genes

Aleksandra Gruca

Silesian University of Technology,
ul. Akademicka 16, 44-100 Gliwice, Poland
Aleksandra.Gruca@polsl.pl

Abstract. This paper presents a method of using the attribute analysis and the rough sets theory for describing and analyzing the Gene Ontology (GO) composition of clusters of genes obtained in DNA microarray experiments. GO terms are understood as attributes of genes and gene clusters are characterized by decision rules related to attributes. A modification of the known algorithm for computing the decision rules for the information system is proposed, which makes it suitable for large-size problems encountered in the analysis of DNA microarray data. Additionally methods are developed for the assessment of the statistical significance of the computed rules. Presented approach is used for DNA microarray data obtained in experiments of measuring transcriptome response of cells to ionizing radiation. The results of computations are compared with those obtained with the use of GO browsers.

1 Introduction

The computational techniques that accompany the development in DNA microarray experiments [1, 2] are most often focused on tasks of (i) identifying genes or groups of genes expressed differentially between different experimental conditions, or (ii) identifying groups (clusters) of genes coexpressed in sequences of experiments [3, 4]. Information on differential expression or coexpression of genes is helpful in predicting outcomes of further experiments or in classifying biological samples on the basis of their gene expression profiles.

Another element of the interpretation of DNA microarray data includes confronting information on levels of gene expressions with the existing biological knowledge on genes, their classes and functions. Including that knowledge to analysis leads to robustifying the conclusions of the study against statistical artefacts, and also supports extracting biological knowledge from performed experiments.

The most widely used source of data on functions of genes is the Gene Ontology (GO) database [5] which provides standardized and structured vocabulary that describes genes and their products. The GO database is represented as three disjoint directed acyclic graphs describing biological process, molecular function and cellular component. Each has a single root and a thousands of depended nodes. Upper nodes represent more general concepts and as the DAG is

traversed towards deeper levels, the definitions are more and more precise. GO browsers, such as [6] allow one to characterize the clusters of genes obtained in some experiment by computing and comparing frequencies of GO terms. Many other studies [7–12] develop various systematic methods to use GO terms in conjunction with various information processing methods to obtain biological interpretation of experimental results.

Method proposed in this paper allows characterizing the composition of clusters of genes on the basis of the decision rules expressed as logical functions of the GO terms. The algorithm for computing logical decision rules on the basis of the rough sets theory [13] is elaborated. A modification of the published method [14] for computing the decision rules in the information system is introduced that makes it possible to use the algorithm for large-size problems. There are also methods developed for the assessment of the statistical significance of the computed rules. Presented approach is used for DNA microarray data obtained in the experiment of measuring transcriptome response of cells to ionizing radiation. The results of computations are compared to those obtained by using of one the popular GO browser.

2 The proposed approach

Let there be defined the set U which is a set of all genes whose probes are placed on the DNA microarray chips used in the analyzed experiment,

$$U = \{x_1, x_2, \ldots, x_N\}. \tag{1}$$

In the above $x_1, \ldots, x_N$ denote genes, distinguished by numbers or labels, and N is the number of genes. GO terms of genes, such as "regulation of transcription", "ribosome" or "tRNA processing" are interpreted as the binary attributes, in other words functions which assign to each of the genes, the value 0 or 1,

$$a\colon x \to a(x) \in \{0, 1\}. \tag{2}$$

In the above x denotes one of the genes and a denotes one of the attributes. So, for example if a corresponds to "tRNA processing", then $a(x) = 1$ holds for all genes x which contain "tRNA processing" among their GO terms. For all other genes $a(x) = 0$.

Considering a cluster of genes, $G \subset U$, $G = x_1, x_2, \ldots, x_P$, the problem is whether and how G can be described by the attributes of the genes $x_1, x_2, \ldots, x_P$?

GO browsers, for example [6], typically describe the gene composition of the cluster G or compare the gene composition of two clusters of genes G_1 and G_2 by computing frequencies of GO terms (attributes). Using terminology proposed in this paper GO browsers characterize G by using single-attribute rules of the following form:

$$\text{IF } a(x) = 1 \text{ THEN } x \in G, \tag{3}$$

which are either true or false. By frequency of GO term corresponding to a in G one can understand how often (3) is true for genes $x \in G$.

This paper proposes a method that allows characterizing clusters of genes by rules of the form more complex than (3), such as the one below:

$$\text{IF } a_1(x) = v_1 \text{ and } a_2(x) = v_2 \ldots \text{ and } a_R(x) = v_R \text{ THEN } x \in G. \tag{4}$$

In the above $v_1, \ldots, v_R$ are values from the set $\{0, 1\}$.

3 The algorithm for computing decision rules

In this section some necessary terminology from the rough sets theory is introduced that is essential to understand the idea of the presented method. An information system S is a pair $S = (U, A)$, where U is called a universe and A is a set of attributes. Objects from U represent the investigated cases (genes in the experiment) and attributes are features describing these objects. The attribute $a \in A$ is a map $a : U \to V_a$, where V_a is the value set of the attribute a.

Decision table $DT = (U, A \cup \{d\})$ is another system of objects including a universe U, a set of attributes A called conditional attributes and a distinguished attribute $d \notin A$, called the decision attribute. The decision attribute determines the partition $\{X_{d1}, \ldots, X_{dk}\}$ of the U with respect to the value of the decision attribute d_i. With any subset $B \subseteq A$ one can associate the equivalence relation called B-indiscernibility relation, defined as:

$$IND(B) = \{x, y \in U \times U\colon \ \forall_{a \in B} \quad a(x) = a(y)\}. \tag{5}$$

The objects from U satisfying the relation $IND(B)$ are indiscernible form each other with respect to attributes from B.

A decision rule is a logical expression of the form:

$$\text{IF } a_1 \in V_{a1} \text{ and } a_2 \in V_{a2} \text{ and } \ldots \text{ and } a_n \in V_{an} \text{ THEN } d = v. \tag{6}$$

where $v \in D_d$, $\{a_1, a_2, \ldots a_n\} \subseteq A$ and $V_{ai} \subseteq D_{ai}$, $i = 1, 2, \ldots, n$. The interpretation of the decision rule is intuitive – values of the attributes on the left-hand side of the rule should imply the value of the decision attribute.

Given object recognizes the rule if its attributes values satisfy the premise of the rule. The object supports the rule if it recognizes the rule and the decision given to the object is the same as a decision from the right side of the rule.

3.1 Feature reduction

Usually among all attributes that describe objects there are attributes irrelevant or redundant with respect to knowledge represented by the whole data set. The process of removing these attributes is called feature selection or reduction. Apart from reduction of the time of further computations and reduction of the amount of achieved data the most important benefit of feature selection is that obtained knowledge is both more structured and more compact and therefore easier to understand. In the rough sets theory, the process of feature selection

is considered as a computation of the subset of attributes called the reduct. For a given information system $S = (U, A)$ and its indiscernibility relation $IND(A)$ with respect to the whole attribute set A, the reduct is a minimal subset $B \subseteq A$ such that $IND(A) = IND(B)$. In other words, the reduct is a minimal subset of attributes that preserves the same abilities of discerning objects as the whole attribute set A. More than one reduct may exist for an information system.

Concerning the decision table $DT = (U, A \cup \{d\})$ there is a notion of the relative reduct, that is a minimal subset $B \subseteq A$ such that $IND(B) \subseteq IND(d)$. The set of all relative reducts in A is denoted by $RED_{DT}(A, d)$. The problem of finding a minimal reduct has been proven to be NP-hard [15], hence, the heuristic algorithm is always employed to find an approximate reduct.

In [14] Nguyen and Nguyen proposed an algorithm for computing an approximate reduct of the information system, based on the equivalence relation, suitable for large data-size problems. Let $S = (U, A)$ be an information system and $X \subseteq U$. For any attribute $a \in A$ there is an indiscernibility relation $IND^X(a)$ over objects form X determined by an attribute a. This relation determines the partition of the set X into equivalence classes denoted by $[INDX(a)]$. Assuming that there is a partition of the X such that $[IND^X(a)] = \{X_1, \ldots, X_m\}$ and cardinalities of the sets $X, X_i, \ldots X_m$ are respectively $x, x_1, \ldots, x_m$. The number of pairs of objects from X discerned by an attribute a may be computed by the following formula:

$$W^X(a) = \frac{\sum_{i \neq j} x_i x_j}{2} = \frac{x^2 - \sum_{i=1}^{m} x_i^2}{2}. \tag{7}$$

The number of pairs of objects discerned by any attribute can be computed using the following pseudocode [14]:

Require: an information system $S = (U, A)$
Ensure: a minimal reduct R, $L = [IND^U(R)]$
  $R = \emptyset$, $L = \{U\}$
  repeat
    for all $a \in A$ **do**
      for all $X_i \in L$ **do**
        search for $[IND^{Xi}(a)]$
        count $W^{Xi}(a)$
      end for
      $W^U(a) = W^{X1}(a) + \ldots + W^{Xm}(a)$
    end for
    choose attribute a with max value of $W^U(a)$
    $A = A \setminus \{a\}, \quad R = R \cup \{a\}$
    $L = [IND^{X1}(a)] \cup \ldots \cup \ [IND^{Xm}(a)]$
  until $A = \{\emptyset\}$ or $W^U(a) = 0$

For the decision table $DT = (U, A \cup \{d\})$, the important information from the feature selection perspective is the information about attributes that discern

objects only from a different decision classes. In other words there is no need to discern objects with the same value of the decision attribute. Hence, this paper introduces modification to the original algorithm that allows computing the reduct relative to the given decision table. The algorithm searches for the attribute that discerns the maximal number of pairs of objects, but only from the different decision classes.

Assuming there is a subset $X \subseteq U$, two decision classes that split the X into two subsets X_{di} and X_{dj} and numbers of objects belonging to these classes are respectively x_{di} and x_{dj}, all pairs of objects discerned by these decision classes can be simply computed by multiplication of x_{di} by x_{dj}. Thus the attribute discerning the most objects from different decision classes can be computed as follows:

$$W^X(a) = \frac{\sum_{i \neq j} x_i x_j}{2} - \frac{\sum_{i \neq j, d(x_i) \neq d(x_j)} x_i x_j}{2}. \tag{8}$$

Assuming that $card(U) = n$ and $card(A) = k$ the time complexity of the whole algorithm is $O(k^2 n \log n)$. The space complexity of the algorithm is $O(n)$.

3.2 Sequential covering algorithm for rules generation based on the relative reduct

After the relative reduct is computed, the next step of the algorithm involves generating decision rules with the use of the sequential covering method. The method is based on a simple idea: learn one rule for a given object, search for the objects recognized by that rule are remove them from the processed data. The decision rules are induced using the properties of minimal relative reduct such that for any $R \in RED_{DT}(A, d)$ the set of reduct attributes determines a decision value d.

Let $DT = (U, A \cup \{d\})$ be a decision table. The coverage of the decision system denoted as $[RUL(DT)]$ is a set of decision rules that for each object $u \in U$ there is at least one rule supporting that object. Computing the set of decision rules can be described by the following pseudocode:

Require: $DT = (U, A\{d\}), \quad R \in RED_{DT}(A, d)$
Ensure: $RUL(DT)$ - set of decision rules for S
  $RUL(S) = \emptyset$
  while $U \neq \emptyset$ **do**
    $r = \forall_{a_i \in R} \quad a_i = a_i(u) \rightarrow d = d(u)$ {create a decision rule r}
    if $r \notin RUL(DT)$ **then**
      $RUL(DT) = RUL(DT) \cup \{r\}$
      $U = U \setminus [RUL(DT)]$
    end if
  end while

4 Statistical significance of the decision rules

Having the clustering of genes obtained in DNA microarray experiment, and the decision rules, the next task is to assess the statistical significance of the rules. Decision rules are statistically significant if the null hypothesis of purely random composition of clusters of genes can be rejected. Statistical significance of decision rules confirms a non-random composition of gene clusters and supports drawing further biological conclusions.

A common method to verify the statistical significance of the decision rule D, given the gene cluster G_i involves comparison of the attributes (decision rules) in gene sets G_i and $U \setminus G_i$. For every rule D there is a contingency table:

Table 1. Contingency table describing the application of the decision rule D to partition of the universe U into clusters G_i and $U \setminus G_i$

Decision rule D	G_i	$U \setminus G_i$
True	N_{GT}	N_{UT}
False	N_{GF}	N_{UF}

where N_{GT} and N_{UT} denote numbers of genes, respectively in G_i and $U \setminus G_i$, $N_{GT}+N_{UT}$ is the number of genes recognizing the rule D and N_{GT} is the number of genes supporting the rule D. N_{GF} and N_{UF} denote numbers of genes, for which the rule is false.

Treating genes in G_i and $U \setminus G_i$ as having two different "colors" the null hypothesis is stated as "the decision rule D is color blind". Under the null hypothesis the probability of obtaining the configuration N_{GT}, N_{UT}, N_{GF} and N_{UF} follows the hypergeometric distribution:

$$p(N_{GT}, N_{UT}, N_{GF}, N_{UF}) = \frac{\binom{N_{GT}+N_{GF}}{N_{GT}}\binom{N_{UT}+N_{UF}}{N_{UT}}}{\binom{N_{GT}+N_{UT}+N_{GF}+N_{UF}}{N_{GT}+N_{UT}}}. \tag{9}$$

which, in the case of testing for the overrepresentation of N_{GT}, leads to the following p-value of the Fisher exact test [16]:

$$p_{FET}(N_{GT}, N_{UT}, N_{GF}, N_{UF}) = \sum_{k=1}^{N_{UT}} p(N_{GT}+k, N_{UT}-k, N_{GF}, N_{UF}). \tag{10}$$

Fisher exact tests like the above, or their approximations, chi-square tests for independence, are typically applied in the GO browsers. Since tests are performed for large number of rules, corrections for multiple testing are applied [3, 4].

The construction of the contingency table, in Table 1, is the same for the single attribute rules (3) and for the multi-attribute rules (4). However, there is a quantitative difference between single attribute rules (3) and multi-attribute

rules (4). For multi-attribute rules the number of genes recognizing the rule is typically very small. The extreme case $N_{UT} = 0$ and $N_{GT} = 1$ is often encountered in the data. In such case the calculated p-value of the test in (10) can be very low, yet there is nothing improbable in the random choice of a single gene.

In order to obtain a statistical model for the small number of genes recognizing the computed rules, more realistic than (9)–(10), there is considered conditional probability of obtaining the configuration N_{GT}, N_{UT}, N_{GF} and N_{UF} given that the decision rule D has already classified one gene as belonging to G_i, given by the expression:

$$p^c(N_{GT}, N_{UT}, N_{GF}, N_{UF}) = p(N_{GT} - 1, N_{UT}, N_{GF}, N_{UF}), \tag{11}$$

where $p(.)$ is given by (10) and the p-value of the corresponding variant of the Fisher exact test is:

$$p^c_{FET}(N_{GT}, N_{UT}, N_{GF}, N_{UF}) = \sum_{k=1}^{N_{UT}} p^c(N_{GT} + k, N_{UT} - k, N_{GF}, N_{UF}). \tag{12}$$

When (11)–(12) is used to the case $N_{UT} = 0$ and $N_{GT} = 1$, $p^c_{FET} = 1$ is obtained which is in accordance with intuition.

5 Application to the DNA microarray data set

The elaborated methodology was applied to the data set obtained in the DNA microarray experiment related to observing the transcriptome dynamic response of the human melanoma cell line ME45 to ionizing radiation [17]. Gene expression profiles of the cell line were measured with the use of the Affymetrix chip U133A [18] containing approximately 22000 spots corresponding to known human genes. Based on the expression profiles of the cell line measured at several time instants genes on the U133A Affy chip were divided into 48 gene clusters [19].

5.1 Computing the decision rules

The annotations of Affymetrix DNA microarray probe sets with GO terms are provided and updated on regular basis [20] and are available on the NetAffx web site: `http://www.affymetrix.com`. For the computations the annotation file from March, 2007 was used. GO terms belonging to three ontologies: biological process, molecular function and cellular component were used to describe each gene. Some of the genes had no GO term assigned so they were removed from further investigations. Having removed these genes the decision table was generated including 12050 objects (genes) described by the 5171 attributes (GO terms). For each gene its attributes values were set in the following way: if the gene had a GO term assigned the attribute value corresponding to that GO term was set to 1 otherwise the value was set to 0. Then the minimal relative reduct was computed that decreased the number of attributes to 627.

Using the sequential covering algorithm based on relative reduct 11522 decision rules were obtained. For each rule the statistical significance was calculated and only 46 rules with p-value lower than 0.05 were considered as rules with good descriptive features that could be used for further description of gene clusters.

5.2 Comparison of the statistical significance of decision rules obtained by using two different methods

The statistical significance for 48 gene clusters $G_1, G_2, \ldots, G_{48}$ of the partitions G_i, $U \setminus G_i$ assessed by using the GO internet tool Fatogo+ [6] was compared with the statistical significance of multi-attribute decision rules presented in the previous subsection. For the GO term the GO level defines the distance from that term to the most upper node of a DAG. For three GO levels: 3, 5, and 7 the lowest (most significant) value of all statistical tests performed was recorded. Statistical tests performed by the applied internet tool were based on expressions (9)–(10). Multiple testing corrections were not taken into account. For multi-attribute rules (4), expressions (11)–(12) were used for computing p-values of the statistical test and, as previously, for each of the partitions G_i, $U \setminus G_i$ the lowest p-value was recorded. Figure 1 presents comparison of the statistical significances of the decision rules analyzed by the Fatigo+ internet tool and the decision rules obtained by presented method.

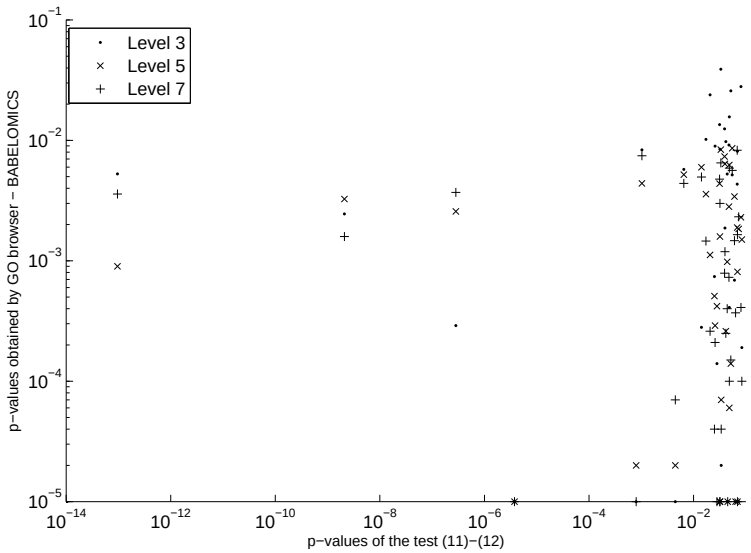


Fig. 1. Comparison of statistical significances of decision rules analyzed by the Fatigo+ internet tool on 3^{rd}, 5^{th}, 7^{th} GO level and decision rules obtained by presented method.

6 Conclusions

The analysis of the contents of gene clusters performed by GO browsers [6] have been interpreted by defining a single-attribute rule (3) and assessing its statistical significance by (9)–(10). A characterization of clusters of genes by multi-attribute rules of the form (4) have been considered. An example of a multi-attribute rule computed on the basis of the analyzed data set [17] is:

```
a1304 = ‘‘binding’’ AND
a1312 = ‘‘iron ion binding’’ AND
a1316 = ‘‘protein binding’’ AND                                    (14)
a1950 = ‘‘transport’’ AND
a3485 = ‘‘oxygen binding’’ AND
a4710 = ‘‘metal ion binding’’ => class is 23 ,
```

recognized by 4 and supported by 3 genes.

Multi-attribute rules, such as (14), are typically recognized by only small number of genes. In order to assess their statistical significance a variant of the Fisher exact test based on probability (11) was employed conditioned on the event of one gene already supporting the rule.

Comparing statistical significances of single- and multi-attribute rules for the same gene clustering, leads to some conclusions. For one group of clusters, roughly half of the data, p-values of both tests are well correlated. For another group there is no correlation. This may suggest different mechanisms behind forming of the detected gene clusters.

Statistical significance of some of the obtained multi-attribute rules suggests the existence of the biological meaning behind them. Information properties of multi-attribute rules, small number of recognizing genes and many GO terms in one rule, provide a new tool for the characterization of gene clusters by their GO terms, which can be useful for biologists.

Acknowledgement. This paper was partially supported by the European FP6 grant, GENEPI, lowRT, Genetic Pathways for the Prediction of the Effects of Ionising Radiation: Low dose radiosensitivity and risk to normal tissue after Radiotherapy.

References

1. Baldi P., Hatfield G. W.: *DNA Microarrays and Gene Expression,* Cambridge University Press, Cambridge (2002)
2. Speed T. (ed.): *Statistical Analysis of Gene Expression Microarray Data. Interdisciplinary Statistics,* Chapman & Hall\CRC, Boca Ration (2003)
3. Allison D. B., Gadbury G., Heo M., Fernandez J., Lee C. K., Prolla T. A., Weindruch R.: *A Mixture Model Approach for the Analysis of Microarray Gene Expression Data,* Comput. Statist. Data Anal. 39, 1–20 (2000).
4. Storey J. D., Tibshirani R.: *Statistical Significance for Genomewide Studies,* Proc. Natl. Acad. Sci. USA. 100, 9440–9445 (2003).

5. Ashburner M., Ball C. A., Blake J. A., Botstein D., Butler H., Cherry J. M., et al.: *Gene Ontology: Tool for the Unification of Biology,* The Gene Ontology Consortium. Nature genetics. 25, 25–29 (2000).
6. Al-Shahrour F., Minguez P., Vaquerizas J. M., Conde L., Dopazo J.: *BABELOMICS: A Suite of Web Tools for Functional Annotation and Analysis of Groups of Genes in High-Throughput Experiments,* Nucleic Acid Research. 33, W460–W464 (2005).
7. Subramanian A., Tamayo P., Mootha V. K., Mukherjee S., Ebert B. L., Gillette M. A., et al.: *Gene Set Enrichment Analysis: A Knowledge-Based Approach for Interpreting Genome-Wide Expression Profiles,* Proc. Natl. Acad. Sci. USA. 102, 15545–15550 (2005).
8. Lee I. Y., Ho J. M., Chen M. S.: *vCLUGO: A Clustering Algorithm for Automated Functional Annotations Based on Gene Ontology,* In: Proc. of the Fifth IEEE Int. Conf. on Data Mining, pp 705–708. IEEE Computer Society, Washington, DC (2005).
9. Lee S. G., Hur J. U., Kim Y. S.: *A Graph-Theoretic Modeling on GO Space for Biological Interpretation of Gene Clusters. Bioinformatics,* 20, 381–388 (2004).
10. Midelfart H., Komorowski H. J.: *A Rough Set Framework for Learning in a Directed Acyclic Graph,* Rough Sets and Current Trends In Computing. 144–155 (2002).
11. Wang H., Azuaje F., Bodenreider O., Dopazo J.: *Gene Expression Correlation and Gene Ontology-Based Similarity: an Assessment of Quantitative Relationships,* In: Proc. of the 2004 IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology. 25–31 (2004).
12. Svensson J. P., Stalpers L. J., Esveldt-van Lange R. E., Franken N. A., Haveman J., et al.: *Analysis of Gene Expression Using Gene Sets Discriminates Cancer Patients with and without Late Radiation Toxicity,* PLoS Med. 3, e422 (2006).
13. Pawlak Z.: *Rough Sets: Theoretical aspects of reasoning about data,* Kluwer Academic Publisher, Dordrecht (1991).
14. Nguyen S. H., Nguyen H. S.: *Some Efficient Algorithms for Rough Set Methods,* In: Proc. of the Conference of Information Processing and Management of Uncertainty in Knowledge-Based Systems, pp. 1451–1456, Granada (1996).
15. Skowron A., Rauszer C.: *The Discernibility Matrices and Functions in Information Systems,* In: Slowinski R. (ed.): Intelligent Decision Support: Handbook of Applications and Advances to Rough Sets Theory, pp. 331–362. Kluwer Academic Publisher, Dordrecht (1992).
16. Rice J. A.: *Mathematical Statistics and Data Analysis,* 2nd edn. Duxbury Press, Belmont (1995).
17. Konopacka M., Rogolinski J., Herok R., Polanska J., Fujarewicz K., Hancock R., Rzeszowska-Wolny J.: *Radiation Induced Genetic Changes in Directly Exposed and Neighboring K562 Cells in Vitro,* Manuscript submitted for publication.
18. *Affymetrix, GeneChip Expression Analysis, Data Analysis Fundamentals,* Affymetrix Manual (2002).
19. Polanska J., Widlak P., Rzeszowska-Wolny J., Kimmel M., Polanski A.: *Gaussian Mixture Decomposition of Time-Course DNA Microarray Data,* In Deutsch A., Brusch L., Byrne H., de Vries G., Herzel H. (eds.): Cellular Biophysics, Development, Biomedicine, and Data Analysis, series: Modeling and Simulation in Science, Engineering and Technology, pp. 351–359. Birkhäuser, Boston (2007).
20. Liu G., Loraine A. E., Shigeta R., Cline M., Cheng J., Valmeekam V., Sun S., Kulp D., Siani-Rose M. A.: *NetAffx: Affymetrix Probesets and Annotations,* Nucleic Acids Res. 31, pp. 82–86 (2003).

Proceedings of the International Multiconference on
Computer Science and Information Technology pp. 103–120
ISSN 1896-7094
© 2007 PIPS

Open-ended Evolution in Flocking Behaviour Simulation

Halina Kwasnicka, Urszula Markowska-Kaczmar and Marcin Mikosik

Wroclaw University of Technology, Poland
halina.kwasnicka, urszula.markowska-kaczmar@pwr.wroc.pl

Abstract. In this research we tried to apply open-ended evolution to breed controllers for artificial organism which would be able to manifest flocking behaviours. In comparison to the previous work in this area the artificial world created in our system is characterized by significant diversity of organisms. Animals were equipped with double and dynamic sight, which were suggested by other authors in their works. In experiments, many different behaviours were observed, which were very similar to those in nature, for instance: the escape of herbivore from predators, making herbivores route towards plants or a pursuit of predators after herbivores. Another interesting behaviour was grouping of predators around plants, where the probability of meeting herbivore is greater than in other places. The most advanced behaviour was creation of flocks, which was the goal of experiments. The observed motion of animals looked natural. However, to the full success it was necessary to apply steered evolution.

1 Introduction

It has been ages since people have been admired flocking behaviour of animals. They have been fascinated by a veering formation of birds or a shoal motion. With the development of computer techniques they try to simulate them in computer animations but it is not an easy task for graphic artists. A smooth motion needs many subsequent frames, drawing of which is very arduous and laborious work. Additionally, each correction of route of an individual needs to take into consideration many other individuals which makes the problem more difficult. That is why this problem has been studied by many researchers. Traditional approaches are based on simulations of mathematical equations of motion ([5, 2, 9, 14]). Their construction usually requires advanced mathematical skills and exact knowledge about the studied animal's behaviour.

One of the first advanced other solution in this area was proposed by [10]. It is the main way of solving a simple flock (as well as herds and schools) simulation so far. In this approach each individual is implemented as independent element called boid. Its behaviour is controlled by rules. They determine the behaviour of a given individual on the basis of its environment (position, direction of motion of other member of flock and information about obstacles). Local interactions cause the flocking behaviour on the global level (of a whole group). In order

to create virtual model of moving flock with using cited above method it is necessary to formulate and implement the set of rules defining single individual behaviour. The approach described by Reynolds in his works ([10] and [12]) was inspiration for many other researchers. For instance, [1] present rules based on fuzzy logic. Another example based on crisp rules is given by [13]. This work enhances the basic Reynolds' system in several ways. First of all it uses three distinct species of agents. Rules are also the basis of the system described by [8]. In this system at each time step, the positions and directions of individuals are updated by a set of simple rules. Individuals perceive the environment around them and can sense the distance between them and other individuals.

Often tuning of these rules is very time consuming. The solution for this problem carries out an application of evolutionary algorithms. As an example of such approach the paper of [16] may be cited here. In this work the authors try to breed population of individuals manifesting flocking behaviour. Artificial fishes move in three-dimensional word. This motion is controlled by a neural network, which is evolved during evolution. Evaluation of a single individual is consisted in creation of clones of this individual (having the same genotype). Then, this group is placed in the environment, its behaviour is monitored and evaluated with a fitness function. In the experiments some elements of flocking behaviour have been noticed, which confirmed that it is the proper direction of studies, but in general,the authors evaluated that the results were little realistic and behaviour of artificial animals does not remind of the one, which exists in reality. The reason that design failed, besides the problem with a proper design of fitness function, the authors have pointed to poor environment in created artificial world. Flocking behaviours in nature are the result of a diversity of external factors as, for instance, a temperature gradient, a light gradient or arrival of other species. Authors – [3] applied genetic programming to breed flocking behaviors. The fitness function is given by the user in an interactive way.

The achievements described in the paper of [15] can be perceived as one step ahead in comparison with the results presented in the work of [16]. These authors also applied evolutionary approach but instead of explicitly defined fitness function they decided to use open–ended evolution, giving the organisms more freedom in their development. Additionally, the environment has been enriched in other species. There were plants, which were food for the first species – herbivores that are the food for predators. The assumption of this system was that flocking behaviours should arrive as a result of conditions existing in artificial world. However in this project the problem with definition of flocking behaviour concept has arrived. It was not necessary to define fitness function because in open-ended evolution the aim of each organism is surviving, that is expressed as possessing of a sufficient level of energy. Here the statistical approach was applied. The common feature of both papers ([16] and [15]) is the way of perceiving surrounding world by individuals. This resulted in the type of information, which was provided as inputs for controllers of individuals. In both cases this information is static. Such organism is informed about location of other objects, but it is not conscious of objects motion.

In the paper of [4], each individual establishes its own neural network, as well. But additionally, reinforcement learning method in a complex and dynamical environment is proposed during evolution.

Flocking behaviour is dynamical phenomenon. It seems to be reasonable to equip the animals in artificial world with the ability to register the objects movement. This suggestion was also the conclusion of [11] but was not implemented nor tested. With this in mind the aim of our research was to built an artificial world in which various flocking behaviour can be observed. As it has been suggested in the previous works of [11] and [16] this ability of artificial animals should be achieved with using open-ended evolution and should be caused by diversity of organisms living in the artificial world.

The paper is organised as follows. The next section presents the assumptions and details of the system Flock, where the artificial world is implemented. It enables investigation of behaviour of animals and easy settings of parameters for experiments. Next, the course of experiments and obtained results are described. The paper ends with conclusions and a suggestion of a future enhancement of the system.

2 The system Flock

As opposed to the system proposed by [15], in our system the artificial world is two dimensional. This assumption results from a greater ability of an animal behaviour observation on a computer monitor. The second reason is efficiency problem (two dimensional world is less computationally expensive). Although it is difficult to generalize the obtained results into three dimensional world, it is worth to notice that many flocking behaviour may be observed in two dimensional world, for instance, animals living in the flocks (packs) on a land.

The world in our system is settled by plants and two kinds of animals: herbivores and predators. All animals have ability to move. The plants are motionless, of course. In order to limit the speed of animals its maximal value is defined for each species: i.e. *maximal speed of herbivore* and *maximal speed of predator*, respectively, which are the parameters set by the user.

2.1 Neural controller of an individual

A behaviour of each organism, more exactly its actual value of a speed vector, is determined by a neural network (NN). Information about nearest surroundings is given on inputs of NN. Its outputs determine a change of the speed vector for a given individual. In other words, the neural network influences the motion direction and the speed of the individual. Interactions between organisms are the results of mutual location. The applied controller is three-layered feedforward neural network presented in Fig. 1. The hyperbolic tangent is applied as an activation function. In the output layer there are four neurons. The outputs of first two neurons determine motion direction change. The first one refers to the turn to the right, the second one to the left. The difference between them

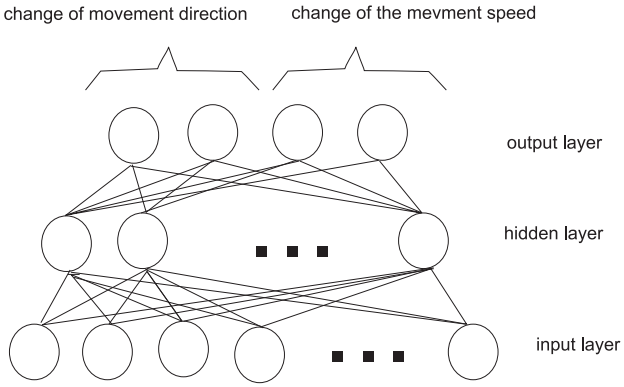


Fig. 1. The architecture of the neural controller

multiplied by the parameter *maximal angle of turn* gives the actual direction of the speed vector. The outputs of two remaining output neurons define a change of an animal speed. Similarly as in the previous case, the difference between two outputs is calculated. This value determines the change of the speed vector length of an individual. The change is limited to the values from an allowed range defined by the user.
The number of neurons in the hidden layer is set by the user, as well.

The number of neurons in the input layer depends on the way of perception of the external world by an animal. The animal is able to observe only those objects (animals or plants), which are within its reach. This means that its distance to the object is less than minimal allowed value, which is set by the user. This parameter is called *sight radius* (Fig 2). In order to give an animal information

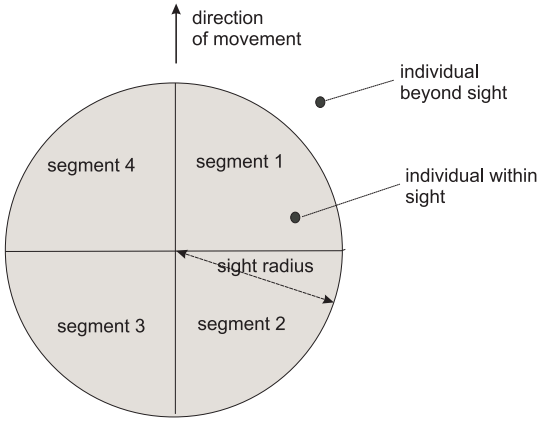


Fig. 2. The principle of vision system of animal

about surrounding objects placement the circle representing the area being in its reach is divided on a number of segments. A boundary between the first and the last segments agrees with the direction of the animal motion. In each segment the following information is essential:

- the number of plants,
- the number of herbivores,
- the number of predators.

More precisely, the location of each object is also essential. That is why information about existence of a given object in the segment is scaled depending on its distance from an animal and all these values are summed. This result is given for one neural network input. Because three different objects types are identified in the system there are three neural network inputs assigned to each segment.

Similarly as in the previous works referring to evolutionary development of collective behaviour (for instance: [16], [15]), animals are equipped with double sensorial system, but with various radiuses. The additional system has radius equal to a half of *sight radius* of the basic vision system. This approach enables delivering information about objects existing in nearest, as well as further, surroundings. In this way an animal has information, which objects that it sees are closer to it. It is expected that such double sight allows to evolve a sophisticated behaviour of animal that depends on an angle in which objects are visible for an animal but also depends on the distance of the objects from an animal.

For experimental study, the additional vision system may be switched off. According to the suggestions of [16] and [15], the system Flock has been enriched by a dynamical vision effect. It is realized by introducing to the controller information from sensor about situation in an animal neighbourhood in two consecutive time steps (from the actual time t but also from the previous time step $(t-1)$). It makes double the number of neural network inputs but enables to register dynamics of changes. In this way the animal will be able to react on the position of the surrounding it objects but also on the velocity and the direction of their motion.

2.2 Energy flow

Each animal has a given level of energy, which is decreased with every time step by the value defined by *cost of motion* parameter. New animal has a value of energy equal to 0,75 of *max energy*. This setting prevents a favouring of an individual that arrives at the end of evolution and does not represent a well adapted individual. An individual with energy equal to 0 dies and it is removed from the artificial world.

An individual loses its energy in various ways. As we mentioned above, every time step energy decreases by the value *cost of motion*. Optionally, cost of motion can be dependent on the value of an animal speed. In this case the lost of energy is directly proportional to its speed. The second possibility of energy loss is a penalty for collisions. In the case two animals from the same species will

be located in the distance less than assumed value defined by parameter *collision distance* then both of them lose energy. In this situation, the movement of both animals is not disrupted nor the speed vector is changed.

In order to increase the level of energy, the animals have to eat organism located directly under them in the food pyramid. This means that herbivores eat plants and predators eat herbivores. If a distance of an animal from its food is less than defined by a parameter *distance to food* the plant or respectively a herbivore is being eaten.

2.3 Evolution

Neural networks, which control a behaviour of animals are evolved by evolutionary algorithm. The chromosome represents weights of this neural network (Fig. 3). Its architecture during one simulation is fixed. One gene refers to a

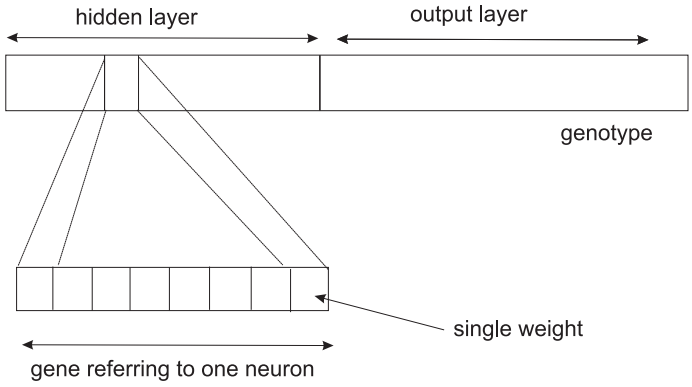


Fig. 3. The chromosome structure

given neuron. It contains weights on outgoing connections. Single weight is encoded as a real number. The number of individuals in each generation is defined by the parameter *population size*. A typical evolutionary algorithm scheme is applied. It means that in each generation, some of individuals are selected for reproduction. Then, they are exposed to genetic operations: crossover and mutation. One point crossover is implemented in the system. Mutation occurs with an assumed probability. It relies on addition or subtraction a small value to a mutated gene. Evolution in Flock system is open-ended. This requires that the only mechanism of evolution is natural selection, which consists in elimination of individuals that are not able to capture or to kill a sufficient amount of food to survive. During the first experiments with the systems it turned out that animals rewarded for staying with individuals from the same species perform better taking into account their flocking behaviour. That is why optionally the animals can be rewarded by increasing its energy proportionally to the number of animals from the own species existing within the sight of a given animal.

The animals live an assumed number of time steps. Next, they are evaluated and reproduced – a new generation is created. Organisms from the previous generation are removed from the artificial world. The measure of their fitness is the value of energy. Roulette wheel is used as a selection method ([6], [7]).

3 The course of simulation

The simulations can be proceed into three mode: real, accelerated and step by step. In the first one, the subsequent steps of simulation proceed with simultaneous visualization until the evolution is stopped on demand of the user. The accelerated mode proceed in the same way but without visualization. In the last mode one step of simulation is performed, which is shown in Fig.4.

Create a new population of individuals
While an actual number of steps in actual generation is less or equal to the *maximal number of steps in generation* do
For each individual
Give information about surrounding it objects on the inputs of its NN
Modify its speed vector according to the values of NN output
end{For}
For each individual
Change a position of individual by a value defined by its speed vector
end{For}
For each individual
If the collision with other object exists then modify its energy
end{For}
For each individual
If the value of energy is less than 0,
then
Remove this individual
Create new organism with the use of the previous generation and roulette wheel mechanism
Locate new organism randomly in artificial world
end{then}
end{For}
end{While}

Fig. 4. The algorithm in pseudocode of simulation a single step in artificial world

4 Experiments

The main criteria to evaluate our artificial world is a satisfaction of herbivores. The aim of the experiments was to test whether in this world arrive flocking behaviours of animals. Because of the large number of parameters in the system, a series of preliminary experiments was performed, the aim of which was to

determine suitable values of parameters that are invariable in the next experiments. The smallest number of simulation steps, needed to appear the evidences of animals adaptation to the artificial world, was the criterion of the choice of parameters values.

4.1 Tuning the parameters

Because the double and dynamic sight are very time consuming in this experiment they were not used. The conclusion from observation of animals behavior in the performed experiments allowed to tune the values of parameters.

The values defined after experiments performed in this series are shown below. They are constant in the next experiments.

- *The number of objects of particular species* – because of the speed of the hardware available to simulation, the number of predators was set to 20. Small population has a negative influence on efficiency of evolutionary algorithm. That is why the number of herbivores was set to 40, because it was the main species, behaviour of which was observed in the experiments. The number of plants is equal to 50. This value is enough to find them by herbivores but as much to make the task nontrivial.
- *The size of the world* – this parameter in connection with the previous ones determines population density, which has significant influence on the development of organisms in the artificial world. In the preliminary experiments a high population density caused unnatural behaviour facilitating survivals of predators. They only gathered near plants and waited for the herbivores.
- *Parameters of evolutionary algorithm* – suitable tuning of evolutionary algorithm is an essential element of success. After preliminary experiments the value of crossover probability was set to 0.8 and probability of mutation 0.001.
- *The number of steps in one generation* – evolution should be long enough in order to eliminate organisms by natural selection which are not adapted to the life in artificial world and to breed flocking behaviour of animals. Too high value of this parameter may cause quick domination of population by individuals that represent local optimum. After some preliminary experiments this value is set to 1000.
- *The number of sight segments* – with the growth of this parameter value the motion of animals becomes stabile and smooth, but more complex behaviour has been not observed. That is why this value is set to 4.
- *Maximal turn angle* – modification of this parameter generated many interesting effects. Animals were too sensitive for high value of this parameter, for instance herbivores were able to turn about-face. In the experiments this parameter is set to 0,5 Π rad
- *Energy parameters* – on the basis of preliminary experiments values of these parameters have been set as follows: $plant\ energy = 50$, $maximal\ energy\ of\ herbivore = 500$, $maximal\ energy\ of\ predator = 500$, $cost\ of\ motion = 1$.

These values were chosen in this way that the life length of individual which is not capable of getting of energy is less than a duration of one generation and so as a herbivore energy was sufficient to reach the next plant (an average distance). The energy of herbivore should be sufficient to reach the plant being in its neighbourhood.

- *Perception of world* – in this group there are three parameters: *maximal speed of herbivores*, *maximal speed of predators* and *sight of animals*. After preliminary experiments these values are set respectively: 5, 5, 50. These values allow an animal moving with maximal speed to have in its sight an area, which the animals are able to reach in 10 time units. A bigger size causes a lack of determination in animal motion. Too many stimuli goes to the animal and it reacts simultaneously with all of them, what gives bad results.
- *Reinforcement value* It is set to 0.1. Too high value of this parameter causes an incorrect performance of a neural network. Because of applied activation function, for values where hyperbolic tangent function is flat it is difficult to train the network.

The values of remaining parameters are set differently in each experiment. In all experiments performed in this series arose behaviour that allowed to survive animals in the artificial world. Typically, herbivores made their way towards plants in order to consume them as it is shown in Fig. 5. In all figures in this paper plant is represented as a small square. A herbivore is shown as an empty triangle while predator as a black triangle. The animals had no serious problems with finding plants and moving towards them. However, in some situations the

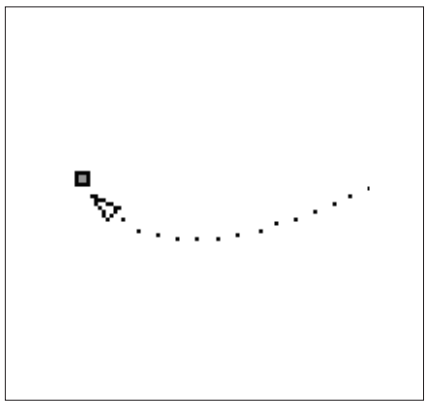


Fig. 5. The herbivore is moving towards a plant (represented as a small square). A herbivore is shown as an empty triangle

simple sight was not sufficient. The animal was not capable to manage situations where a few plants was in its neighbourhood. Sometimes it happened that animal

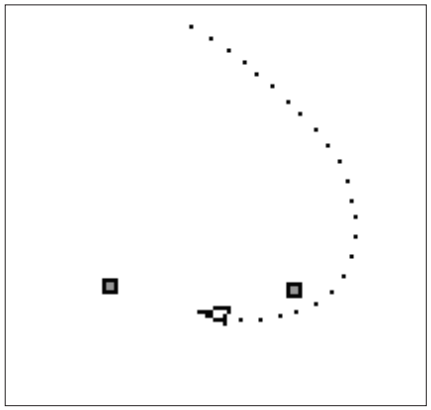


Fig. 6. The problem of herbivore (the empty triangle) with hitting a plant (a small square)

went around a subsequent plants without eating them (Fig. 6). In the result the animal cannot increase its energy.

Additionally, herbivores had a gift to avoid predator attacks. It is worth to notice that they did not apply simple dodges. The herbivore seeing close predator tried to make its way towards him in order to pass it (Fig.7). After this manoeuvre a predator was not able to turn round because of a limited turn angle and in consequence it was not able to catch a herbivore. The herbivore was rescued.

The predators have developed basic mechanisms allowing them to survive in

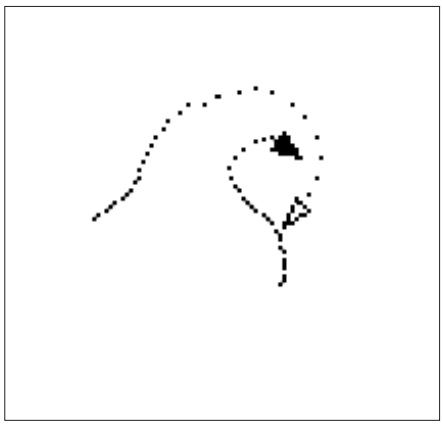


Fig. 7. A herbivore (represented as the empty triangle) tries to escape from a predator (represented by the black triangle)

artificial world, as well. Animals had ability to pursuit of herbivores until to catch them (Fig.8). Because the maximal speed of predators was equal to the maximal

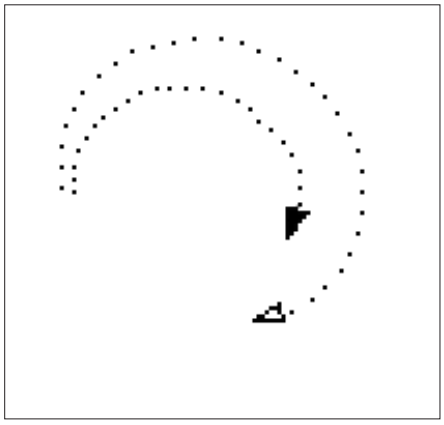


Fig. 8. A predator (the black triangle) in pursuit of a herbivore (the empty triangle)

speed of herbivores the last ones in order to catch their victim applied the method of cutting a corner, which slowly brought them to their goal. Moreover, predators apart from going after herbivores showed the tendency to move to plants (Fig.9). This phenomenon can be easily explained by higher probability to find there herbivores searching for food in this region.

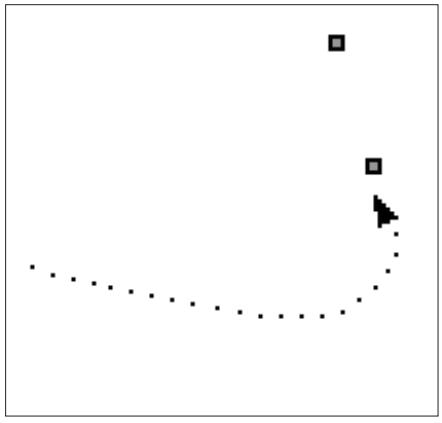


Fig. 9. A predator (the black triangle) moving towards a plant (a small square)

4.2 Dependency of cost of motion in relation of speed

The aim of this experiment was to investigate what is an influence of changes of animals energy depending on their speed on appearance of complex behaviours in our artificial world. In each time step the energy of an animal was decreased in proportion to its speed. The values of parameters that were not set in the initial experiment were amounted respectively to:

- Cost of motion depending on speed – TRUE;
- Dynamic sight – FALSE;
- Double sight – FALSE;
- Punishment for collision – 10;
- Distance to attack – 5;
- Distance to eat plant – 5;
- Distance of collision – 5.

The last three parameters define how close has to be an animal to other object in order to eat it or to run into this object. Unfortunately, the results of this experiment were not satisfying. A higher cost for moving with a higher speed has a negative influence on predators behaviour. They have small willingness

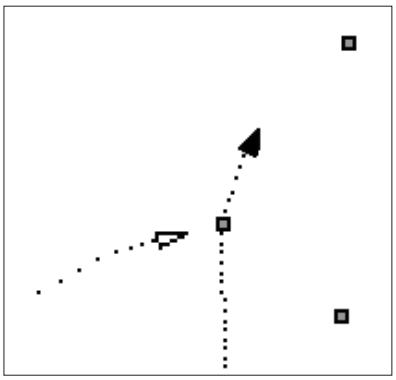


Fig. 10. The predator (the black triangle) ignores the herbivore (the empty triangle) and is moving towards a plant (a small square)

to chase of herbivores. In most cases the predators almost ignored the presence of herbivores. The plants located in their sight radius had much more influence on their motion direction. The predators most frequently followed them. It is showed in Fig.10. That is why a herbivore was eaten only by chance in the neighbourhood of plant, which both species made their way towards (Fig.11). It has been followed that a decreasing of animals energy depending on their actual speed caused the predators were tired after chasing. Because it took place with a high speed, energy resources of hunter were violated. Even the hunt has been

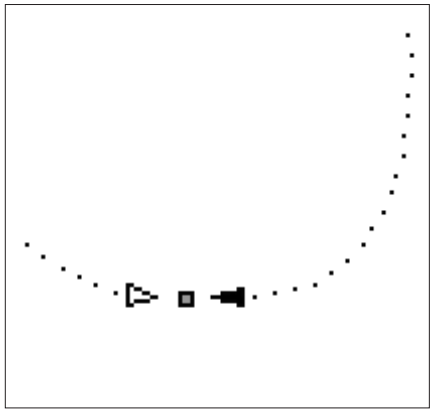


Fig. 11. The accidental collision between a herbivore and a predator nearby a plant

finished with success the victim was exhausted and its level of energy was small. For this reason it presented too small reward for the used energy. That is why in further experiments the parameter *cost of motion dependent on speed* is set to FALSE.

4.3 Elimination of collision between individuals from the same species

Preliminary experiments have shown that animals have no capability to correct a path of their motion, which resulted in collision of individuals from the same species. Often a collision of herbivores happened during a time when they brought closer to the same plant (Fig.12). Because each of them tried to be the first one nearby a plant they run across each other. Similar situations were noticed between predators during their hunting the same victim (Fig.13). Such phenomenon contradicts flocking behaviour therefore next experiments were performed in order to eliminate this undesired behaviour. They focused on balancing of profits and losses. To solve this problem the parameters referring to a collision were changed. The following parameters were tested: *punishment for collision*, *the distance to attack*, *the distance to eat a plant*, *the distance to a collision*. Each parameter was tested with values from a wide range in order to establish optimal values, which are set as follows:

- Cost of motion dependent on speed - FALSE;
- Dynamic sight – FALSE;
- Double sight – FALSE;
- Punishment for collision – $< 5; 100 >$;
- Distance to attack – $< 5; 100 >$;
- Distance to eat plant – $< 5; 100 >$;
- Distance to collision – $< 5; 100 >$.

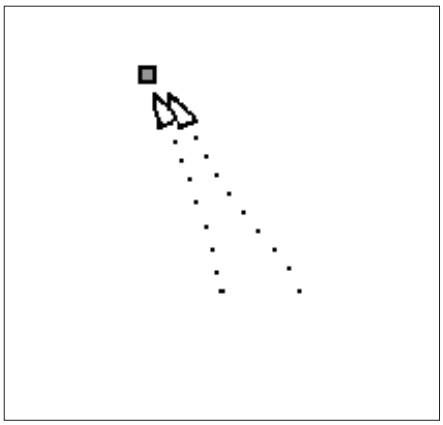


Fig. 12. A collision of herbivores nearby a plant

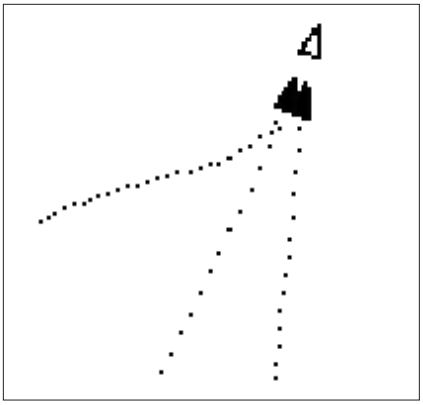


Fig. 13. A collision of predators during a pursuit of herbivore

It is worth to mention that the parameters referring to a collision were selected to ensure that the distance between animals of the same species was much larger than this one needed to catch a food (a plant for herbivores and herbivores for predators.

The best results were obtained for *punishment for collision* = 50, *the distance to attack* = 5; *the distance to eat plant* = 5; *the distance to collision* = 5. Decreasing the value of the last parameter has brought expected effects. Its values less than 5 caused that the principles managing our artificial world have become unrealistic. For instance, a predator that gained momentum moved close to a slowly moving herbivore without any harm made to it. It is because the last step before a catching of victim was too far from herbivore, the next step after overtaking a victim was also in the safe distance from a chased animal.

Unfortunately, none of flocking behaviour was observed in this series of experiments. Both, herbivores and predators have found the way to cope with problems in artificial life without any collective working. The barrier in the flocking behaviour emergence may be caused by simple sight. In all presented experiments more advanced sight system was not applied. That is why in the next experiments, the influence of double and dynamic sight was investigated. The values of parameters used in the experiments are as follows:

- Cost of motion dependent on speed – FALSE;
- Dynamic sight – TRUE;
- Double Sight – TRUE;
- Punishment for collision – 50;
- Distance to attack – 5;
- Distance to eat plant – 5;
- Distance to collision – 5.

Unfortunately, this experiment, like the previous one, does not result in flocking behaviour. On this basis we can conclude that the building of sight system of animals has no influence on their flocking behaviour. But more probable is that the proper sensor system is one of many elements that should be present in order to breed flocking behaviours.

4.4 Steered evolution

All experiments described so far have disclosed a weakness of the Flock system. The simulated artificial world does not contain sufficient mechanisms, that would stimulate flocking behaviour. In the next experiment we try to search for the reason of this problem. The system has been modified in this way that individuals were rewarded for staying close to other animals from the same species. This solution represents an approach applying steered evolution (the parameter *reward for staying close to the other organisms from the same species* set to TRUE), because the fitness function will reward individuals for concrete desired behaviour. In this experiment all plants were removed from the world. The only way to increase herbivore energy was staying close to other herbivores. The increase of animal energy in each time step was equal to the number of individuals of the same species being present in the sight radius of this individual. The parameters were set as follows:

- Number of plants – 0;
- Cost of motion dependent on speed – FALSE;
- Dynamic sight – TRUE;
- Double Sight – TRUE;
- Punishment for collision – 50;
- Distance to attack – 5;
- Distance to eat plant – 5;
- Distance to collision – 5.

This experiment showed more desired effects in comparison with the previous ones. Animals very quick gathered in small groups consisting of four individuals. It is no wonder since the main stress of fitness function was put on the decrease of distance between animals in our artificial world. The most frequently happened motion of animals is shown in Fig.14.

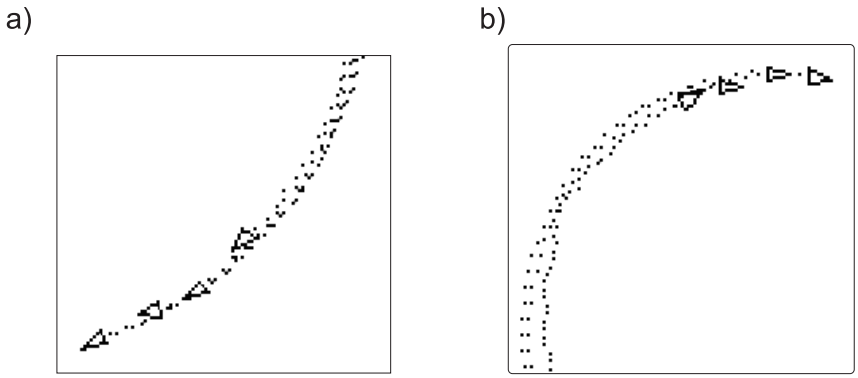


Fig. 14. Two examples of moving of herbivores in a flock

The observed results are very similar to those described by [16], but it is worth to mention that in our case the motion is not gross artificial, as it was in the cited paper. The motion of a group has been influenced of the passed objects, what caused that the group often changed direction of its motion. Therefore the formation of organisms motions may be listed here as the main weakness of the solution. The animals lined up, what resulted in motion that was similar to flocking of ducks running one by one. This experiment does not result in sufficient flocking behaviour however it confirmed the hypothesis formulated by [16]. They suggested that in order to achieve natural collective behaviour, a diversified environment is needed, in which the motion of flock will be modified by a set of stimuli.

Let us underline that the experiment was performed without plants. This may suggest that motionless objects are not essential elements for quality of behaviour manifested by herbivores. Much more crucial role play predators, mainly because they are moving objects. Predators manifested complex behaviour in this experiment also. Just like herbivores, they formed small groups moving together. The examples of their behaviour are shown in Fig. 15.

5 Conclusions and future works

In this work we tried to apply open-ended evolution to design controllers for artificial organism, which would be able to demonstrate flocking behaviours. In

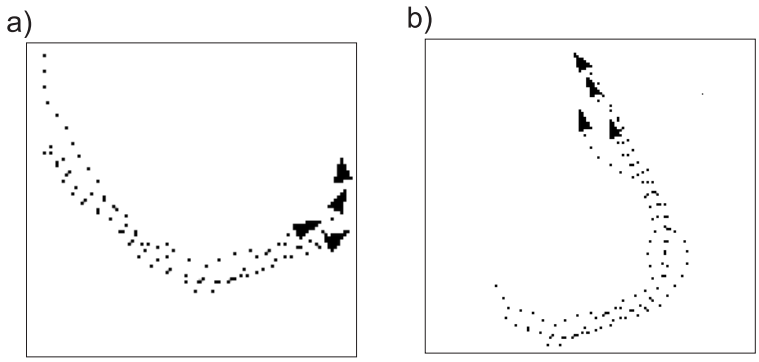


Fig. 15. Two examples of a group of predators moving in a flock

comparison to the previous works of other researchers the artificial world represented in our system was characterized by a diversity of organisms. Additionally, double and dynamic sight was introduced, which was suggested in the previous work in this area. All these attempts has the aim to breed flocking behaviours faithfully simulating the conducts existing in nature.

During performed experiments many different behaviours were observed. One of them were flocks. The observed motion of animals looked natural. The animals moved in groups reacting to the passed objects from the artificial world. To the full success, however, it was necessary to apply steered evolution.

Apart from forming a group by animals, many interesting behaviours have been manifested by the organisms. The escape of herbivore from predators, making herbivores way towards plants or a pursuit of predators after herbivores are the examples of basic reactions of animals observed during experiments. Another interesting behaviour was grouping of predators around plants, where the probability to meet herbivores is greater than in other places.

As the experiments have shown, the approach based on open-ended evolution has turned out to be insufficient solution to observe flocking behaviour. The problem in a such system is our poor knowledge about mechanisms managing flocking behaviour. The researchers are still not able to explain the reasons of forming such collective behaviour of species in nature. Therefore computer aided evolutionary simulation of this effect is still a great challenge.

Greater complexity of artificial world, where organisms can be developed stimulated by various stimuli, may be pointed out as further works in this area. Unfortunately, such solution has its own weakness. It increases the complexity of the system and results in difficulties in tuning the parameters of the system. But as suggest our experiments, the increase of diversity of simulated world in a connection with steered evolution seems to be a good solution for that purpose.

References

1. Bajec I. L., Zimic N., Mraz M., 2005. Simulating flocks on the wing: the fuzzy approach. *Journal of Theoretical Biology*, 233, pp.199–220.
2. Chen S., Fang H., 2006. Modeling and control of scalable engineering swarm. In *Proceedings of the 6th World Congress on Intelligent Control and Automation*, pp.526–530.
3. Kwong H., Jacob C., 2003. Evolutionary exploration of dynamic swarm behaviour. In *Proceedings of the 2003 Congress on Evolutionary Computation*, vol. 1, pp.367–374.
4. Lee M.-R., Rhee H., 2001. The effect of evolution in artificial life learning behaviour. *Journal of Intelligent and Robotic Systems*, 30, pp.399–414.
5. Lee S.-H., 2006. Predator's attack-induced phase-like transition in prey flock. *Physics Letters A*, 357, pp.27–274.
6. Michalewicz Z., 1998. *Genetic Algorithms + Data Structures = Evolution Programs*. Springer.
7. Michalewicz Z., Fogel D. B., 2004. *How to Solve It: Modern Heuristics*. Springer,
8. Mirabeta V., Augera P., Lett, Ch. 2007. Spatial structures in simulations of animal grouping. *Ecological Modelling*, 201, pp.468–476.
9. Oboshi T., Kato S. Mutoh A., Itoh H., 2002. Collective or scattering: Evolving schooling behaviors to escape frompredator. In *Proceedings of the eighth international conference on Artificial life*, pp. 386–389.
10. Reynolds C. W., 1987. Flocks, herds and schools: A distributed behavioral model. *Computer Graphics*, 21 (4), pp.25–34.
11. Reynolds C. W., 1994. An evolved, vision-based behavioural model of coordinated group motion. In Christopher G. Langton, editor, *in Artificial Life III, Santa Fe Institute Studies in the Sciences of Complexity*, vol. XVI. Addison-Wesley.
12. Reynolds C. W., 1999. Steering behaviors for autonomous characters. In *Proceedings of the 1999 Game Developers Conference*, pp.763–782.
13. Spector L., Klein J., Perry C., Feinstein M., 2005. Emergence of collective behavior in evolving populations of flying agents. *Genetic Programming and Evolvable Machines,*, 6: pp. 111–125.
14. Sugawara K., Watanabe T., 2002. Swarming robots—collective behavior of interacting robots. In *Proceedings of the 41st SICE Annual Conference*, vol. 5, pp. 3214–3215.
15. Ward C. R., Gobet F., Kendall G., 2001. Evolving collective behaviour in an artificial ecology. *Artificial Life*, pp. 191–201.
16. Zaera N., Cliff D., Bruten J. 1996. (not) evolving collective behaviours in synthetic fish. In P. Maes et al., editors, *From animals to animats, Fourth International Conference on Simulation of Adaptive Behaviour*, pp. 635–644. MIT Press.

Proceedings of the International Multiconference on
Computer Science and Information Technology, pp. 121 – 126
ISSN 1896-7094
© 2007 PIPS

A Hybrid Clustering Method for General Purpose and Pattern Recognition

Jan W. Owsiński [1] , and Mariusz Tomasz Mejza [2]

[1] Systems Research Institute, Polish Academy of Sciences, Newelska 6,
01 447 Warsaw, Poland
owsinski@ibspan.waw.pl
[2] Warsaw School of Information Technology, Newelska 6,
01 447 Warsaw, Poland
mejza@wsisiz.edu.pl

Abstract. The present short paper outlines a simple hybrid technique of clustering, based on the joint use of k-means and nearest neighbour algorithms. The technique starts with the k-means algorithm, performed as the first stage for an adequately high number of centroids and continues with the nearest neighbour algorithm, executed for the clusters obtained in the first stage, as the set of initial objects to be merged. It is shown on examples how the thus defined technique performs in terms of identification of relatively complex shapes.

1 Introduction

The short note is devoted to presentation of a clustering technique, which is meant to overcome some of the shortcomings of the techniques existing as of now, while retaining their strong points, known from literature. The Introduction section will highlight the premises behind and the purposes of development of the new technique.

1.1 The Search for a More Flexible Technique

Clustering, i.e. *placing the similar together and the dissimilar apart*, is not only a model of the basic intellectual activity, and not only a fundamental problem in multivariate analysis, but, first and foremost, a tool used in multiplicity of domains. Although the general clustering problem can be considered to have found an ultimate solution – if at all – through the formulations like those of [1, 3] and [4], the search for more powerful (in terms of finding "true solutions"), more adapted (to numerous specific situations) and more efficient (in terms of computational effort) techniques is still on. Due to this, and due to the varied properties of the techniques proposed, the domain of cluster analysis is still developing, also in its theoretical aspect, as witnessed, e.g., by books such as [2].

The dozens of existing approaches have each some merits in one or more of the fields mentioned, but also display, inevitably, poor performance with respect to some other ones. This, again, propels the development, especially of the narrowly designed techniques, like those specializing in definite tasks of pattern recognition, document retrieval etc. Indeed, the technique, which we propose in the present paper, goes

along these lines, while at the same time preserving some decent level of generality. The ultimate goal, of course, is a flexible technique that could be adapted to various application cases.

1.2 The Reasons for the New Development

The reasons for the development of the new technique can be summarized as follows: (1) to develop a method of clustering that providing effective means for visualization; then (2) to thereby develop a method that can be effectively used in pattern recognition; and (3) to improve on the existing techniques in more general terms. These reasons acted in the sequence as here provided: from a more modest goal to a much more ambitious one. It was, namely, hoped that effective visualization would constitute a good starting point to the other two goals, but even if it were to stop there, the exercise would be worth the effort.

It must be added that while aiming at the technique aiding in visualisation of the data sets we planned to have, at this stage of work, a simple instrument that would be tested on several cases through human verification. Actually, this is also what visualisation is about.

2 The Technique

We shall now give the complete description of the approach, with just a short consideration of the technical details, which are left to ampler future publications on the subject, linked with further developments of the method.

2.1 Notation

Assume we deal with n objects (observations, items), indexed i, $i \in I = \{1,\ldots,n\}$. Each object is described with m variables (attributes, features), of any character, and such description is denoted x_i, $x_i \in X_I$. We can postulate that these variables form the space of all potential objects, denoted E_X, $X_I \subseteq E_X$. Assume, further, that we can define a distance in E_X, with distance between objects considered denoted $d(x_i,x_j) = d_{ij}$. Distances d_{ij} form a symmetric matrix $D = \{d_{ij}\}_{ij}$.

The set I is divided (partitioned) in the clustering problem into subsets (clusters) denoted A_q, $q = 1,\ldots,p$, where p is the number of clusters, forming a partition P.

Whenever applicable, the representative object of a cluster, whether belonging to X_I, or to E_X-X_I, will be denoted x^q (it is assumed that there is only one such object per cluster).

2.2 General Scheme of the Algorithm

The algorithm is divided into two stages:

In the first stage the k-means algorithm is performed with predefined number of clusters, p_1 (user's choice) at a relatively "high" level, anyway – much higher than the

expected "ultimate" ("objective"?) number of clusters. Just as a hint, for a wide range of values of n one can use $p_1=n^{1/2}$. In this manner, clusters A^1_q are obtained, $q=1,\ldots,p_1$.

Once the first stage terminated, the matrix of distances between clusters A^1_q is calculated, D^1. On the basis of this matrix the classical progressive merger procedure of single link (nearest neighbour) is performed.

Just like p_1, the number of clusters ultimately determined might be an explicit choice of the user, p_2, or the merger procedure can be carried out to the very end, with p_2 determined on the basis of additional information.

2.3 The Rationale

The use of the two known algorithms in the way here proposed is justified by the following reasoning: It is well known that k-means form spherical or ellipsoidal clusters and converge very quickly. If the clusters formed are "small", and "dense" in the set of objects considered, the convergence is even quicker, and there is little hazard of finding a local minimum, so that either only few repetitions are needed, or they can be given up at all. By applying k-means in this way we obtain an effective breakdown of the data set into small, compact subsets, even though these subsets may have very little to do with the actual "shape" of the proper clusters sought. In the second stage single link is used, which agglomerates the subsets according to minimum distances, so that if these subsets form any linear and complex shape, it should get uncovered, with the single link algorithm not so much penalised by the necessity of maintaining and re-calculating the distance matrix, owing to the shrinking of the dimension of the problem in the first stage.

2.4 Some Technical Remarks

The primary issues of technical nature, which arise in the implementation and running of the algorithm are quite obvious:

- **i.** determination of p_1: besides the hint provided above, caution must be made of the maintenance of the reasonable (?) proportions between n, p_1 and the envisaged p_2; one might also use a constant divisor, bringing n down to p_2; this issue is, of course, closely associated with the fact that neither k-means nor single link by themselves provide a way to determine the "proper" p_2;
- **ii.** generation of the initial centroid candidates for the k-means stage: given that we start with a much bigger number of centroids than the sought number of final clusters, the initial centroid candidates can be determined by a method different from random choice in E_X (or X_I);
- **iii.** calculation of the distance matrix D^1; this is the key issue in the computational efficiency of the algorithm; in the application developed to implement the method, a user is offered three options at this point: (**1**) complete enumeration (i.e. $d(A^1_q,A^1_{q'}) = \min \{d_{ij}: x_i \in A^1_q, x_j \in A^1_{q'}\}$ is obtained on the basis of all pairs i,j such that $x_i \in A^1_q$, $x_j \in A^1_{q'}$; (**2**) the value of $d(A^1_q,A^1_{q'})$ is calculated as the d_{ij} between $x_i \in A^1_q$ that is the closest to $x^{q'}$ and $x_j \in A^1_{q'}$ that is the closest to x^q; (**3**) a predefined proportion (user's

choice) of objects in both clusters is compared conform to the scheme (2) above.

The fact that these choices are offered in determination of D^1 comes from the contribution of this phase of functioning of the algorithm to the overall computational burden, both in terms of time and memory requirements.

3 Examples

This section presents two characteristic examples of functioning of the algorithm, selected so as to show the merits of the technique and to compare it with the classical k-means.

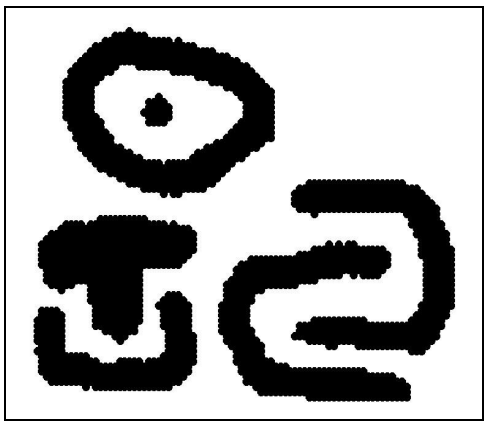


Fig. 1. A „clinical"case treated by the algorithm proposed, with $n = 2277$, $m = 2$

For the example shown schematically in Fig. 1 above, with relatively complex shapes of the clusters "to be identified", the new algorithm, with $p_1 = 227$ and distances in D^1 calculated according to option (2), allowed to identify precisely the visually obvious clusters on the basis of the aggregation distance diagram. The classical k-means, for which $p^2 = 6$ was set, was unable to produce these clusters within a reasonable number of repetitions.

The other case treated, which is shown here, had a different objective. As shown in Fig. 2, it was a "simpler" data set, in which the major difficulty was associated with the hierarchical structure.

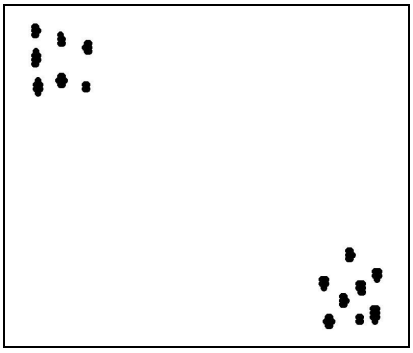


Fig. 2. A „simple" case treated by the algorithm proposed, with $n = 106$, $m = 2$

In this case the new algorithm was used with $p^1 = 45$ and, again, distances in D^1 calculated with option (2). The solution provided was that into two clusters, and the one into 15 clusters could only be identified via an additional analysis of the agglomeration diagram. In both cases, though, the clusters obtained were fully conform to eye inspection.

On the other hand, the classical k-means produced for $p^2 = 2$ the same result, in accordance with the image, and performed relatively well for $p^2 = 15$, although, for standard reasons, committed some misclassification errors.

4 Conclusions

So, a simple hybrid clustering scheme was implemented, which can be used effectively, as this was verified on a series of examples, for visualization purposes, and, in the same vein, for pattern recognition (also well beyond m=2). It can, of course, also be used as a general purpose clustering algorithm. In the examples treated to date the use of approximation in the calculation of distances D^1 for the second stage of the algorithm has not resulted in "incorrect" results.

We have on purpose not quoted here the computational statistics, because, first, these depend heavily on the choices made by the user (especially p^1) and can easily change the order of magnitudes, and second – we have been, at this stage of work, mainly interested in the uncovering of the "true" structure of data.

The work on the method is being continued. It concerns both the important technical details, commented upon in the paper, and some more general issues. These include, first of all, the final selection of p^2 on the basis of the agglomeration diagram (a classical problem, here its solution being assisted by application of the objective function from [3] and [4]), and the tuning of the second stage (e.g. a choice of the progressive merger procedure according to the Lance-Williams-Jambu formula) oriented at identification of various kinds of shapes of the clusters sought.

References

1. Marcotorchino F., Michaud P.: *Optimization in ordinal data analysis,* IBM France Scientific Centre. Technical Report, Paris (1978).
2. Mirkin B.: *Mathematical Classification and Clustering. Kluwer Academic Publishers,* Dordrecht Boston London.
3. Owsiński J. W.: *On a quasi-objective global clustering method,* In: Diday E., Jambu M., Lébart L., Pagés J., Tomassone R. (eds.): Data Analysis and Informatics III. North Holland, Amsterdam (1984) 293-306.
4. Owsiński J. W.: *On a new naturally indexed quick clustering method with a global objective function,* Applied Stochastic Models and Data Analysis, 6 (1990).

Proceedings of the International Multiconference on
Computer Science and Information Technology, pp. 127 – 139
ISSN 1896-7094
© 2007 PIPS

Application of Genetic Algorithm in Median Filtering

Sandra Sovilj-Nikic, Ivan Sovilj-Nikic

University of Novi Sad, Faculty of Technical Sciences
Trg Dositeja Obradovica 6, 21000 Novi Sad, Serbia and Montenegro
sandrasn@eunet.yu

Abstract. Images are often corrupted by impulse noise due to errors generate d in noisy sensors or communication channels. Two types of impulse noise can be defined: 1) fixed-valued and 2) random-valued. In many applications it is very important to remove noise in the images before some subsequent processing such as edge detection, object recognition and image segmentation. In this paper an adaptive filtering using genetic algorithm is proposed. In the simulations over various images, the proposed partition based median (PBM) filter using genetic algorithm in training have demonstrated better results in noise suppressing then competitive filters based on median filtering in terms of SNR(dB) as well as the perceived image quality.

1 Introduction

Images are often corrupted by impulse noise due to errors generate d in noisy sensors or communication channels. Two types of impulse noise can be defined: 1) fixed-valued and 2) random-valued. Fixed-valued impulses occur when noisy pixel values are out of receiver ' s range and they can have one of two values (MIN or MAX). This type of noise called salt&peper noise and it appears as black and/or white impulses on the image. Another type of noise generated after incorrect decoding of binary represented image and in this case noisy pixels can have arbitrary value.

In many applications it is very important to remove noise in the images before some subsequent processing such as edge detection, object recognition and image segmentation.

At the beginning of development of techniques in digital image processing, linear techniques are used extensively because of their mathematical simplicity and the existence of appropriate characteristics (e.g. principle of superposition) making them easy to design and implement. In the case where noise can be modulated as additive Gaussian noise linear techniques offer satisfactory performance for noise removal. However, in many cases the noise is impulsive and in this case linear techniques do not usually perform well. Another example where linear techniques fail is the case of nonlinear image degradations. Such degradations occur during image formation and during image transmission through nonlinear channels. The human visual perception mechanism has been shown to have nonlinear characteristics. Human vision is very sensitive to high-frequency information such as edges and fine details on the image (e.g. lines and corners). Therefore, high-frequency content of image is very important

for visual perception. However, most of linear filters have low-pass characteristics. They tend to blur edges and to destroy lines and other fine details of image.

All above mentioned reasons led to leave linear techniques and to the use of nonlinear filtering techniques. One of nonlinear filters family includes filters based on order statistics. The family of order statistics based filters is very rich, where the median filter is the best known. Median filter and its modification [11] have shown good efficiency in suppressing impulse noise and capability of preserving image edges. Nevertheless, because the median filters are location-invariant, i.e. they are implemented uniformly across the entire image. They tend to alter both noise pixels and undisturbed good pixels. Therefore, they remove fine details in the image.

Partition-based filtering is one of the concepts to exceed disadvantages of median filtering. In this paper an adaptive filtering using partition based median (PBM) filter is proposed. With proposed PBM filter, at each location, observed vector is classified into one of M exclusive partitions, and a particular filtering operation is then activated. The observation vector space is formed on differences between current pixel value and the outputs of the center weighted median (CWM) filters with variable center weights. The estimate at each location is formed as a linear combination of current pixel value and the outputs of CWM filters. Optimal weighting vector of each partition is derived using genetic algorithm in training. Optimal weights are derived to minimize the total square error of estimation at each location. In the simulation over variety of images, the proposed PBM filter using genetic algorithm in training on reference image has demonstrated better results in noise suppressing then competitive filters based on median filtering in terms of the SNR (signal/noise ratio) as well as the perceived image quality. The proposed filter outperforms other median based filters in removing both fixed-valued and random-valued impulses

2 Definition of PBM filter

Schematic diagram of PBM filter is shown on Figure 1 [4].

Let $C=\{(c1,c2) \mid 1\le c1\le P,\ 1\le c2\le Q\}$ denote pixel coordinates of a digital image, where P and Q are its height and width, respectively. At each location $\mathbf{c}(c1,c2)\in C$ a filter window of size $W=2n+1$ simetrically surrounding the current pixel, and

$$\mathbf{x(c)}=\{x_i(c): i=1,2,...,2n+1\} \tag{1}$$

denotes set of observed pixels via filter window, where $x(c)=x_{n+1}$ is original or center pixel.

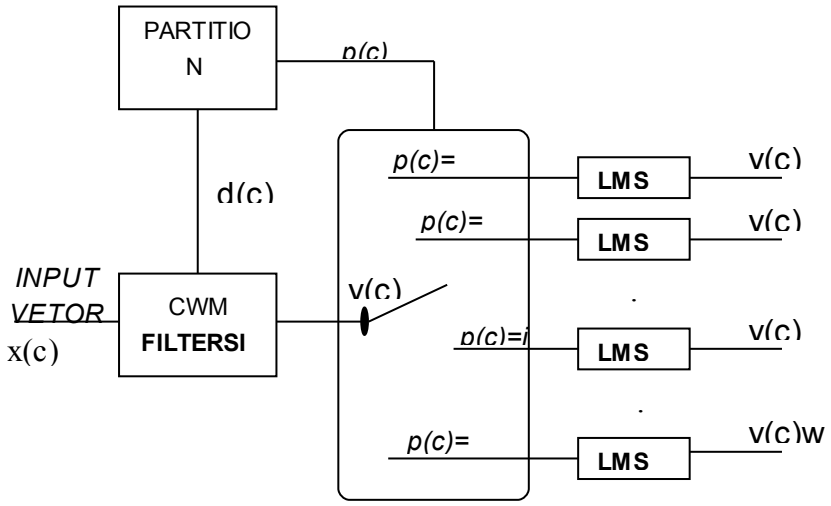


Fig. 1. - Schematic diagram of PBM filter

Another step in filtering is to apply CWM filters with variable center weights on input vector $\mathbf{x(c)}$. The output of CWM filter is described as:

$$y_k(c) = median(\mathbf{x}_{2k+1}(c)) = median(\mathbf{x}_{\omega}(c)) \tag{2}$$

where ω denotes center weight and $\omega = 2k+1$, k is nonnegative integer.

$$x_{\omega}(c) = \{x_1(c), x_2(c), x_3(c), .., x_n(c), \omega \quad x_{n+1}(c), x_{n+2}(c), .., x_{2n+1}(c)\} \tag{3}$$

Operator $\diamond$ denotes repetition operation. Two limit cases exist in the CWM filtering. Output of CWM filter y_0 (*k=0, w=1*) corresponds to the output of standard median filter, and for $k \geq n$ CWM filter is an identity filter $y_k = x(c)$. In the proposed filtering scheme k lies in $[0, n-1]$.

At each location, for the current pixel, the following differences are defined:

$$d_k(c) = |y_k(c) - x(c)| \tag{4}$$

where $k = 0, 1, \ldots, n-1$ and $\mathrm{d_k}(c) \leq \mathrm{d_{k-1}}(c)$.

At each location $\mathbf{c}$, the estimation of pixel value is derived as a linear combination of outputs of CWM filter with variable center weights and a current pixel value, which is equivalent to the output of the CWM filter with $k = n$ as follows:

$$\hat{x}(c) = \mathbf{y(c)} \cdot \mathbf{w}^{\mathbf{T}}_{p(c)} \tag{5}$$

where $p(c)$ corresponds to the partition index for $\mathbf{x(c)}$ which will be defined later and $\mathbf{y(c)}$ is the input vector for filtering at observed location:

$$y(c) = [y_0(c), y_1(c), \ldots, y_{n-1}(c), y_n(c)] = [y_0(c), y_1(c), \ldots, y_{n-1}(c), x(c)] \tag{6}$$

Vector $\mathbf{w}_{p(c)} = \mathbf{w}_i = [w_{i,o}, w_{i,1}, \ldots, w_{i,n}]$ represents weighting vector of the particular filter for the *i* th partition $(i = 1, 2, \ldots, M)$. Optimal weighting vector for each partition is derived using genetic algorithm.

To control the dynamic range of outputs, weighting vector satisfies a location-invariance constraint:

$$\mathbf{e}_{n+1} \cdot \mathbf{w}_i^{\mathbf{T}} = \mathbf{w}_i \cdot \mathbf{e}_{n+1}^{\mathbf{T}} = 1 \qquad (i = 1, 2, \ldots, M) \tag{7}$$

where $\mathbf{e}_N = [1,1, \ldots, 1]$ is $1 \times N$ vector.

If $x(c) = median(\mathbf{x(c)})$, then $y_k(c) = x(c)$ for any $k \geq 0$ and current pixel $x(c)$ remains unaltered. In this case, let $p(c) = i$, we have:

$$\hat{x}(c) = y(c) \cdot w_i^T = e_{n+1} \cdot x(c) \cdot w_i^T = x(c) \tag{8}$$

2.1 Region partitioning

At each location $\mathbf{c}$, difference vector is obtained by applying (2.4) as follows:

$$d(c) = [d_0(c), d_1(c), \ldots, d_{n-1}(c)] \in R^n. \tag{9}$$

The difference vector space is divided into *M* exclusive regions, $\{\Omega_i : i = 1, 2, \ldots, M\}$. The pertition index for each input vector $\mathbf{x(c)}$ is given such that $p(c) = i$ for $\mathbf{d(c)} \in \Omega_i$. Partition in regions can be fulfilled by different methods, e.g. the scalar quantization, the vector quantization etc. Due to its simplicity and computational efficiency, scalar quantization is applyed in this paper. Quantization levels are determined experimentally and remain fixed through simulations.

Let $\{q_{k,v} : v = 0, 1, \ldots, L\}$ be a set of monotonically ascending points, i.e.:

$$q_{k,v} < q_{k,v+1} \quad , \qquad 0 \leq v \leq L-1 \tag{10}$$

for the *k* th dimension of difference vector space, where $k = 0, 1, \ldots, n-1$.

At each location, for each input vector $\mathbf{x(c)}$ and appropriate difference vector $\mathbf{d(c)}$ is defined vector of the quantization level indices as $\mathbf{z}_i = [z_{i,0}, z_{i,1}, \ldots, z_{i,n-1}]$ for $i = 1, 2, \ldots, M$.

In the other words scalar quantization is described as:

$$\mathbf{Q}(\mathbf{d(c)}) = \mathbf{z}_i \tag{11}$$

where $z_{i,k} = v$ if $q_{k,v} \le d_k(c) \le q_{k,v+1}$ for $k = 0, 1, \ldots, n-1$.

In that way partitioning of n -dimensional space in $M = L^n$ regions is fulfilled, and unique vector $\mathbf{z}_i$ defines a distinct region for $i = 1, 2, \ldots, M$. Experimentally determined quantization levels for 3×3 filter window are given in Table 1 [4]

Table 1: Quantization levels for $\{q_{k,v} : 0 \le k \le n-1,\ 0 \le v \le L\}$ obtained via a 3×3 $(n = 4, L = 6)$ filter window

	v						
k	0	1	2	3	4	5	6
0	0	5	20	35	50	70	256
1	0	5	15	25	35	55	256
2	0	5	10	15	25	35	256
3	0	2	5	10	15	20	256

2.2 LMS weights optimization

Optimal weights are determined to minimize the mean square error, or equivalently the total sqare error:

$$\varepsilon = \sum_{c \in C} \left[s(c) - \hat{x}(c) \right]^2 \tag{12}$$

where $s(c)$ and $\hat{x}(c)$ denote the original pixel value and its estimate, respectively.

Taking partitioning of n -dimensional space in M exclusive regions imto consideration we can write:

$$\varepsilon = \sum_{i=1}^{M} \left\{ \sum_{c:p(c)} \left[s(c) - \hat{x}(c) \right]^2 \right\} \tag{13}$$

where the inner sum, i.e. ε_i is error attributed to the pixels classified into the i th partiotion. Thus, entire error ε can be minimized by achieving the independent minimization of ε_i for $i = 1, 2, \ldots, M$

Following (5) we have:

$$\varepsilon_i = \sum_{c: p(c)=i} [s(c) - \hat{x}(c)]^2 = \sum_{c: p(c)=i} [s(c) - y(c) \cdot w_i^T]^2 \qquad (14)$$

Optimization problem is to find optimal weighting vector $\mathbf{w}_i$ of i th partition for $i = 1, 2, \ldots, M$ and ε_i of each partition should be minimal.

In this paper genetic algorithm is used to derive optimal weighting vector $\mathbf{w}_i$ for each partition.

2.3 Recursive implementation

In this paper PBM filter is implemented recursively because of obtained results through simulations on variety of images. Superior results have been produced by the recursive PBM filter in removing noise than nonrecursive design. The estimate of current pixel in recursive filtering depends on past filter outputs, with input

$$x'(c) = \{\hat{x}_1(c), \hat{x}_2(c), \ldots, \hat{x}_n(c), x_{n+1}(c), x_{n+2}(c), \ldots, x_{2n+1}(c)\} \qquad (15)$$

In the recursive PBM filtering the partition index for current pixel depends on previous filter outputs.

Recursive design of PBM filtering is shown on Figure 2 [4].

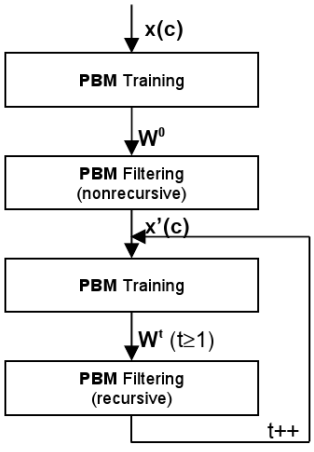


Fig. 2. Schematic diagram of recursive PBM filter

3 Optimization using genetic algorithm

Evolution programs are methods for solving optimization problems which based on the principles of Darwin's evolution theory, i.e. natural selection and survival of the fittest. The best known algorithms in this class include genetic algorithm, genetic programming, evolutionary programming, evolution strategies, classifier systems, and neural networks. All above mentioned algorithms are based on the same concept-simulating the evolution of organisms of some population through selection, recombination and mutation process.

Taking the type of optimization problem in this paper into consideration, genetic algorithm has been choose n to solve the problem.

3.1 Training process

Matrix of weighting vectors for each partition is obtained by training the filter over reference image. Optimal weights for each partition are obtained using genetic algorithm and estimate error of current pixel should be minimal for each partition.

3.1.1 O ptimization function by applying genetic algorithm

Genetic algorithm has been applied for each partition and final result of that procedure is a matrix which consists of weighting vectors for each partition. First step in application of genetic algorithm is to choose population size, i.e. number of individuals in the population. Typical values of population size are from 20 to 500 individuals. Simulations on different values of population size have been shown that population of 120 individuals is compromise solution for the problem in this paper. Individuals of population are generated randomly using Matlab's random numbers generator. Each individual represents potential problem solution, i.e. optimal weighting vector $\mathbf{w}$ which is one 1×5 vector. Columns of vector $\mathbf{w}$ are chromosomes (genes) of individuals. Range of values for individual chromosomes is $[-1, 1]$ and weighting vectors satisfy location-invariance constraint

$$\mathbf{e}_{n+1} \cdot \mathbf{w}^{\mathbf{T}} = \mathbf{w} \cdot \mathbf{e}_{n+1}^{\mathbf{T}} = 1 \tag{16}$$

where $\mathbf{e}_N = [1, 1, \ldots, 1]$ is an $1 \times N$ vector.

After creating an initial population should evaluate fitness function for each individual in the population, In this case fitness function is a total square error ε, i.e. deviation between estimate of current pixel value and original (uncorrupted) pixel value:

$$\varepsilon = \left(s - \mathbf{y} \cdot \mathbf{w}^{\mathbf{T}}\right)^2 \tag{17}$$

where is:

s uncorrupted pixel value

$\mathbf{y}$ vector of CWM filter outputs

$\mathbf{W}$ weighting vector (individual of population)

After generating initial population comes iterative procedure of genetic operators (selection, crossover and mutation) application to individuals until convergence criteria would not be satisfied. If convergence criteria is not satisfied the fittest individual in the current population is found. That individual has been kept in the memory. Then, a new population is formed by selecting the more fit individuals. In this paper tournament selection is applied to select potential parents. Using Matlab's random numbers generator two parents are choosed randomly and they create their two offsprings in the crossover operation. Crossover enables to exchange information between different potential solution. In this paper decade encoding and real arithmetic crossover is applyed.

Crossovered offspring genes are formed as:

$$c1(i)=\frac{k1\cdot r1(i)+k2\cdot r2(i)}{2}$$
$$c2(i)=\frac{k2\cdot r1(i)+k1\cdot r2(i)}{2} \tag{18}$$

where are:

$c1(i)$ and $c2(i)$ offspring genes at *i* th position

$r1(i)$ and $r2(i)$ parent genes at *i* th position

for $i = 1, 2, \ldots, 5$

Coefficients *k1* and *k2* are selected randomly from range $[0.9,\ 1.0]$, taking the limitation for range of values of individual genes into consideration. Individual genes, i.e. elements of vector **w** have to be in range $[-1,\ 1]$. Offsprings become members of population instead two individuals which have the largest value of fitness function. Then using Matlab's random generator one individual is selected and also one its gene which will be alter. Mutation introduces new genetic material into population. Mutated gene of individual is formed as:

$$c1(i) = k \cdot r1(i) \tag{19}$$

where is:

$c1(i)$ mutated gene of individual at *i* th position

$r1(i)$ parent gene at *i* th position

$i \in [1, 5]$

Coefficient *k* is selecting randomly from range $[0.9,\ 1.0]$ with the same reason as crossover operation.

After mutation convergence criteria is examined. New population, after application of genetic operators, becomes initial population for next generation. The worst individual in the current generation is replaced with a best one in the previous generation. Then, convergence criteria is tested and iterative procedure of applying genetic operations is repeated until convergence criteria would not be satisfied. Simulation have been shown that genetic algorithm converges after 50 generations. Satisfying of convergence criteria represents the end of genetic algorithm. And a result is the the fittest individual (with an optimal fitness function) i.e. optimal weighting vector **w** which in a linear combination with vector **y** gives the minimal error of current pixel value estimate.

4 Simulation results and comparison with competitive filters

Performance of PBM filter has been evaluated by simulations on variety of 256×256 test images. In simulations 3×3 filter window is used, and then $W=2n+1$, i.e. n =4. Therefore, $k=0, 1, 2, 3$ and the difference vector space is 4-dimensional. Quantization levels are determined experimentally and they are shown in Table 2.1. Each dimension has $L=6$ intervals in the range of $[0, 255]$ and the total number og regions is $M=L^n=6^4=1296$. Simulations have been fulfilled on variety of images which have been corrupted with different type of noise:

- IMPULSE NOISE
- GAUSSIAN NOISE
- MIXED GAUSSIAN AND IMPULSE NOISE

For a corruption by impulses with a noise ratio p, only p of total pixels are replaced with impulses and the others keep noise-free. Here, both fixed-valued and random-valued impulses are used. For gray-scale images, noise intensity in the first case corresponds to 0 or 255 with equal probability (i.e. p /2), while in the second case, it is uniformly distributed within $[0, 255]$. Another type of noise assumed is the zero-mean additive Gaussian noise with standard deviation σ. The mixed noise is also used by adding Gaussian noise and the fixed-valued impulse noise together.

In this paper PBM filter is implemented recursively, because of better results in SNR (signal-noise ratio) obtained by the recursive filtering over nonrecursive design. Results of recursive filtering are shown on Figure 3 for various images corrupted by fixed-valued impulses with p =20%.

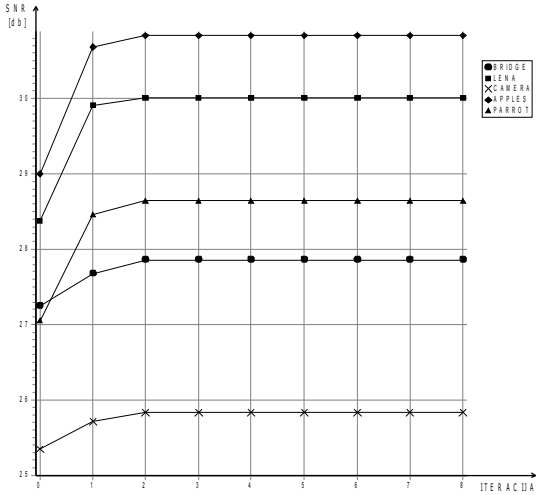


Figure 3 – Results of recursive PBM filtering for different images corrupted by fixed-valued impulses with p=20%

Noticable gain has been achieved by recursive filtering and its value is image depen dent. It is interesting to note that recursive filtering converges after two iterations of training. This can see on Figure 3, too.

4.1 Comparison with competitive filters

The performance of PBM filter is examined through comparison with competitive filters based on median filtering, including median filter (MED), CWM filter ($\omega = 3$) in recursive and nonrecursive design, fuzzy median filter, florencio&schafer, SDROM (state dependent rank order median) filter in recursive and nonrecursive design and nonrecursive PBM filter.

The comparative SNR results of filtering the *Bridge* image are given on Figure 4 where *p* ranges from 10% to 30%. The PBM filter is trained assuming the corruption by 20% impulses, while the same type of noise is used in training and filtering.

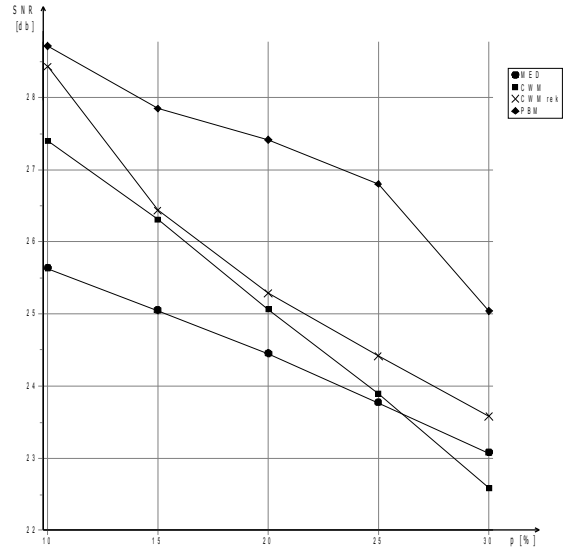


Figure 4 – Performance of different filters in filtering the Bridge image corrupted by random-valued impulses with p=20%

The PBM filter yields significant improvement over the other filters. Proposed filter has demonstrated exellent robustness in respect to impulse ratios, regardless of that used in the training.

Tables 2 and 3 present the comparative SNR results of removing different type of noise on various images. The PBM filter yields better results in suppressing impulse noise as well as Gaussian noise.

Table 2 - Comparative results in SNR (dB) of filtering different images corrupted by fixed-valued impulse noise with p=20%

FILTER	IMAGE				
	BRIDGE	LENA	CAMERA	APPLES	PARROT
MEDIAN	24.00	26.97	24.20	27.37	25.87
CWM	24.08	25.96	24.39	24.68	23.78
CWM RECURSIVE	24.53	27.36	25.30	27.81	26.03
FLORENCIO &SCHAFER	24.29	24.67	23.85	24.24	23.97
SDROM	25.43	27.55	24.54	27.77	26.20
SDROM RECURSIVE	26.71	29.51	25.48	30.41	28.35
FUZZY MEDIAN	27.42	28.70	25.67	28.25	28.41
PBM NON-RECURSIVE	27.25	28.38.	25.35	29.00	27.05
PBM RECURSIVE	27.86	30.01	25.84	30.84	28.64

Table 3. - Comparative results in SNR (dB) of filtering different images corrupted by Gaussian noise with $\sigma = 20$

FILTE R	IMAGE				
	BRIDGE	LENA	CAMERA	APPLES	PARROT
MEDIAN	24.37	26.95	25.15	27.75	26.07
CWM	25.14	26.73	25.75	27.10	26.21
CWM RE CUR-SIVE	25.11	27.20	26.00	27.78	26.53
FLORENCIO &SCHAFER	23.34	24.17	23.95	24.39	23.84
SDROM	22.90	23. 76	22.84	24.07	23.44
SDROM RE CUR-SIVE	22.95	23.90	22.76	24.14	23.51
FUZZY MEDIAN	22.75	22.8 0	22.0 6	22.81	22.67
PBM NONRE-CURSIVE	26.13	27.53	26.63	27.87	27.02
PBM RECURSIVE	25.96	27.71	25.57	28.11	27.11

The performance of PBM filter in reducing fixed-valued impulse noise can notice on Figure 5. PBM filter yields better subjective quality with respect to noise suppression and detail preservation than the other methods, by producing a visually more pleasing image.



Figure 5. - Comparison of the restoration performance of different filtering type for the Bridge image corrupted by 20% fixed-valued impulses

5 Conclusion

In this paper an adaptive filtering using partition based median filter is proposed. With proposed PBM filter, at each location, observed vector is classified into one of *M* exclusive partitions, and a particular filtering operation is then activated. Optimal weighting vector of each partition is derived using genetic algorithm in training the filter over a reference image. Recursive implementation of the proposed filter is applied and shown to produce better results than its nonrecursive design. In the simulations over variety of images, the proposed PBM filter using genetic algorithm in training has demonstrated better results in suppressing different types of noise than competitive filters in terms of the SNR (dB) as well as the perceived image quality.

References

1. Balič J., *Manufacturing Systems for the Third Millenium,* University of Maribor, Faculty of Mechanical Engineering, Laboratory for Intelligent Manufacturing Systems, Maribor, 2001.

2. Brezočnik M., *Uporaba genetskega programiranja v inteligentnih proizvodnih sistemih,* Univerza v Mariboru, Fakultet za strojništvo, Maribor, 2000.
3. Brezočnik M., Balič J., *Comparison of Genetic Programming with Genetic Algorithm,* University of Maribor, Slovenia
4. Chen T., Wu R. H., *Application of Parition-Based Median Type Filter for Suppressing Noise in Images,* IEEE Trans. Image Processing, vol. 10, pp. 829-836, June 2001.
5. Florencio D. A. F., *Decision-based median filter using local statistics*, Proc. SPIE Symp.Visual Communication Image Processing, vol. 2038, pp. 268-275, September 1994.
6. Herzeg D., *Modelovanje frekvencijski zavisnih impedansi uz pomoć genetskog algoritma,* Novi Sad, 2001.
7. Ko S. J., Lee Y. H., *Center weighted median filters and their application to image enhancement,* IEEE Trans. Circuits System, vol. 38, pp. 984-993, September 1991.
8. Kotropoulos C., Pitas I., *Nonlinear Model-Based Image/Video Processing and Analysis,* John Wiley&Sons, 2001.
9. Kuzmanović S., Dautović S., *Genetički algoritam za određivanje minimuma udruživanja na grafu,* Fakultet tehničkih nauka, Univerzitet u Novom Sadu, 1997.
10. Novak L., Dautović S., Kuzmanović S., *Genetski program za približno rešavanje problema nalaženja minimum udruživanja na grafu,* Fakultet tehničkih nauka, Univerzitet u Novom Sadu, 1997.
11. Pitas I., Venetsanopoulos A., *Nonlinear Digital Filters-Principles and Applications,* Kluwer International Series, 1991.
12. Pitas I., Venetsanopoulos A., *Order Statistics in Digital Image Processing,* Proceedings of the IEEE, vol. 80, no. 12, December 1992.

Proceedings of the International Multiconference on
Computer Science and Information Technology, pp. 141 – 150
ISSN 1896-7094
© 2007 PIPS

Linguistic Knowledge Representation for Stochastic Systems

Anna Walaszek-Babiszewska

Opole University of Technology,
Institute of Control and Computer Engineering,
Sosnkowski 31, 45-272 Opole, Poland
a.walaszek-babiszewska@po.opole.pl
http://www.po.opole.pl

Abstract. The paper deals with an idea of a linguistic knowledge representation and a linguistic inference. Relational linguistic fuzzy model with weights of rules is utilising. The probability of linguistic values of antecedent and consequent variables, calculated according to Zadeh's definition, is proposed to formulate a linguistic fuzzy model of a stochastic system. The linguistic inference procedures and the exemplary MISO model are presented.

1 Introduction

Linguistic fuzzy knowledge-based systems, linguistic preparation of information and linguistic inference procedures are based on the notion of a linguistic variable [1]. Real knowledge-based systems involve both type of information: numeric and linguistic. Knowledge from human experts is expressed in linguistic categories, and information discovered from experiments is numerical in its character. Recently, there is an increasing interest in using both sources of information for creating models of a high accuracy, flexible, and compatible to the features of real systems. It is necessary to define the interfaces for the clear preparation both types of information [2], by using fuzzy meaning and fuzzy description of a numeric value, defined in the work of Zadeh [1].

Stochastic modelling of real economic, biological or technological systems has been developed by many authors, and there are many stochastic models created under different assumptions and hypotheses. In the same time and in the same environment of imprecision, knowledge of human experts is a base for rational decisions, or risk estimations, expressed by market investors in a form of linguistic variables, sentences or rules.

This paper discusses possibilities of utilising linguistic fuzzy model ling and linguistic inference for stochastic vector process. The idea of a linguistic variable [1] has been used for determining linguistic states of the process. The template modelling [7] has been used for the partition of the input-output space and to determine the sets of linguistic variables. The empirical probability distributions of linguistic variables of the model are calculated according to Zadeh's definition [3]. Considered linguistic fuzzy model can be treated as a fuzzy relational model with weights [4, 5]. Weights of

the model represent probabilities of fuzzy events in antecedents and consequences of rules [6].

2 Representations of a Linguistic Variable

Linguistic variable which takes values in a form of words or sentences is used for expressing the linguistic knowledge representation about the behaviours of examined systems.

According to Zadeh [1] (also [8] and [2]), the definition of a linguistic variable can be expressed by the quintuple:

$$< x_{name}, L(X), X, G, M > \tag{1}$$

where

x_{name} – the name of the linguistic variable, e.g., *'temperature'* ;

X – the universe of discourse, the domain of the examined variable, e.g. $X = [-40,40] \in R$;

$L(X)$ – the set of linguistic values (terms) that x_{name} takes, e.g. $L(X)=\{low, middle, high\}$;

G – a syntactic rule for generating the names LX from the term-set $L(X)$;

M – a semantic function that assigns to each linguistic value LX a fuzzy meaning $M(LX)$, as a fuzzy subset of X, e.g.:

$$M(low) = \{(x, \mu_{low'}(x)), x \in X\} \tag{2}$$

where $\mu_{low'}(x)$ is a grade of membership of $x \in X$ in $M(low)$.

The membership functions define the family fuzzy subsets in X, for every linguistic value from $L(X)$, e.g.:

$$\mu_{low'}(x) = \begin{cases} 1, & x \in [-40,-20) \\ \dfrac{-x}{20}, & x \in [-20,0] \end{cases}, \tag{3}$$

$$\mu_{middle'}(x) = \begin{cases} \dfrac{x+20}{20}, & x \in [-20,0) \\ \dfrac{20-x}{20}, & x \in [0,20] \end{cases}, \tag{4}$$

$$\mu_{high'}(x) = \begin{cases} \dfrac{x}{20}, & x \in [0,20) \\ 1, & x \in [20,40] \end{cases}. \tag{5}$$

In many applications we use the linguistic vector variable $x^T_{name}=(x^1_{name},\ldots,x^p_{name})$, which is expressed by p linguistic variables considered in the universe of discourse, e.g. $\boldsymbol{X}=R^p$; $\boldsymbol{x}_{name}$ is the name of the linguistic vector, e.g. ' *state of market* ', ' *technological situation* ' or ' *material characteristics* '; $\boldsymbol{L}(X)$ is the set of linguistic values, e.g. $\boldsymbol{L}(X)=\{weak, good, very\ good\}$, $\boldsymbol{L}(X)=\boldsymbol{L}(X_1) \times \ldots \times \boldsymbol{L}(X_p)$.

In knowledge-based systems we can distinguish two dual interfaces, in which the two dual functions are implemented:

- information processing from the linguistic form into numeric, according to the fuzzy meaning M,
- information processing from the numeric form into linguistic, by using the fuzzy description D.

The fuzzy description $D(x)$ of a crisp value x is a discrete fuzzy subset on the set of linguistic values $L(X)$, defined as [1], [2]:

$$D(x)=\left\{\left(LX,\mu_{D(x)}(LX)\right),\ LX\in L(X)\right\} \tag{6}$$

where $\mu_{D(x)}(LX)$ is the degree of membership of the linguistic value LX in a discrete fuzzy set $D(x)$.

The following relationship between the membership functions is fulfilling:

$$\mu_{M(LX)}(x)=\mu_{D(x)}(LX),\quad \forall x\in X,\ \forall LX\in L(X)\,. \tag{7}$$

For example, the fuzzy description of the crisp value x =15, for linguistic variable *temperature* given above, is a fuzzy subset

$$D(15)=\{(low,0),\ (middle,0.25),\ (high,0.75)\} \tag{8}$$

or in another notation

$$D(15)=0/low+0.25/middle+0.75/high\,. \tag{9}$$

3 Stochastic System with Linguistic States

Consider a certain real system, represented by the ordered pair (x,y) of input and output (states) variables:

$$\left\{\left(x(t,\omega),y(t,\omega)\right):\ t\in T,\ x\in X,\ y\in Y,\ \omega\in\Omega\right\} \tag{10}$$

where X is the system input domain, Y is the output (states) domain of the system, T represents a time domain, and Ω is an elementary events domain. There is a certain probabilistic, reason-result relationship between variables x and y, where x

plays the role of a reason, and y – the result. We assume the probabilistic measure $p(x,y)$ on a set of realizations of the processes $\{x(t), y(t)\},\ t=t_k, k=1,2,\ldots K$ observed at the discrete moments.

There are many models of stochastic systems, for example an input-output dynamic model, discrete in a time domain T:

$$y(t_k)=f[x(t_k), x(t_{k-1}),\ldots, x(t_{k-n}),\ldots, y(t_{k-1}),\ldots, y(t_{k-m})] \qquad \textbf{(11)}$$

where $f()$ can be a multivariable regression function. These types of models are well known as Box-Jenkins' time series models [9].

Other types of models take into account a multivariable distribution function of the processes $\{x(t), y(t)\},\ t=t_k, k=1,2,\ldots K$ observed at the discrete moments

$$p(x,y)=p[x(t_k), x(t_{k-1}),\ldots, x(t_{k-n}),\ldots, y(t_k), y(t_{k-1}),\ldots, y(t_{k-m})] \qquad \textbf{(12)}$$

.

These models are used in more simple forms, as the first order models (e.g. white noise) or the second order models (e.g. Markov's process).

Consider now (x,y) as a pair of linguistic variables and two types of fuzzy models that can give a fuzzy representation of the stochastic process: Mamdani type fuzzy model (13) and Takagi-Sugeno fuzzy model (15) (according to [10], [5]):

$$\begin{aligned} &w_i[\ IF\ x(t_k)\ is\ A_{i,k}\ AND\ x(t_{k-1})\ is\ A_{i,k-1}\ldots AND\ x(t_{k-n})\ is\ A_{i,k-n}\ldots \\ &\ AND\ y(t_{k-1})\ is\ B_{i,k-1}\ldots AND\ y(t_{k-m})\ is\ B_{i,k-m} \\ &THEN\ y(t_k)\ is\ B_{i,k}\ ] \end{aligned} \qquad \textbf{(13)}$$

where $i=1,\ldots,I$ – numbers of rules, x,y –linguistic variables, $x\in X$, $y\in Y$ with linguistic values sets $L(X)$, $L(Y)$, determining linguistic states of the system, $A_{i,k}, A_{i,k-1},\ldots, A_{i,k-n}$ - fuzzy subsets corresponding to linguistic values of variables $x(t_k), x(t_{k-1}),\ldots, x(t_{k-n})$, $x\in X$, $B_{i,k}, B_{i,k-1},\ldots, B_{i,k-m}$ - fuzzy subsets corresponding to linguistic values of variables $y(t_k), y(t_{k-1}),\ldots, y(t_{k-m})$, $y\in Y$, w_i - weight of the *i-th* rule, probability of the fuzzy event representing the fuzzy relation R_i in $X^{n+1}\times Y^{m+1}$ (according to [6, 11]):

$$w_i=P(R_i)=P(A_{i,k}\times A_{i,k-1}\times\ldots\times A_{i,k-n}\times B_{i,k}\times B_{i,k-1}\times\ldots\times B_{i,k-m}) \qquad \textbf{(14)}$$

$$\begin{aligned} &IF\ x(t_k)\ is\ A_{i,k}\ AND\ x(t_{k-1})\ is\ A_{i,k-1}\ldots AND\ x(t_{k-n})\ is\ A_{i,k-n}\ldots \\ &\ AND\ y(t_{k-1})\ is\ B_{i,k-1}\ldots AND\ y(t_{k-m})\ is\ B_{i,k-m} \\ &THEN\ y_i(t_k)=a_{i,0}x(t_k)+a_{i,1}x(t_{k-1})+\ldots+a_{i,n}x(t_{k-n}) \\ &+b_{i,1}y(t_{k-1})+\ldots+b_{i,m}y(t_{k-m}) \end{aligned} \qquad \textbf{(15)}$$

where $a_{i,0}, a_{i,1}, \ldots, a_{i,n}, b_{i,1}, \ldots b_{i,m}$ are the parameters of the regression equation in the *i-th* part of the $X^{n+1} \times Y^{m+1}$ domain, calculated by using e.g. the least square method and data sets.

Mamdani type fuzz y model with weights, as (13), allows to express the fuzzy and probability uncertainty, by using the frequency of fuzzy events. Takagi-Sugeno fuzzy model (15) levels the probabilistic uncertainty by using the regression method. Both models use the information granulation, made by the fuzzy partition of the consideration space.

4 General Form of the MISO Linguistic Fuzzy Model

4.1 Rule Base

A linguistic fuzzy model consists of a collection of rules, the number of rules is a combination of all linguistic values of variables in the antecedent and the consequent, in the form [10, 5]:

$$\begin{array}{l} R_i: w_i (IF\ x_1 is\ A_{1,i}\ AND\ x_2 is\ A_{2,i} \ldots AND\ x_p is\ A_{p,i} \\ THEN\ y\ is\ B_1\ (w_{1/i}) \\ _____ \\ ALSO\ y\ is\ B_j\ (w_{j/i}) \\ _____ \\ ALSO\ y\ is\ B_J\ (w_{J/i})) \end{array} \qquad (16)$$

where $i=1,\ldots,I$ – numbers of rules, $A_{1,i}, \ldots, A_{p,i}$ - linguistic values of the antecedent variables $x^T=[x_1 \ldots x_p]$, $x \in R^p$, defined by corresponding membership functions $\mu_{A_{1,i}}(x_1), \ldots \mu_{A_{p,i}}(x_p)$, $A_{1,i} \in L(X_1), \ldots, A_{p,i} \in L(X_p)$; B_j, $j=1,\ldots,J$ - linguistic values of the consequent variable $y \in Y \subset R$, defined by corresponding membership functions $\mu_{B_j}(y)$, $B_j \in L(Y)$;

w_i, $w_{j/i}$, $i=1,\ldots,I$, $j=1,\ldots,J$ - weights of rules, representing probabilities of fuzzy events in antecedents and consequents of the model, calculated according to Zadeh's definition [3], as follows (according to [6] and [11]):

w_i is a marginal probability of a joint fuzzy event in the antecedent

w_{ij} is a joint probability of the fuzzy event (fuzzy relation) in the antecedent and consequent of the elementary rule

$$w_{ij} = P(A_i \times B_j) = \sum_{(x,y) \in X \times Y} p(x,y) T\left(\mu_{A_i}(x), \mu_{B_j}(y)\right) \quad , \qquad (18)$$

$w_{j/i}$ is a conditional probability of the consequent fuzzy event

$$w_{j/i} = P(B_j / A_i) = \frac{P(A_i \times B_j)}{P(A_i)}, \tag{19}$$

and *p(x,y)* is a joint probability function, in a classical sense, which assigns to each Borel's set in $X \times Y$ a real number $p \in [0,1]$.

The weights fulfil the relations

$$\sum_{i=1,2,\ldots,I} w_i = 1 \quad , \quad \sum_{i=1,\ldots,I} \sum_{j=1,\ldots,J} w_{ij} = 1 \quad , \quad \sum_{j=1,\ldots,J} w_{j/i} = 1 \quad . \tag{20}$$

The model (16) can be also written as:

$$R_i : w_i (IF\ x_1 is A_{1,i}\ AND \ldots AND\ x_p is A_{p,i}\ THEN\ y\ is \sum_{B_j \in L(Y)} w_{j/i} / B_j) \tag{21}$$

4.2 Linguistic Inference

The linguistic fuzzy model (16) describes the fuzzy relation *R* on the linguistic space $L(X_1) \times \ldots \times L(X_p) \times L(Y)$. According to linguistic inference presented in [2] and using the notion of a probability of linguistic values of the model variables [11], the relation can be written as:

$$\begin{gathered} \forall A_{1,i} \in L(X_1), \ldots, \forall A_{p,i} \in L(X_p), \\ \forall B_j \in L(Y):\ \mu_R(A_{1,i}, \ldots, A_{p,i}, B_j) = w_{ij} \end{gathered} \tag{22}$$

The linguistic fuzzy output

$$D(y) = \left\{ \left(B_j, \mu_{D(y)}(B_j) \right),\ B_j \in L(Y) \right\},\ j = 1, \ldots, J \tag{23}$$

can be inferred by using Zadeh's compositional rule and fuzzy input defined on $L(X_1) \times \ldots \times L(X_p)$. The membership function for every linguistic value $B_j,\ j = 1, \ldots, J$ can be given by the expression:

$$\mu_{D(y)}(B_j) = \underset{i}{S} \left\{ T(T(\mu_{D(x_1)}(A_{1,i}), \ldots, \mu_{D(x_p)}(A_{p,i})), \mu_R(A_{1,i}, \ldots, A_{p,i}, B_j)) \right\} \tag{24}$$

where T, S are a t-norm and t-conorm, respectively.

5 Exemp lary Linguistic Fuzzy Model of the Stochastic Process

The set of data $\{x_{t_k}\}$ of the euro/Polish zloty exchange rate, observed daily in the first year of involving it into 12 countries of the EU, has been recognized as a

realization of a certain stochastic process. The process will be modelled to predict some linguistic value of the process and the probability of its occurrence.

We assume two variables with delays: $x_{t_{k-1}}, x_{t_{k-2}}$ as antecedent variables, so the created model will represent a fuzzy relation $R(x_{t_k}, x_{t_{k-1}}, x_{t_{k-2}})$ of the linguistic variables in a form of rules (16). Three linguistic states of the process have been distinguish: *L(X)={low, middle, high}* and the fuzzy meaning have been defined in the process domain *X*:

‘ *low value of x* ’: $A_1 = 1/a_1 + 0.5/a_2$,

‘ *middle value of x* ’: $A_2 = 0.5/a_2 + 0.5/a_3$,

‘ *high value of x* ’: $A_3 = 0.5/a_3 + 1/a_4$,

where $a_1, \ldots, a_4$ are disjoint intervals in $X = [3.4,\ 4.2] \in R$.

T he joint empirical probability distribution $p(x_{t_k}, x_{t_{k-1}}, x_{t_{k-2}})$ has been calculated, using disjoint cube intervals $a_i \times a_j \times a_k \in X^3$, i,j,k=1,…,4.

The 3D empirical probability distribution $P(x_{t_{k-1}}, x_{t_{k-1}}, x_{t_{k-2}})$ of linguistic variables, taking the fuzzy states A_1, A_2, A_3 observed at moments t_k, t_{k-1}, t_{k-2} has been computed, according to (18). Then, the marginal and conditional probability distributions of fuzzy states have been calculated, by summation and using (19), respectively.

The ling uistic fuzzy model consists of 7 file rules, because two weights representing probabilities of fuzzy events in antecedents are equal to zero:
$P((x_{t_{k-2}} \text{ is } low) \cap (x_{t_{k-1}} \text{ is } high)) = 0$,
$P((x_{t_{k-2}} \text{ is } hihg) \cap (x_{t_{k-1}} \text{ is } low)) = 0$.

The first rule is as follows:

$$R_1: 0.365 (IF\ x_{t_{k-2}} \text{ is } low\ AND\ x_{t_{k-1}} \text{ is } low \quad THEN\ y \text{ is } 0.84/low + 0.16/middle) \tag{25}$$

Table 1. presents all 7 file rules of the created model in a form of a decision table. The weights (real numbers) represent values of conditional and marginal probabilities, as in (21) and (25). It is easy to note, that $\sum_{i=1,\ldots,7} w_i = 1$ and $\sum_{j=1,2,3} w_{j/i} = 1,\ i = 1,\ldots,7$.

We show the linguistic inference algorithm taking two exemplary numeric values of the process: $x_{t_{k-2}} = 39.5,\ x_{t_{k-1}} = 40.5$ and the elaborated fuzzy model. We want to predict the linguistic fuzzy set *B* of the consequent variable x_{t_k}

$$B = \mu_B(low)/low + \mu_B(middle)/middle + \mu_B(high)/high \quad \textbf{(26)}$$

Table 1. The decision table representing the rule-based fuzzy model of the examined stochastic process

x_{t-1} / x_t	*low*	*middle*	*high*
low	$0.365\{x_t\ is\ A_1/0.84;\ x_t\ is\ A_2/0.16;\ x_t\ is\ A_3/0$	$0.06\{x_t\ is\ A_1/0.42;\ x_t\ is\ A_2/0.55;\ x_t\ is\ A_3/0.03$	0
middle	$0.07\{x_t\ is\ A_1/0.4;\ x_t\ is\ A_2/0.43;\ x_t\ is\ A_3/0.17$	$0.07\{x_t\ is\ A_1/0.10;\ x_t\ is\ A_2/0.85;\ x_t\ is\ A_3/0.05$	$0.025\{x_t\ is\ A_1/0;\ x_t\ is\ A_2/0.48;\ x_t\ is\ A_3/0.52$
high	0	$0.03\{x_t\ is\ A_1/0;\ x_t\ is\ A_2/0.30;\ x_t\ is\ A_3/0.70$	$0.380\{x_t\ is\ A_1/0;\ x_t\ is\ A_2/0.07;\ x_t\ is\ A_3/0.93$

By applying the fuzzy description notion (6) and the assumed partition of the input-output space, we find linguistic fuzzy sets of the antecedent variables, as follows:

$$D_{(39.5)}(x_{t_{k-2}}) = 0.5/middle + 0.5/high$$

$$D_{(40.5)}(x_{t_{k-1}}) = 1/high\ .$$

The membership functions for estimated linguistic fuzzy set B can be calculated according to (24), as follows:

$$\mu_B(low) = S\{T\left(T\left(\mu_{D(39.5)}(middle), \mu_{D(40.5)}(high)\right), \mu_R(middle, high, low)\right), \quad \textbf{(27)}$$
$$T\left(T\left(\mu_{D(39.5)}(high), \mu_{D(40.5)}(high)\right), \mu_R(high, high, low)\right)\}$$

The value of the expression (27) is equal to zero, because the membership values of the relations are equal to zero: $\mu_R(middle, high, low) = 0$, $\mu_R(high, high, high) = 0$ (see Table 1.) and t-norms are also equal to zero.

$$\mu_B(midle) = S\{T\left(T\left(\mu_{D(39.5)}(middle), \mu_{D(40.5)}(high)\right), \mu_{R_6}(midle, high, middle)\right), \quad (28)$$
$$T\left(T\left(\mu_{D(39.5)}(high), \mu_{D(40.5)}(high)\right), \mu_{R_7}(high, high, middle)\right)\}$$

$$\mu_B(high) = S\{T\left(T\left(\mu_{D(39.5)}(middle), \mu_{D(40.5)}(high)\right), \mu_{R_6}(middle, high, high)\right), \quad (29)$$
$$T\left(T\left(\mu_{D(39.5)}(high), \mu_{D(40.5)}(high)\right), \mu_{R_7}(high, high, high)\right)\}$$

Applying a product t-norm $T(a,b) = ab$ and a bounded sum $S(a,b) = \min(1, a+b)$ as a t-conorm to the expressions (28) and (29) we calculate the fuzzy linguistic value *B* of the consequent variable x_{t_k}, as follows:

$$\mu_B(midle) = S\{T(T(0.5, 1), 0.48), T(T(0.5, 1), 0.7)\} = \quad (30)$$
$$S(0.24, 0.035) = 0.275$$

$$\mu_B(high) = S\{T(T(0.5, 1), 0.52), T(T(0.5, 1), 0.93)\} = \quad (31)$$
$$S(0.26, 0.465) = 0.725$$

Output discrete fuzzy set, determined on the linguistic term set *L(X)={low, middle, high}*, representing the forecast of the linguistic variable *x* and the levels of affiliation to linguistic values is equal to:

$$B = 0/low + 0.275/middle + 0.725/high \quad (32)$$

We can also use one of the defuzzyfication methods to calculate the numeric value of $x^*_{t_k}$ [2]. Both, the fuzzy linguistic set (32) and a possible numeric value state the predicted values of the linguistic consequent variable. We can also use the probability of events $P((x_{t_{k-2}} isA_i) \cap (x_{t-1} isA_j))$ to compute the expected value of predicted numeric values. Details are shown in [11].

References

1. Zadeh L. A.: *The Concept of a Linguistic Variable and its Application to Approximate Reasoning,* Information Sciences, Part I: 8 (1975) 199-249.
2. Nakoula S., Galichet S., Foulloy L.: *Identification of Linguistic Fuzzy Models Based on Learning,* In: Hellendoorn H., Driankov D. (eds): Fuzzy Model Identification; Selected Approaches. Springer-Verlag, Berlin Heidelberg (1997) 281-319.
3. Zadeh L. A.: *Probability Measures of Fuzzy Events,* Journal of Mathematical Analysis and Applications, Vol. 23, No 2 (1968) 421-427.
4. Pedrycz W.: *An Identification Algorithm in Fuzzy Relational Systems,* Fuzzy Sets and Systems, Vol. 13, (1984) 153-167.

5. Rutkowski L.: *Methods and Techniques of Artificial Intelligence* (in Polish). PWN, Warsaw (2005).
6. Walaszek-Babiszewska A.: *IF-THEN Linguistic Fuzzy Model of a Discrete Stochastic System,* In: Cader A., Rutkowski L., Tadeusiewicz R., Żurada J. (eds): Book series: Challenging Problems of Sciences; Computer Sciences. EXIT, Warsaw (2006) 169-174.
7. Yager R. R., Filev D. P.: *Template Based Fuzzy Systems Modelling. Journal of Intelligent Systems,* Vol. 2 (1994) 39-54.
8. Kacprzyk J.: *Fuzzy Sets in System Analysis* (in Polish). PWN, Warsaw (1986).
9. Box G. E. P., Jenkins G. M.: Time Series Analysis; Forecasting and Control (in Polish). PWN, Warsaw (1983).
10. Yager R. R., Filev D. P.: *Essentials of Fuzzy Modeling and Control* (in Polish). WNT and Wiley, Warsaw (1995).
11. Walaszek-Babiszewska A.: *Construction of Fuzzy Models Using Probability Measures of Fuzzy Events,* In: Kaszyński R. (ed): Proc. 13 th IEEE/IFAC International Conf. 'Methods and Models in Automation and Robotics', CD, Szczecin, Poland (2007) 661-666.

Proceedings of the International Multiconference on
Computer Science and Information Technology pp. 151–160
ISSN 1896-7094
© 2007 PIPS

Implementation and evaluation of the fuzzy system for the assessment of cadastre operators' work

Tomasz Karczyński and Bogdan Trawiński,

Wrocław University of Technology, Institute of Applied Informatics,
Wybrzeże S. Wyspiańskiego 27, 50-370 Wrocław, Poland
bogdan.trawinski@pwr.wroc.pl, tomkarczynski@googlemail.com

Abstract. The Mamdani fuzzy system for the multi-criteria assessment of cadastral system operators' work is presented in the paper. The system comprises five input criteria: productivity, complexity, quality, availability and report production and was implemented using JAVA. Moreover two alternative methods of assessment were incorporated into the system: first based on self-organizing maps and second employing linear function of input criteria. All three methods of operator assessment were evaluated using actual data taken from one cadastre information centre for the period of 70 weeks.

1 Introduction

The methods of computational intelligence are often applied to solve multi-criteria non-linear problems [1–4, 7]. The Mamdani fuzzy model for the multi-criteria assessment of cadastral system operators' work is presented in the paper. Cadastre systems belong to state resources and are designed for the registration of parcels, buildings and apartments as well as their owners and users. Those systems have complex data structures and sophisticated procedures of data processing. There are above 400 information centres located in district local self-governments as well as in the municipalities of bigger towns in Poland. One of critical tasks of cadastre system maintaining is the input of changes into its database. The changes are allowed only on the basis of legal documents such as notarial deeds, extracts from perpetual books, results of surveyors' works, administrative decisions and others. Managers of information centres often complain they have no adequate tool for the assessment of work of cadastre system operators. The first fuzzy models for the assessment of information system operators' work comprising three criteria: productivity, complexity and quality were proposed and evaluated in [5, 6]. The extension of the model of two more criteria i.e. operators' availability and report production as well as the implementation of the assessment system is presented in the paper.

2 Fuzzy Model for the Assessment of Operators' Work

Fuzzy model for the assessment of operators' work is a multi-criteria one that comprises five input variables which values are automatically calculated using operators' activity data recorded in the cadastral database. The input variables are: productivity (further denoted by P) expressed by the number of changes input into the cadastre database by an operator within a given period, complexity of changes (C) specified as the mean number of objects which were modified in the database within a given period, quality of work (Q) defined as the percentage of changes made without any corrections, report production (R) determined by the number of reports generated by an operator within a given period and the availability (A) qualified as the percentage of work time spent on fulfilling tasks using cadastral system.

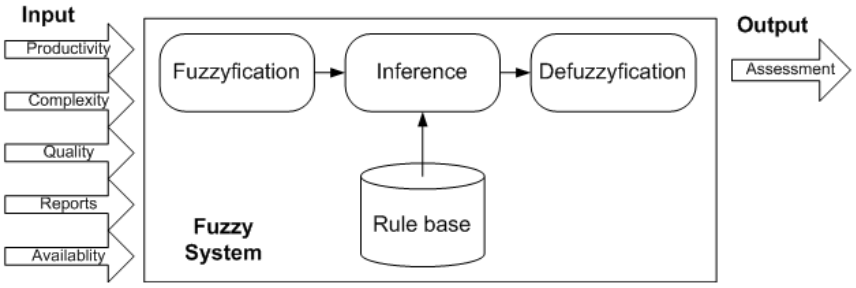


Fig. 1. Classic model of a fuzzy system

The classic model of a fuzzy system (FS) is given in Figure 1. The crisp values of input variables are fuzzyfied, the inference process is performed using rule base and next the fuzzy result of aggregation is defuzzyfied making final crisp assessment value. However in the case of five input criteria no expert is able to create a reliable rule base, so that before implementation the fuzzy model had to be restructured. Having rule bases of selected couples of input variables expressed in the form of two dimensional matrices, eventually seven FSs were constructed and put together in the structure of a tree (see Figure 2). Input of the system is directed to four FSs at level 0 and their output crisp values constitute the inputs to next two FSs at level 1, and in the end the results of the latter are used as the inputs of the terminal FS at level 2. Final assessments depend on all five input variables. The input variables were coupled by an expert who took into account some logical relations between them. For example P and C are related because the more changes completed the more database object modified. However some operators, those more experienced, are able to perform more complex changes whereas others confine to simpler changes. In turn the more complex changes and the bigger number of database updates the higher probability of errors. However some operators are more careful and others less.

And sometimes the operators are not assigned their main tasks but they are asked to generate various kinds of system reports. But both input of changes and report production are good indicators of operators' availability.

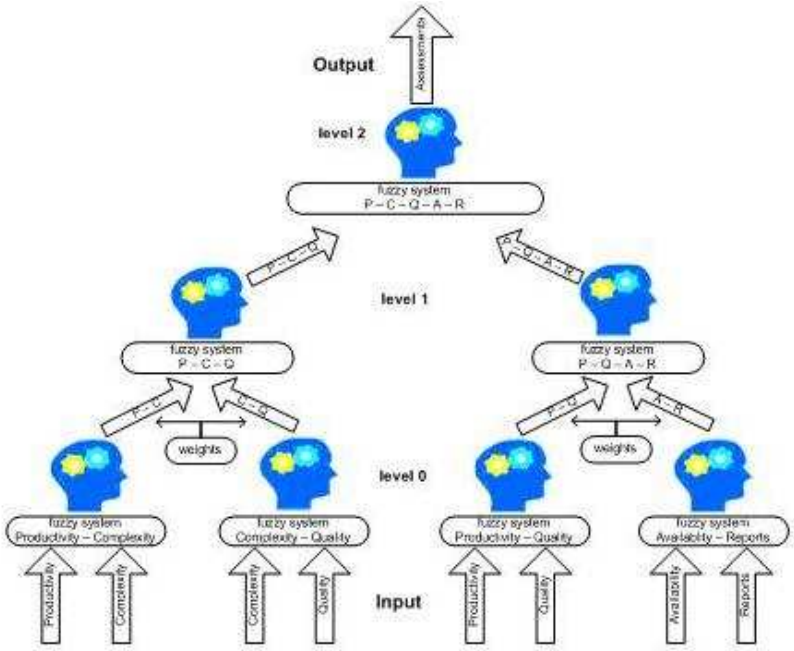


Fig. 2. Tree structure of fuzzy systems.

Mamdani models with fuzzy sets expressed by means of triangular and trapezoidal functions were used to construct fuzzy systems at respective levels. Three different models of input and output variables were used, they were named 7x9, 9x11 and 11x11. In the 7x9 model 7 fuzzy sets determine each input criteria and the output is defined by 9 fuzzy sets. In the 9x11 and 11x11 models there are 9 and 11 fuzzy sets for each input as well as 11 for each output respectively. Fuzzy membership functions used to define P, C, Q, A and R input variables and an output are very similar to those described in [5, 6]. The statistics subsystem provides initial parameters of the model and values of input criteria. The idea of obtaining the final assessment consists in calculating the average value of P, C, Q, A and R criteria taking into account the activity records in a cadastre database for all operators and for long period of time, e.g. a year or a half of year. These average values are used as the middle values of the universe of disclosure for fuzzy sets describing individual input variables. Standard deviations for each criterion are calculated for a long period of time too and they are used for the determination of the width of the basements of triangular and trape-

zoidal fuzzy sets. The domain for output is an arbitrary assessment scale from 0 to 200, with 0 being the lowest and 200 the highest mark. The initial reference average values and the widths of triangle or trapezoid basements are treated as customization parameters and can be modified by managers during exploitation. The rule base for each model was proposed by an expert and contains simple IF-THEN rules where the condition consists of only two input variables combined by AND operator and the conclusion is built by one variable. An example of a rule is as follows: ***IF Complexity is low AND Productivity is medium THEN Assessment is very low.*** Rule bases for FSs at successive levels are shown in tables 1, 2, and 3, where UL, EL, VL, L, BM, M, AM, H, VH, EH, UH denote Ultra Low, Extremely Low, Very Low, Low, Below Medium, Medium, Above Medium, High, Very High, Extremely High and Ultra High respectively. There is following relationship between the meaning of linguistic values UL $\prec$ EL $\prec$ VL $\prec$ L $\prec$ AL $\prec$ BM $\prec$ M $\prec$ AM $\prec$ BH $\prec$ H $\prec$ VH $\prec$ EH $\prec$ UH. In order to be able to determine the importance of a given pair of input criteria it is possible to assign a weight to a rule base at level 1.

Table 1. Representation of rule base in matrix form for level 0 (7x9 model)

	VL	**L**	**BM**	**M**	**AM**	**H**	**VH**
VL	EL	EL	VL	L	L	BM	M
L	EL	VL	L	L	BM	M	AM
BM	VL	L	L	BM	M	AM	H
M	L	L	BM	M	AM	H	H
AM	L	BM	M	AM	H	H	VH
H	BM	M	AM	H	H	VH	EH
VH	M	AM	H	H	VH	EH	EH

Table 2. Representation of rule base in matrix form for level 1 (9x11 model)

	EL	**VL**	**L**	**BM**	**M**	**AM**	**H**	**VH**	**EH**
EL	UL	EL	VL	VL	L	L	L	BM	BM
VL	EL	EL	VL	L	L	BM	BM	M	M
L	VL	VL	L	L	BM	BM	M	AM	AM
BM	VL	L	L	BM	BM	M	AM	H	H
M	L	L	BM	BM	M	AM	AM	H	H
AM	L	BM	BM	M	AM	AM	H	H	VH
H	L	BM	M	AM	AM	H	H	VH	VH
VH	BM	M	AM	AM	H	H	VH	VH	EH
EH	BM	M	AM	H	H	VH	VH	EH	UH

Table 3. Representation of rule base in matrix form for level 2 (11x11 model)

	UL	**EL**	**VL**	**L**	**BM**	**M**	**AM**	**H**	**VH**	**EH**	**UH**
UL	UL	UL	EL	VL	VL	L	L	AL	AL	BM	BM
EL	UL	EL	VL	VL	L	L	AL	AL	BM	M	M
VL	EL	VL	VL	L	L	AL	BM	BM	M	AM	AM
L	VL	VL	L	L	AL	BM	BM	M	AM	AM	BH
BM	VL	L	L	AL	MB	BM	M	AM	AM	BH	BH
M	L	L	AL	BM	MB	M	AM	AM	BH	BH	H
AM	L	AL	AL	BM	M	AM	AM	BH	BH	H	H
H	AL	AL	BM	M	AM	AM	BH	BH	H	VH	VH
VH	AL	MB	M	AM	AM	BH	BH	H	VH	VH	EH
EH	BM	M	AM	BH	BH	BH	H	VH	VH	EH	UH
UH	BM	M	AM	BH	BH	H	H	VH	EH	UH	UH

3 Alternatives to the Fuzzy Model

Two alternatives to the fuzzy model were implemented in the system for operators' work assessment: a mechanism based on self-organizing map (SOM) and one employing a linear function of input criteria. The SOM is a type of unsupervised artificial neural network, as only one neuron fires during learning it is called a winner-take-all-network [8]. The architecture of SOM implemented is shown in Fig. 3 where all neurons are arranged in a circular way. The lateral feedback function with the profile of a Mexican hat was employed. The network was trained using the activity data for 10 operators for the period of 70 weeks with 100 neurons and 3000 train iterations.

Second alternative implemented to calculate the assessment was the linear function of input criteria expressed by means of the following equation

$$Ass(x) = 100 + (w_p * \frac{P_x - P_n}{P_n} + w_c * \frac{C_x - C_n}{C_n} + w_q * \frac{Q_x - Q_n}{Q_n} + w_a * \frac{A_x - A_n}{A_n} + w_r * \frac{R_x - R_n}{R_n}) * 100$$

where P_x, C_x, Q_x, A_x, R_x are activity data for user X determined for a given week, and P_n, C_n, Q_n, A_n, R_n are average values of productivity, complexity, quality, availability and report production calculated for all users for a long period. Due to the lack of actual assessments it was not possible to determine coefficients by individual variables using multiple regression. So that criterion importance weights W_p, W_c, W_q, W_a, and W_r were used and those weights could be set by centre managers.

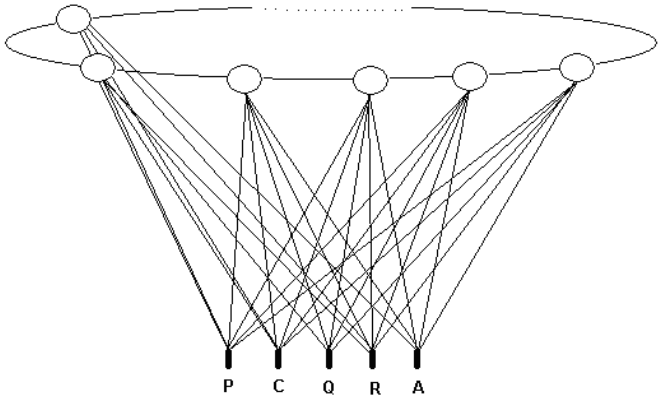


Fig. 3. The architecture of self-organizing map implemented

4 Implementation of the System

The system for the assessment of operators' work was developed for Windows environment using JAVA programming language and SQL Server 2005 database management system. It is composed of following modules: Data gain, Statistics, Assessment, Visualization, Customization and Help subsystems (see Figure. 4).

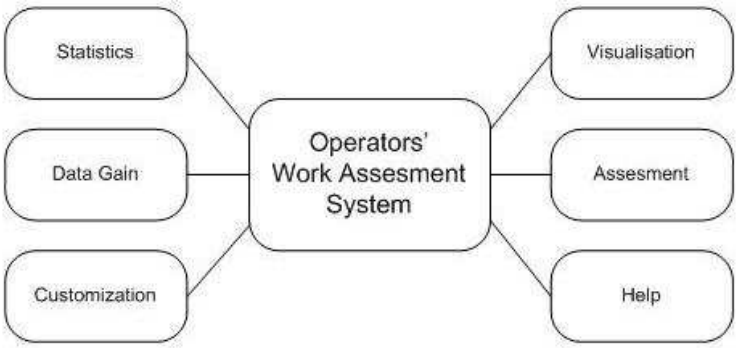


Fig. 4. Subsystems of the system for operators' work assessment.

The task of data gain subsystem (see Figure 5) is to scan periodically, eg. once a day or once a week, the database of the cadastral system, draw operators' activity data for successive days, process them and save into a proprietary database of the subsystem. In order to calculate the input values for FSs a period of day and night was divided into 48 time quanta each 30 minutes long

and the data gain subsystem, calculates the frequencies within five dimensions: productivity, complexity, quality, availability and report production and assigns them to corresponding time quanta. Such distribution of activity data assures the assessment system to be flexible and easily parameterized and customized. Especially the manager can determine time scope i.e. what period is an unit of the assessment: a week or a month, what time of a day should be taken into account: either only working hours, or also overtime in the afternoon or early coming to work or night work. There is a possibility to define what days are interesting for monitoring users activity: either only working days or also Saturdays and holidays.

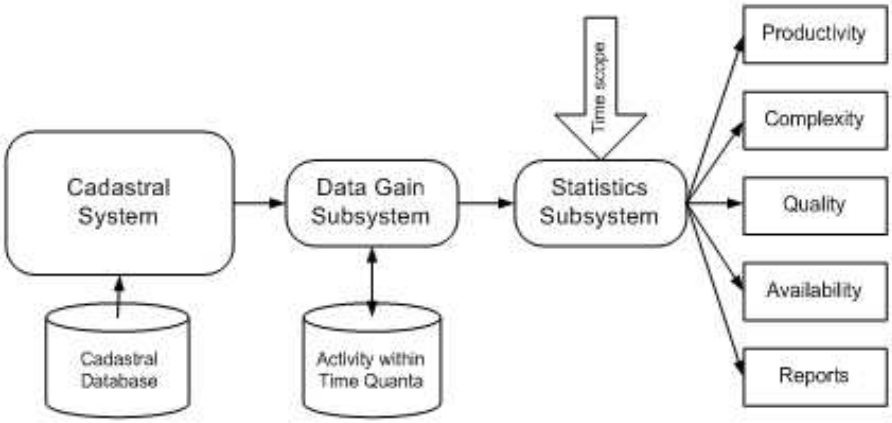


Fig. 5. Preparation of input values for FSs.

The main goal of the Customization subsystem is to provide tools to define work parameters of each operator and whole information centre as well as to set the parameters of the fuzzy models and the SOM neural network. It is possible to define

for each operator his weekly work schedule, leaves, sickness absences etc. As for the centre you can determine working hours, free days, and holidays. Finally the shapes of fuzzy sets, inference and defuzzyfication method, rule importance weights can be set for FSs and the number of neurons, variation range of neurons, number of iterations in the training phase, etc.

5 Evaluation of the System

The evaluation of the system was carried out using cadastre database taken from one information centre and the activity records for 10 operators working with the system during the period of 71 weeks from September 2005 till January 2007 were analyzed. In Fig 7 final assessments produced by fuzzy mechanism and

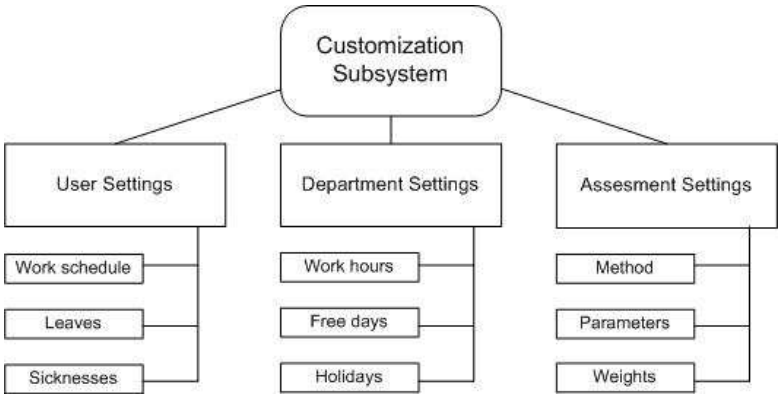


Fig. 6. Preparation of input values for FSs.

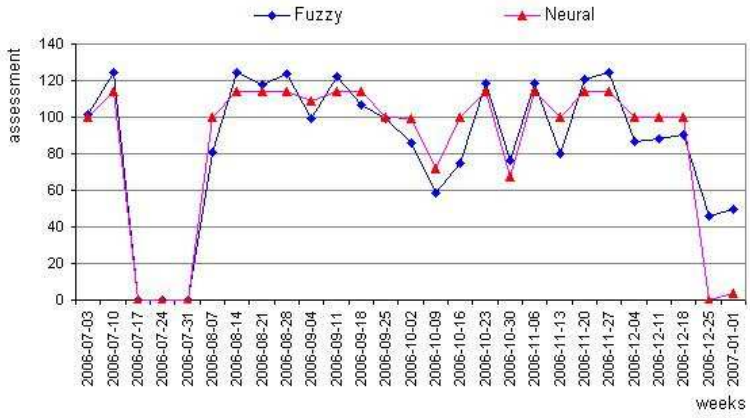


Fig. 7. Comparison of assessments produced by fuzzy set and SOM mechanisms for one user

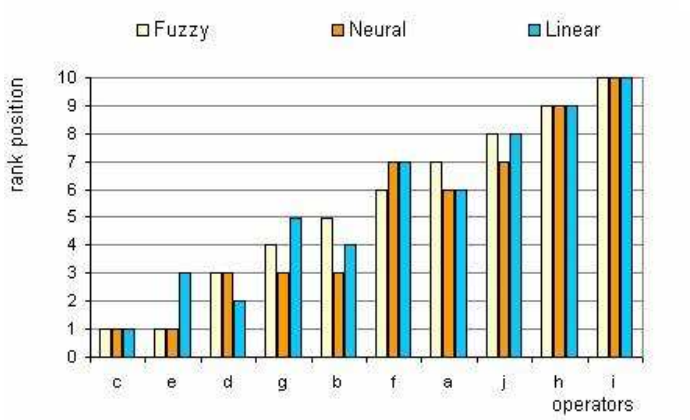


Fig. 8. Comparison of operator rankings made by fuzzy, neural and polynomial methods

self-organizing map for the user a for the period of half year are compared. The results produced by both methods are similar and they describe the variation of operators' work well. The summer leave period in August and holidays in December where no or minor activity was detected can be easily noticed in the chart.

Rank positions of individual operators produced by the system using fuzzy, neural and polynomial methods are presented in Figure. 8. It can be seen that all three methods equally recognized the best and the worst operators. This is similar to the results presented in [5] where the manager was able to distinguish easily only utmost operators and had substantial difficulties to assign different assessment to average ones.

6 Conclusions and Future Works

The model for the multi-criteria assessment of cadastral operators' work has been designed and programmed with the aid of an expert and information centre managers. The evaluation of the model has proved that it is reasonable to use operators' work statistics as the initial parameters and the input values for the model. The model basing on operators' activity records saved in cadastral database produces the assessments for defined periods of time automatically. Two additional assessment methods were implemented in the system, namely one based on self-organizing map and second employing linear function of input variables. The architecture of proposed model enables the centre managers to modify system parameters to customize them to local circumstances. It is planned to carry out further evaluation experiments with the active participation of the centre managers. They will be instructed how the fuzzy system operates and will be got familiar with the statistics of operators' work within a

given time. Moreover they will be able to determine the importance weights to successive criteria in order to adjust the system to their preferences. Subjective opinions of the usefulness of the fuzzy system will also be gathered.

References

1. Ajith A.: *Neuro-Fuzzy Systems: Sate-of-the-Art Modelling Techniques,* Proceedings of the 6th International Work-Conference on Artificial and Natural Neural Networks, pp. 269–276, (2001).
2. Bonissone, P. P., Yu-To Chen, Goebel K., Khedkar P. S.: *Hybrid soft computing systems: industrial and commercial applications,* Proceedings of the IEEE,vol. 87, no. 9, pp. 1641–1667 (1999).
3. Cordón O., Gomide F., Herrera F., Hoffmann F., Magdalena L.: *Ten years of genetic fuzzy systems: current framework and new trends,* Fuzzy Sets and Systems 141 5–31 (2004).
4. Herrera F.: *Genetic Fuzzy Systems: Status, Critical Considerations and Future Directions,* International Journal of Computational Intelligence Research 1(1) pp. 59–67 (2005).
5. Król D., Kukla G., Lasota T., Trawiński B.: *Investigation of the fuzzy system for the assessment of cadastre operators' work,* In: Advances in Web intelligence and data minning. M. Last et al. (eds.) Studies in Computational Intelligence, pp. 131–140. Springer, Berlin (2006).
6. Król D., Kukla G., Lasota T., Trawiński B.: *Fuzzy model for the assessment of operator's work in a cadastre information system,* LNAI, vol. 4253, pp. 774–781. Springer, Heidelberg (2006).
7. Rutkowski L.: *Methods and Techniques of Artificial Intelligence,* Computational Intelligence (in Polish: Metody i techniki sztucznej inteligencji. Inteligencja obliczeniowa). PWN Warszawa (2005).
8. Skapura D. M.: *Building Neural Networks,* Addison Wesley (1996).

Proceedings of the International Multiconference on
Computer Science and Information Technology, pp. 161 – 170
ISSN 1896-7094
© 2007 PIPS

Fuzzy contrast enhancement for images in the compressed domain

Camelia Popa, Aurel Vlaicu, Mihaela Gordan and Bogdan Orza

Centre for Multimedia Technologies and Distance Education,
Technical University of Cluj-Napoca,
Daicoviciu 15, Cluj-Napoca, Romania
{Camelia.Popa, Aurel.Vlaicu, Bogdan.Orza}@com.utcluj.ro,
Mihaela.Gordan@bel.utcluj.ro
http://www.ctmed.utcluj.ro

Abstract. Our objective is to investigate image processing directly in the compressed domain, without full decompression. Compressed domain image processing algorithms provide a powerful computational alternative to classical (pixel level) based implementations. The field is just emerging and the algorithms reported in the literature are mostly based on linear arithmetic operations between pixels. In this paper, the problem of implementing a non-linear operator, using compressed domain processing is addressed. A new algorithm for digital image enhancement, using fuzzy theory, adapted to the frequency content of each coefficient block in the DCT (Discrete Cosine Transform) encoded JPEG image is developed and proposed.

1 Introduction

The image storage in bitmap format is not so used in day-to-day applications; even medical images are stored in JPEG format. This is because of the advantages offered by the JPEG format: little storage space needed and a better capacity to send information (via internet). For images stored in JPEG format, it is recommended to process them directly in the compressed domain, in order to reduce the time needed to process data. This time economy is due to the fact that it is no longer necessary to decompress the image, to process it at pixel level and to recompress it back and that processing images in the compressed domain means that there are fewer data to process.

Linear processing is easier to implement in the compressed domain because there are no problems to process an image involving linear operations, such as the adding with a constant, multiplying with a constant, adding of two images, progressive cross fade of two images. The implementation of operations involving nonlinear operators is challenging, but not impossible, and when the right combination is found, the processing if much faster due to the reduced number of coefficients (instead of 64 in pixel level processing, corresponding to a 8x8 block of pixels, in the compressed domain just 20% of the pixels are processed).

Implementations of image and video sequence processing in the compressed JPEG/MPEG domain are already presented in the literature. Tang [5] – defined an algorithm based on the contrast measure as the ratio of high-frequency content and low-frequency content in the bands of the DCT matrix. Kebin [6] – use the Tang

algorithm but he enhanced each block according to its block classification: smooth block or high activity block. Kim and Peli [9] – developed an image enhancement algorithm, MPEG based, for people with low-vision – contrast enhancement is performed by modifying the quantization matrices for Inter and Intra frames. Sangkeun Lee [4] – use a basic concept of the Retinex theory for enhance the images.

In the paper, a processing algorithm in the compressed domain is presented. The enhancement of images uses fuzzy techniques, involving nonlinear and thresholding operations. The techniques for image enhancement in the compressed domain are based on modifying the frequency transform of an image.

2 Image enhancement using fuzzy algorithm

Image enhancement involves processing an image in order to make it satisfactory to viewers. Our goal is to provide a fast and accurate alternative implementation in the compressed domain, to the classical fuzzy image enhancement algorithm, based on global threshold, which is usually performed at pixel level. Such an algorithm can significantly increase the computational efficiency in image processing algorithms applied to a JPEG image, avoiding compression and decompression prior and after the processing.

Further we will present the basic steps used to compress/decompress the JPEG images [1, 12]. The image is first divided into 8x8 blocks, and each 8x8 block is computed. After that, a DCT is applied on each block and results the DCT coefficients which are quantized. Many small coefficients, usually high frequency, are quantized to zero. The next step is zig-zag scan of the DCT matrix followed by RLE (Run Length Encoding) coding and entropy coding (Huffman coding). In the decoder, the compressed image is decoded and then dequantized and inverse-DCT-transformed.

There are two ways to enhance the images compressed with JPEG: the compressed domain processing (no decompression/compression) and the pixel level processing (enhance the image after decompression):

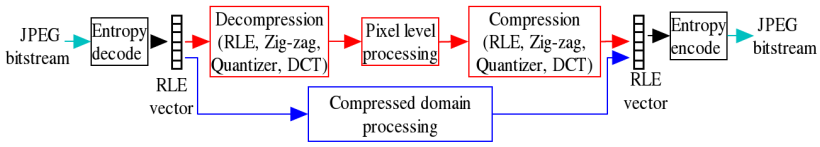


Fig. 1. The two ways to process the image compressed JPEG

For 8x8 blocks of pixels, instead of 64 in the pixel level processing, in the compressed domain only a small quantity of the pixels are processed, because the majority of the coefficients in the DCT domain are zero after quantization. In our algorithm we use an adaptive minimal decompression: full decompression is no longer needed, but decompression is used for the block having many details, for an improved accuracy of processing.

2.1 Contrast enhancement using an INT-Operator from fuzzy theory

In recent years, many researchers have applied the fuzzy set theory [2] to develop new techniques for contrast improvement. A basic fuzzy algorithm for image enhancement, using a global threshold, will be briefly recalled. Let us consider a gray level digital image, represented by the gray level values of the pixels scaled to the range [0;1]. Let l be any gray level of a pixel in this digital image, $l \quad [0;1]$.

Contrast improvement is a basic point processing operation aiming mainly to maximize (increase) the dynamic range of the image. A higher contrast in an image can be achieved by darkening the gray level in the lower luminance range (typically under 0.5 on a [0;1] scale) and brightening the ones in the upper luminance range [13]. This processing usually implies the use of a non-linear function; a typical form of such a function could be the one presented in Figure 2. A possible mathematical expression of such a nonlinear function, $Int : [0;1] \quad [0;1]$, is as follows:

$$Int(l) = \begin{cases} 2 \cdot l^2, \ if \quad 0 \le l \le 0.5 \\ 1 - 2 \cdot (1-l)^2, \ if \quad 0.5 \le l \le 1 \end{cases}, \quad l \in [0;1]. \tag{1}$$

The expression in (1) represents a well known operator in the fuzzy sets theory, namely the intensification (INT) operator; when applied on digital images, it has the effect of contrast enhancement.

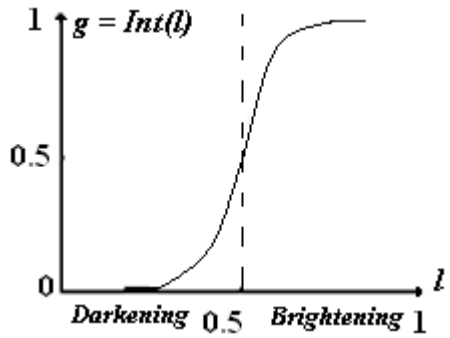


Fig. 2. Fuzzy intensification

Let us denote the resulting gray levels in the contrast enhanced image by g given by:

$$g = Int(l), \quad \forall \quad l \in [0;1]. \tag{2}$$

Thus, the contrast enhanced image will have gray levels of its pixels given by the nonlinear point-wise transformation in figure 2, applied to the original gray level image.

2.2 Convolution of the image in the compressed domain

In order to implement a fuzzy algorithm in the compressed domain, linear operations and nonlinear operations are necessary. For example, the square of an image is needed, which is a nonlinear operation and can be obtained using the multiplication algorithm of two images, in the compressed domain, developed by B. Smith [1].

The following notation will be introduced: considering the original HxW gray level image, divided into 8x8 pixels blocks (as in the typical process of JPEG encoding), we denote any such block of pixels by the matrix $U[8x8]$, containing the gray level values of the pixels scaled to the range [0;1]:

$$U = \{u(i,j) \in [0,1],\ where\ i = 0,\ 1,...,\ 7,\ \ j = 0,\ 1,...,\ 7\} \tag{3}$$

Consider $U_{dct}[8x8]$ to be the matrix of the DCT coefficients of any 8x8 pixels block from the original image U (directly available in the JPEG image coding, as shown in figure 1). In order to apply the contrast enhancement algorithm described above, we need to compute the square of each pixel luminance in the 8x8 block, in the compressed domain. That is, we should obtain the DCT of the 8x8 block of squared luminance. We denote this DCT matrix by $U_{dct,sq}[8x8]$. According to [1], $U_{dct,sq}[8x8]$ can be obtaining from $U_{dct}[8x8]$ as follows (using the notations from [1]):

$$U_{dct,sq}(x_1,x_2) = \frac{1}{4\cdot Q(x_1,x_2)} \sum_i \sum_j C(i,x_1)\cdot C(j,x_2)\cdot [u(i,j)]^2 =$$

$$= \sum_{y_1,y_2,w_1,w_2} U_{dct}(y_1,y_2)\cdot U_{dct}(w_1,w_2)\cdot W_Q(y_1,y_2,w_1,w_2,x_1,x_2)$$

$$where:$$

$$W_Q(y_1,y_2,w_1,w_2,x_1,x_2) = \frac{Q(y_1,y_2)\cdot Q(w_1,w_2)}{256\cdot 64\cdot Q(x_1,x_2)} W(x_1,y_1,w_1)\cdot W(x_2,y_2,w_2) \tag{4}$$

$$with:\quad W(x,y,w) = \sum_i C(i,x)\cdot C(i,y)\cdot C(i,w)$$

$$C(i,x) = A(x)\cos\frac{(2\cdot i+1)\cdot x\cdot \pi}{16};\quad A(x) = \begin{cases} \frac{1}{\sqrt{2}}, for\ x = 0 \\ 1, for\ x \neq 0 \end{cases}$$

We can efficiently compute this sum by noticing several facts: firstly, that for compressed images, U_{dct} is zero for most values and, secondly, that in the $W_Q(y_1,y_2,w_1,w_2,x_1,x_2)$ function, only about 4% of the terms are non-zero.

2.3 Fuzzy image enhancement in the compressed domain

We should take into account that, in the compressed domain, prior to applying the DCT based compression of the JPEG algorithm, the luminance values are scaled from

the [0, 255] range to the [-128, 127] range, symmetrical towards 0. Consequently, the resulting DC coefficient will also be scaled towards 0 and the threshold for the fuzzy algorithm in the compressed domain should have the value 0 (instead of 0.5, in the case of the fuzzy algorithm applied on the pixel level).

The algorithm is applied on the DCT coefficients and, in this case only, the DC coefficient will be compared with the threshold, for each and every 8x8 block. The proposed algorithm in order to compute the luminance values in the enhanced image $IntU_{dct}$ is given by the following formula:

$$IntU_{dct}(i,j)=\begin{cases}U_{dct,sq}(i,j)+2\cdot U_{dct}(i,j)-128, & if \quad U_{dct}(0,0)\le 0\\ 2\cdot U_{dct}(i,j)-U_{dct,sq}(i,j)-128, & if \quad U_{dct}(0,0)>0\end{cases} \tag{5}$$

notice that : $U_{dct}(0,0)$ is the DC coefficient coresponding of the 8x8 pixels block of the DCT matrix

where : $i=0,1,...,7;\ j=0,1,...,7;$

In case the processing is done in the compressed domain, the threshold is done once for each 8x8 block, meanwhile in pixel level processing, it is performed separately for each pixel inside the block (64 operations).

Although, the use of the threshold only once for an 8x8 block can lead to an eronate classification, because applying the same formula for each pixel from an 8x8 block could be wrong. This is the case of the blocks with a highly non-uniform luminance distribution, which is with large AC energy content. This disadvantage can be avoided by total decompression of all the blocks containing higher energy factor, and by processing them at the pixel level. In this way, processing in the compressed domain is much more efficient, because the number of blocks needing total decompression is small, as it can be seen in the experimental part (table 1).

The fuzzy image enhancement algorithm is defined for each 8×8 pixels block of the JPEG image, as follows:

1) The average amount of the AC coefficients energy from DCT block, denoted E_{AC}, is computed as:

$$E_{AC}=\frac{\sum_{i,j}^{7}\left|U_{dct}(i,j)\right|^2}{64} \tag{6}$$

2) If $E_{AC}<e_{th}$ (where the e_{th} value represents the selection threshold between the uniform block and the blocks with a significant number of details) then the block can be considered to have uniform luminance and it will be processed in the compressed domain.

3) If $E_{AC}>e_{th}$, then the block has a significant content of details and it will be decompressed, every pixel from the block being processed separately, using classical enhancement techniques.

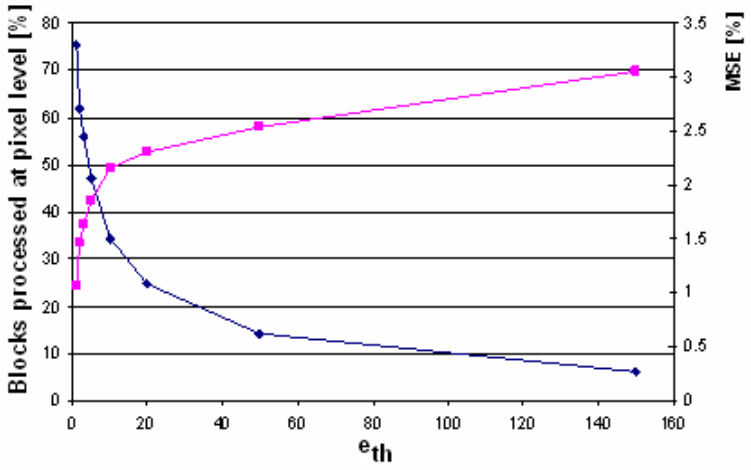


Fig.3. e_{th} influence for an image (e.g. Cars.jpg)

The e_{th} value is chosen from the experiments, taking into account the image type. A very small e_{th} value will always lead to process images of a very good quality, but the number of processed blocks in the compressed domain will be quite small, so the complexity of the computing algorithm is not significantly smaller, compared to the direct processing on the pixel. An appropriate e_{th} value will lead to the increase of the number of blocks processed in the compressed domain and, in this way we can obtain a fast fuzzy algorithm for image enhancement.

3 Experimental results

The software implementation was done using the C++ programming language. A set of different images, with different dimensions, content, details content, contrast and medium luminance has been chosen for implementation. The performance of the algorithm has been examined, relatively to:

1) the quality of the image resulted from the fuzzy enhancement process, compared to the fuzzy enhancement in the spatial domain (on pixel level);
2) the estimated efficiency of the processing method with 8x8 blocks directly in the compressed domain, the numbers of blocks processed at pixel level;

Results of enhanced images using the proposed algorithm are presented in the next figures:

a.  b. 

Fig. 4. a. Original image Medical.jpg; b. Enhanced image using the proposed algorithm

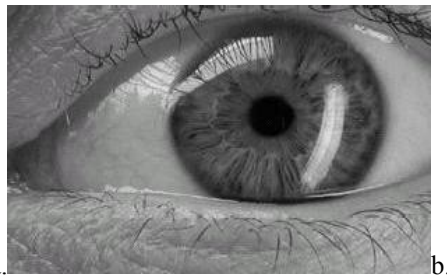
a.  b. 

Fig.5. a. Original image Eye.jpg; b. Enhanced image using the proposed algorithm

a.  b. 

Fig.6. a. Original image Lisa.jpg; b. Enhanced image using the proposed algorithm

a.  b. 

Fig.7. a. Original image Cars.jpg; b. Enhanced image using the proposed algorithm

Our goal is to obtain the same performance as with the standard pixel-level enhancement, but with a significant reduction of the computational complexity. We use as quality performance measure the MSE (Mean Squared Error), between the pixel level process images and the images processed in the compressed domain with our algorithm.

Table 1 . Results for different values e_{th}

Image name	e_{th}	MSE [%]	**Number of blocks processed at pixel level**	**Total number of blocks 8x8**	**Blocks processed at pixel level [%]**
Mouse.jpg	8	9.803	46	1000	4.6
Lisa.jpg	3	5.38	175	1000	17.50
Cars.jpg	10	2.55	61	1200	5.08
Lena.jpg	5	5.716	633	4096	15.45
Nature.jpg	10	1.65	31	1728	1.79
Eye.jpg	10	5.98	103	912	11.29
Medical.jpg	10	9.53	89	1768	5.03

In figure 8 we compare our image enhancement algorithm with other two algorithms: the global histogram equalization and the contrast enhancement by Tang [5]:

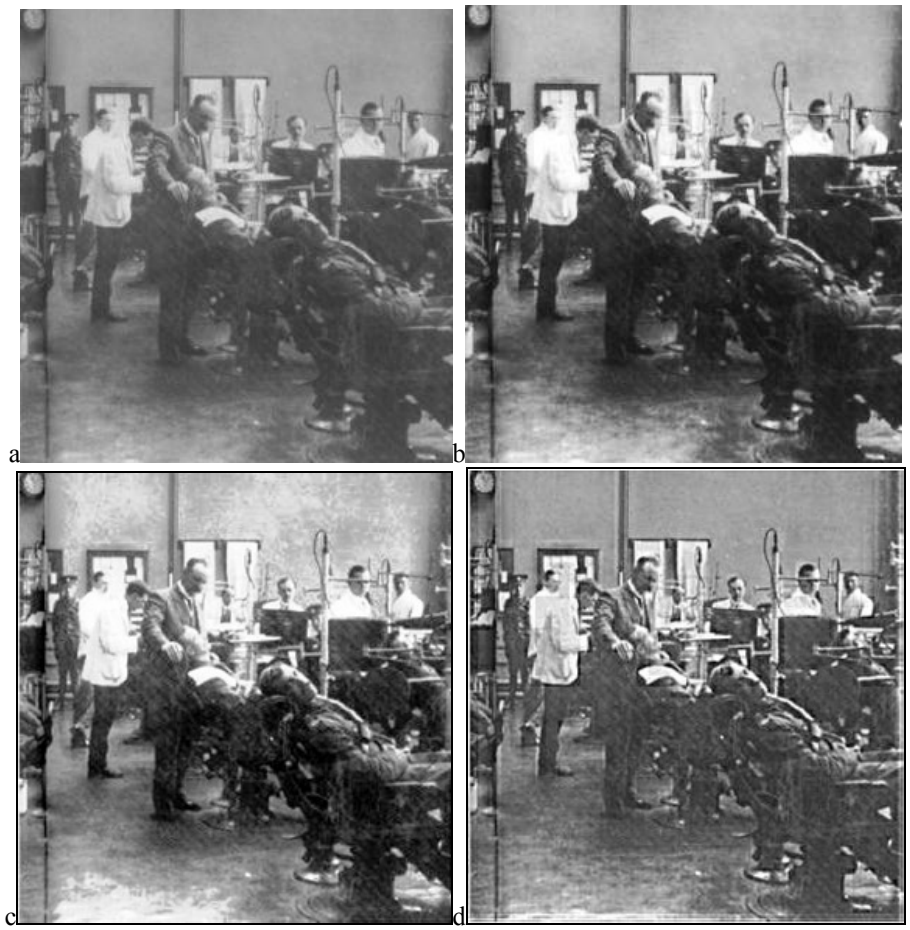


Fig.8. a. Original image; b. enhanced image using the proposed algorithm; c. enhanced image using global histogram equalization; d. enhanced image using contrast enhancement

4 Conclusion

A new algorithm has been proposed and software implemented, in the class of digital image processing in the compressed domain, dedicated to image enhancement using fuzzy theory. The algorithm was tested on different images having different statistics, with different contrast classes and medium luminance. The experimental performance results show a better computation efficiency, compared to the standard processing method (which needs a total decompression of the image), in the same conditions of preservation for the quality of the image.

The algorithm can be optimized with an automatic selection method of the threshold, in order to classify the blocks according to their content of AC, e_{th} energy, so that the best compromise between processing speed and quality of the resulted image is obtained. This issue is currently under investigation, as well as the application of this algorithm to other compressed domain processing of digital images (e.g. segmentation).

References

1. Brian Smith, Lawrence A. Rowe, *A New Family of Algorithms for Manipulating Compressed Images ,* IEEE Computer Graphics and Applications, Sept. 1993, vol.13, (no.5), pp: 34-42.
2. Pal S. K., King R. A., *Image Enhancement Using Smoothing with Fuzzy Sets,* IEEE Trans. Systems Man Cybernet, Vol. SMC-11, No. 7, 1981, pp. 494-501.
3. G. A. Triantafyllidis, M. Varnuska, D. Sampson, D. Tzovaras, M. G. Strintzis, *An efficient algorithm for the enhancement of JPEG-coded images,* Computers and Graphics, Volume 27(4), August 2003, pp. 529-534; Publisher: Elsevier
4. J. Tang, *A contrast based image fusion technique in the DCT domain,* Digital Signal Processing, 2004, Volume 14(3), pp. 218-226.
5. Lee Sangkeun, *Content-based image enhancement in the compressed domain based on multi-scale alpha-rooting algorithm,* PRL (27), No. 10, 15 July 2006, pp. 1054-1066.
6. J. Tang, E. Peli, S. Actonm *Image enhancement using a contrast measure in the compressed domain,* - Signal Processing Letters, IEEE, Vol. 10, Issue: 10, Oct. 2003, pp. 289 – 292.
7. J. Kim, E. Peli, *MPEG based image enhancement for people with low vision,* Soc for Information Display, Digest of Technical Papers, 2003, pp. 1156-1159.
8. V. Mezaris, I. Kompatsiaris, M. G. Strintzis, *Compressed-domain object detection for video understanding,* Proc. Workshop on Image Analysis For Multimedia Interactive Services (WIAMIS), April 2004, Lisbon, Portugal.
9. V. Mezaris, I. Kompatsiaris, E. Kokkinou and M. G. Strintzis, *Real-time compressed-domain spatiotemporal video segmentation,* Proc. Third International Workshop on Content-Based Multimedia Indexing, France, 2003, pp. 373-380.
10. Pedro Miguel Fonseca and Jan Nesvadba – *Face Tracking in the Compressed Domain,* EURASIP Journal on Applied Signal Processing, vol. 2006, artic. ID 59451, 11 pg, 2006.
11. B. K. Gunturk, Yucel Altunbasak, *Multiframe Resolution-Enhancement Methods for Compressed Video,* Ieee Signal Processing Letters, Vol. 9, No. 6, June 2002.
12. Brian C. Smith, *A survey of compressed domain processing techniques,* in Proc. NSF Workshop on Reconnecting Science and Humanities in Digital Libraries, Oct. 1995.
13. Gonzales R. C., Woods R. E, *Digital Image Processing,* Adison Wesley, 1993.

Proceedings of the International Multiconference on Computer Science and Information Technology pp. 171–181
ISSN 1896-7094
© 2007 PIPS

A Comparative Analysis of Crossover Variants in Differential Evolution

Daniela Zaharie

Faculty of Mathematics and Computer Science
West University of Timişoara
bv. Vasile Pârvan, nr. 4, 300223 Timişoara, Romania
dzaharie@info.uvt.ro

Abstract. This paper presents a comparative analysis of binomial and exponential crossover in differential evolution. Some theoretical results concerning the probabilities of mutating an arbitrary component and that of mutating a given number of components are obtained for both crossover variants. The differences between binomial and exponential crossover are identified and the impact of these results on the choice of control parameters and on the adaptive variants is analyzed.

Keywords: differential evolution, crossover operators, parameter control

1 Introduction

Differential evolution (DE) [9] is a population based stochastic heuristic for global optimization on continuous domains which is characterized by simplicity, effectiveness and robustness.

Its main idea is to construct, at each generation, for each element of the population a mutant vector. This mutant vector is constructed through a specific mutation operation based on adding differences between randomly selected elements of the population to another element. For instance one of the simplest and most used variant to construct a mutant vector, y, starting from a current population $\{x_1, \ldots, x_m\}$ is based on the following rule: $y = x_{r_1} + F \cdot (x_{r_2} - x_{r_3})$ where r_1, r_2 and r_3 are distinct random values selected from $\{1, \ldots, m\}$ and $F > 0$ is a scaling factor. This difference based mutation operator is the distinctive element of DE algorithms allowing a gradual exploration of the search space. Based on the mutant vector, a trial vector is constructed through a crossover operation which combines components from the current element and from the mutant vector, according to a control parameter $CR \in [0, 1]$. This trial vector competes with the corresponding element of the current population and the best one, with respect to the objective function, is transferred into the next generation. In the following we shall consider objective functions, $f : D \subset \mathbb{R}^n \to \mathbb{R}$, to be minimized.

The general structure of a DE (see algorithm 1) is typical for evolutionary algorithms, the particularities of the algorithm being related with the mutation

and crossover operators. By combining different mutation and crossover operators various schemes have been designed. In the DE literature these schemes are denoted by using the convention $DE/a/b/c$ where a denotes the manner of constructing the mutant vector, b denotes the number of differences involved in the construction of the mutant vector and c denotes the crossover type.

Algorithm 1 The general structure of a generational DE

```
Population initialization X(0) ← {x_1(0), ..., x_m(0)}
g ← 0
Compute {f(x_1(g)), ..., f(x_m(g))}
while the stopping condition is false do
  for i = 1,m do
    y_i ← generateMutant(X(g))
    z_i ← crossover(x_i(g),y_i)
    if f(z_i) < f(x_i(g)) then
      x_i(g+1) ← z_i
    else
      x_i(g+1) ← x_i(g)
    end if
  end for
  g ← g+1
  Compute {f(x_1(g)), ..., f(x_m(g))}
end while
```

The behavior of DE is influenced both by the mutation and crossover operators and by the values of the involved parameters (e.g. F and CR). During the last decade a lot of papers addressed the problem of finding insights concerning the behavior of DE algorithms. Thus, parameter studies involving different sets of test functions were conducted [4, 8, 7] and a significant number of adaptive and self-adaptive variants have been proposed [2, 6, 10, 12]. Most of these results were obtained based on empirical studies. Despite some theoretical analysis of the DE behavior [1, 3, 11] the theory of DE is still behind the empirical studies. Thus theoretical insights concerning the behavior of DE are highly desirable.

On the other hand, most of DE variants and studies are related with the mutation operator. The larger emphasis on mutation is illustrated by the large number of mutation variants, some of them being significantly different from the first versions of DE (e.g. [5, 1]). The crossover operator attracted much less attention, just two variants being currently used, the so-called binomial and exponential crossover. If the exponential crossover is that proposed in the original work of Storn and Price [9], the binomial variant was much more used in applications. Besides statements like "*The crossover method is not so important although Ken Price claims that binomial is never worse than exponential*" [13] or some recent experimental studies involving both binomial and exponential variants [7] no systematic comparison between these two crossover types was conducted.

The aim of this paper is to analyze the similarities and differences between binomial and exponential crossover emphasizing on their theoretical properties and on their influence on the choice of appropriate control parameters. By such an analysis we can hope to find insights into the behavior of DE and to find explanations for statements like: "*if you choose binomial crossover like, CR is usually higher than in the exponential crossover variant*" [13].

The rest of the paper is structured as follows. Section 2 presents the implementation details of binomial and exponential crossover variants. In section 3 some theoretical results concerning the probability of selecting a component from the mutant vector and concerning the average number of mutated components are derived. Based on these results, in section 4 is analyzed the influence of crossover variants on the choice of control parameters. Section 5 concludes the paper.

2 Crossover variants in differential evolution

The crossover operator aims to construct an offspring by mixing components of the current element and of that generated by mutation. There are two main crossover variants for DE: binomial (see Algorithm 2) and exponential (see Algorithm 3). In the description of both algorithms *irand* denotes a generator of random values uniformly distributed on a finite set, while *rand* simulates a uniform random value on a continuous domain. For both crossover variants the mixing process is controlled by a so-called crossover probability usually denoted by CR.

In the case of binomial crossover a component of the offspring is taken with probability CR from the mutant vector, y, and with probability $1-CR$ from the current element of the population, x. The condition "$rand(0,1) < CR$ or $j = k$" of the **if** statement in Algorithm 2 ensures the fact that at least one component is taken from the mutant vector. This type of crossover is very similar with the so-called uniform crossover used in evolutionary algorithms. On the other hand, the exponential crossover is similar with the two-point crossover where the first cut point is randomly selected from $\{1, \ldots, n\}$ and the second point is determined such that L consecutive components (counted in a circular manner) are taken from the mutant vector. In their original paper [9], Storn and Price suggested to choose $L \in \{1, \ldots, n\}$ such that $Prob(L = h) = CR^h$. It is easy to check that this is not a probability distribution on $\{1, \ldots, n\}$ but just a relationship which suggest that the probability of mutating h components increases with the parameter CR and decreases with the value of h, by following a power law. Such a behavior can be obtained by different implementations. The most frequent implementation is that described in Algorithm 3 where $\langle j+1 \rangle_n$ is just $j+1$ if $j < n$ and it is 1 when $j = n$.

Besides the fact that the exponential crossover allows to mutate just consecutive, in a circular manner, elements while binomial crossover allows any configuration of mutated and non-mutated components there is also another difference between these strategies. In the binomial case the parameter CR determines

Algorithm 2 Binomial crossover

```
crossoverBin (x,y)
k ← irand({1,...,n})
for j = 1,n do
    if rand(0,1) < CR or j = k then
        z^j ← y^j
    else
        z^j ← x^j
    end if
end for
return z
```

Algorithm 3 Exponential crossover

```
crossoverExp (x,y)
z ← x; k ← irand({1,...,n}); j ← k; L ← 0
repeat
    z^j ← y^j; j ← ⟨j+1⟩_n; L ← L+1
until rand(0,1) > CR or L = n
return z
```

explicitly the probability for a component to be replaced with a mutated one. In the implementation of the exponential crossover, CR is used to decide how many components will be mutated. However, in both situations CR influences the probability for a component to be selected from the mutant vector. This probability, denoted in the following by p_m, is in fact similar with the mutation probability in genetic algorithms and it is expected that its value has an influence on the DE behavior.

Because of these differences between the two crossover variants one can have different mutation probabilities and different distributions of the number of the mutated components for the same value of CR. These aspects are analyzed in more details in the next section.

3 A theoretical analysis

From a statistical point of view, the binomial crossover is achieved by a set of n independent Bernoulli trials, the result of each trial being used in selecting a component of the offspring from the mutant vector. If the constraint of having at least one mutated component is applied, the successful event in each Bernoulli trial is the union of two independent events, one of probability CR (event "$rand(0,1) < CR$") and one of probability $1/n$ (event "$j = irand(\{1,\ldots,n\})$"). Thus the probability that a component is mutated is $p_m = CR(1-1/n)+1/n$.

The number, L, of components selected from the mutated vector has a binomial distribution with parameters n and p_m. Thus the probability that h components are mutated is $Prob(L=h) = C_n^h p_m^h (1-p_m)^{n-h}$. Based on the

properties of the binomial distribution it follows that the average of the number of mutated components is $E(L) = np_m$.

If the stopping condition of the **repeat** loop in the Algorithm 3 would be just $rand(0,1) > CR$ then L would take values according to the geometric distribution on $\{1, 2, \ldots\}$ corresponding to the parameter $1 - CR$ (CR being interpreted as the success probability). In such a situation the probability that the number of mutated components is h would be $Prob(L = h) = CR^{h-1}(1 - CR)$. However in the exponential crossover the number of mutated components is bounded by L, thus we are dealing with a truncated geometric distribution.

Thus the probability distribution of L is given by:

$$Prob(L = h) = \begin{cases} (1 - CR)CR^{h-1} & \text{if } 1 \leq h < n \\ CR^{n-1} & \text{if } h = n \end{cases} \tag{1}$$

Using eq. 1 it follows that the average of L is $E(L) = (1 - CR^n)/(1 - CR)$. It remains now to find the value of p_m in the case of exponential crossover. There are two random variables simulated in the implementation of exponential crossover: the index, k, of the first mutated component and the number of mutated components. An arbitrary component, j, will be mutated if $d(j,k) < L$, where $d(j,k) = j - k$ if $j \geq k$ and $d(j,k) = n + j - k$ if $j < k$. Since k can take any value from $\{1, \ldots, n\}$ with probability $1/n$ the probability that an arbitrary component, j, is replaced with a component from the mutant vector is:

$$Prob(z^j = y^j) = \frac{1}{n}\sum_{k=1}^{n} Prob(d(j,k) < L) = \frac{1}{n}\sum_{d=0}^{n-1} Prob(L > d) \tag{2}$$

Since $Prob(L > d) = CR^d$ it follows that

$$Prob(z^j = y^j) = \frac{1}{n}\sum_{d=0}^{n-1} CR^d = \frac{1 - CR^n}{n(1 - CR)} \tag{3}$$

A summary of all these values corresponding to binomial and exponential crossover is presented in Table 1.

Table 1. Summary of theoretical results

Crossover type	p_m	$Prob(L = h)$	$E(L)$
Binomial	$CR(1 - 1/n) + 1/n$	$C_n^h p_m^h (1 - p_m)^{n-h}$	$CR(n-1) + 1$
Exponential	$\frac{1 - CR^n}{n(1 - CR)}$	$\begin{cases} (1 - CR)CR^{h-1} & 1 \leq h < n \\ CR^{n-1} & h = n \end{cases}$	$\frac{1 - CR^n}{1 - CR}$

Thus both the probability that an arbitrary component is mutated and the probability distribution of the number of mutated components are different between binomial and exponential crossover. More specifically, the dependence between p_m and CR is linear in the case of binomial crossover and nonlinear in the exponential case. Figure 1 illustrates the fact that for the same value of $CR \in (0, 1)$ the mutation probability is smaller in the case of exponential crossover than in the case of binomial one, the difference being more significant if n is larger.

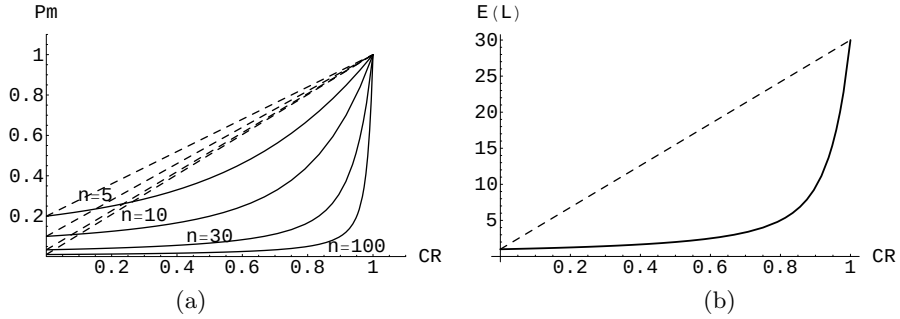


Fig. 1. Influence of CR on the mutation probability (a) and on the average of the number of mutated components for $n = 30$ (b) in the case of binomial crossover (dashed line) and exponential crossover (continuous line)

4 Influence of the crossover variant on the choice of control parameters

Since for the same value of CR the probability of mutating a component and the average number of mutated components are different for binomial and exponential crossover it follows that the results of a parameter study conducted for one crossover variant are not necessary true for the other one. The correspondence between values of CR and mutation probability in binomial and exponential crossover is presented in Table 2 for two dimensions of the problem ($n = 30$, $n = 100$). As figure 1 also suggests, in the case of exponential crossover there are two ranges of values for CR with different impact on the effect of crossover. The first range, $[0, CR_1]$, is characterized by a low sensitivity of the algorithm behavior to the value of CR while the second one $[CR_1, 1]$ characterized by a high sensitivity. The threshold value, CR_1, is higher for higher values of n. Thus, in the case of exponential crossover, as n is higher the sensitive range of CR is smaller (being included in $[0.9, 1]$). For instance when $n = 100$ for $CR \in [0, 0.9]$ the mutation probability, p_m, varies only between 0.01 and 0.09. This means

Table 2. Correspondence between CR and the mutation probability, p_m, for binomial and exponential crossover

n	CR	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	0.92	0.95	0.97	0.99	1
30 (bin)	p_m	0.03	0.13	0.23	0.32	0.42	0.52	0.61	0.71	0.81	0.90	0.92	0.95	0.97	0.99	1
30 (exp)	p_m	0.03	0.03	0.04	0.04	0.05	0.06	0.08	0.11	0.16	0.31	0.38	0.52	0.66	0.86	1
100 (bin)	p_m	0.01	0.11	0.21	0.31	0.41	0.51	0.60	0.70	0.80	0.90	0.92	0.95	0.97	0.99	1
100 (exp)	p_m	0.01	0.01	0.01	0.01	0.01	0.02	0.02	0.03	0.05	0.09	0.12	0.19	0.31	0.63	1

that parameters studies concerning CR should be differently conducted in the case of binomial and exponential crossover.

In order to illustrate the influence of the crossover variant on the sensitivity of DE to different values of CR some empirical tests were conducted. Tables 3 and 4 present the dependence between the number of function evaluations (nfe*1000) until the global optimum is approached with an accuracy of $\epsilon = 10^{-6}$ and values of CR in the case of two test functions: a multimodal separable one (e.g. Rastrigin [6]) and a multimodal nonseparable one (e.g. Griewank [6]) both of dimension $n = 30$ and having a global minimum in 0. In both cases we used a $DE/rand/1/c$ variant, a rather small population size ($m = 50$) and the same value for the scaling factor, $F = 0.5$. The maximal number of evaluations was set to 250000 (nfe=250). The absence of the nfe value means that the algorithm did not approximated the global minimum with the desired accuracy. All results are averages obtained for 30 independent runs. They confirm the fact that the behavior of the algorithm depends on the value of the mutation probability, p_m, meaning that for the same value of p_m (which corresponds to different values of CR for binomial and exponential crossover) similar behavior is observed. In the case of Rastrigin function, which is a separable one, good behavior is obtained for small values of p_m [8] which in the case of the binomial crossover corresponds to small values of CR. In the case of exponential crossover, since for a large range of CR the mutation probability is small, the set of CR values for which the algorithm is able to identify the global optimum with the desired accuracy is significantly larger than in the case of binomial crossover.

For the Griewank function the best behavior is obtained for values of CR in the range $[0.1, 0.5]$ in the case of binomial crossover and in the range $[0.7, 0.95]$ in the case of exponential crossover. Both ranges of CR values corresponds to similar ranges of p_m: $[0.13, 0.52]$ (binomial crossover) and $[0.11, 0.52]$ (exponential crossover). The different results obtained by the two crossover variants for similar values of p_m can be explained by the fact that in the case of nonseparable functions mutating a sequence of components (as in exponential crossover) or arbitrary components (as in binomial crossover) generates different exploration patterns.

The previous experiments were based on the same value of the scaling parameter F. Since the control parameters in DE are interrelated one would expect to have different appropriate values of F for the same value of CR when different types of crossover are used. Since DE is prone to premature convergence

Table 3. Number of function evaluations (nfe*1000) needed to approximate the optimum with the accuracy $\epsilon = 10^{-6}$. Test function: Rastrigin, $n = 30$; Algorithm: DE/rand/1, $m = 50$, $F = 0.5$.

CR	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	0.92	0.95	0.97	0.99	1
p_m (bin)	0.03	0.13	0.23	0.32	0.42	0.52	0.61	0.71	0.80	0.90	0.92	0.95	0.97	0.99	1
nfe (bin)	45.4	74.6	182.8	-	-	-	-	-	-	-	-	-	-	-	-
p_m (exp)	0.03	0.04	0.04	0.05	0.06	0.07	0.08	0.11	0.16	0.31	0.38	0.52	0.66	0.86	1
nfe (exp)	45.5	46.0	46.8	47.8	49.2	51.3	53.8	59.0	68.4	99.2	115.4	162.2	240	-	-

Table 4. Number of function evaluations (nfe*1000) needed to approximate the optimum with the accuracy $\epsilon = 10^{-6}$. Test function: Griewank, $n = 30$; Algorithm: DE/rand/1, $m = 50$, $F = 0.5$.

CR	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	0.92	0.95	0.97	0.99	1
p_m (bin)	0.03	0.13	0.23	0.32	0.42	0.52	0.61	0.71	0.80	0.90	0.92	0.95	0.97	0.99	1
nfe (bin)	62.6	39.0	36.2	35.1	36.6	39.5	41.4	45.7	59.3	72.4	72.9	72.9	75.9	96.5	91.9
p_m (exp)	0.03	0.04	0.04	0.05	0.06	0.07	0.08	0.11	0.16	0.31	0.38	0.52	0.66	0.86	1
nfe (exp)	60.9	58.6	58.7	56.2	51.6	49.6	46.6	43.5	40.2	42.2	41.2	44.7	48.8	65.1	94.4

one first issue in choosing the control parameters of DE is to try to avoid such a situation. Starting from the ideas that premature convergence is related with loss of diversity and that the diversity is related with the population variance in [11] is derived a theoretical relationship between the control parameters and the population variance after and before applying the variation operators. More specifically, if $\mathrm{Var}(z)$ and $\mathrm{Var}(x)$ denote the averaged variance of the trial and current populations respectively then the following relationship is true:

$$\mathrm{Var}(z) = (2p_m F^2 - \frac{2p_m}{m} + \frac{p_m^2}{m} + 1)\mathrm{Var}(x) \tag{4}$$

Based on this linear dependence between these two variances one can control the impact of the mutation and crossover steps on variance modification by imposing that

$$2p_m F^2 - \frac{2p_m}{m} + \frac{p_m^2}{m} + 1 = c \tag{5}$$

where c should be 1 if we are interested to keep the same value of the variance or slightly larger than 1 in order to stimulate an increase of the diversity (for instance $c = 1.05$ means an increase of the population variance with 5% and $c = 1.1$ means an increase with 10%). The result in [11] was obtained only in the case of binomial crossover considering that $p_m = CR$. By replacing in eq. 5 p_m with $CR(1-1/n)+1/n$ or with $(1-CR^n)/(n(1-CR))$, one obtains equations involving F,CR,m and n. By solving these equations with respect to F one can obtain lower bounds for F which allow avoiding premature convergence. The dependence of such lower bounds of F on the values of CR for two values of the constant c is illustrated in Figure 2. The differences between the value of F_{min} in the case of binomial and exponential crossover suggest that for the same value

of $CR \in (0, 1)$ the exponential crossover needs a larger value of F in order to induce the same effect on the population variance.

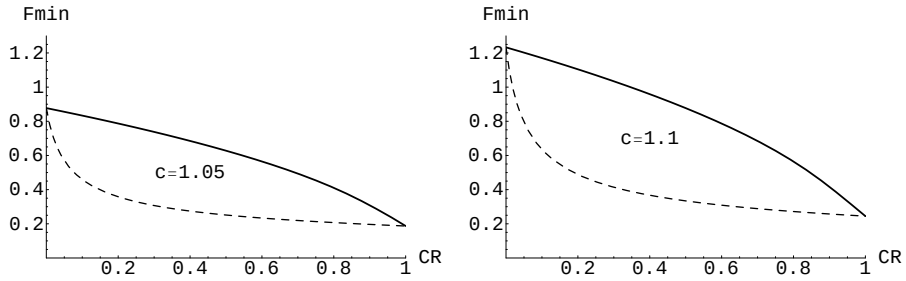


Fig. 2. Lower bound for F vs. CR for binomial crossover (dashed line) and exponential crossover (continuous line). Parameters: $m = 50$, $n = 30$

These differences should be taken into account when conducting parameter studies on DE variants involving binomial and exponential crossover(like in [7]). For instance when tuning the value of CR the use of a uniform discretization of $[0, 1]$ as in [7] is appropriate for binomial crossover but not necessarily appropriate for exponential crossover (since exponential DE is more sensitive to values of CR between $(0.9, 1]$ than to values between $(0, 0.9]$). Attention should be also paid when using in combination with exponential crossover an adaptive or self-adaptive variant of DE which was initially designed for binomial crossover. For instance in the adaptive variant designed to avoid premature convergence [12] the adaptation rules for F and CR should be modified according to eq. 4 and to the relationship between p_m and CR.

On the other hand for self-adaptive variants which use random selection for the control parameters values (like in [2]) one have to take into account the fact that if for binomial crossover a uniform distribution of CR values is appropriate (leading to a uniform distribution of p_m) a different situation appears in the case of exponential crossover. In this case uniformly distributed values of CR do not necessarily lead to uniformly distributed values of p_m, meaning that the adaptation strategy should be changed.

5 Conclusions

The comparative analysis of binomial and exponential crossover variants offered us some information about the influence of parameter CR on the behavior of DE. The dependence between the mutation probability, p_m and the crossover parameter, CR, was derived both in the case of binomial and of exponential crossover applied to differential evolution. This dependence is linear in the binomial case and nonlinear in the exponential one. For the same value of CR the

mutation probability is larger in the case of binomial crossover than in the case of exponential one, the difference being larger as the problem size, n, is larger. This means that in order to reach a similar effect by the mutation step a DE algorithm with exponential crossover should use a larger value for CR.

Moreover, in the case of exponential crossover one have to be aware of the fact that there is a small range of CR values (usually $[0.9, 1]$) to which the DE is sensitive. This could explain the rule of thumb derived for the original variant of DE: *"use values of CR in the range $[0.9, 1]$"*. On the other hand, for the same value of CR the exponential variant needs a larger value for the scaling parameter, F, in order to avoid premature convergence.

Acknowledgment

This work is supported Romanian grants
99-II CEEX 03-INFOSOC 4091/31.07.2006 and CNCSIS-MindSoft.

References

1. Ali M. M. and Fatti L. P.: *A Differential Free Point Generation Scheme in the Differential Evolution Algorithm,* Journal of Global Optimization, **35**, pp. 551-572, 2006.
2. Brest J., Boškovič B., Greiner S., Žurner V. and Maučec M. S.: *Performance comparison of self-adaptive and adaptive differential evolution algorithms,* Soft Computing, **11**(7), pp. 617–629, 2007.
3. Ter Braak J. F. C.: *A Markov Chain Monte Carlo version of the genetic algorithm Differential Evolution: easy Bayesian computing for real parameter spaces,* Stat Comput, **16**, pp. 239–249, 2006.
4. Gämperle R., Müller S. D. and Koumoutsakos P.: *A Parameter Study for Differential Evolution,* in A. Grmela, N. E. Mastorakis (eds.), Advances in Intelligent Systems, Fuzzy Systems, Evolutionary Computation, WSEAS Press, pp. 293–298, 2002.
5. Fan H. Y. and Lampinen J.: *A Trigonometric Mutation Operation to Differential Evolution,* Journal of Global Optimization, **27**, pp. 107–129, 2003.
6. Liu J. and Lampinen J.: *A fuzzy adaptive Differential Evolution,* Soft Computing, **9**, pp. 448–462, 2005.
7. Mezura-Montes E., Velásquez-Reyes J. and Coello Coello C. A.: *A Comparative Study of Differential Evolution Variants for Global Optimization,* in Maarten Keijzer et al. (editors), 2006 Genetic and Evolutionary Computation Conference (GECCO'2006), Vol. 1, ACM Press, Seattle, Washington, USA, July, pp. 485–492, 2006.
8. Rönkkönen J., Kukkonen S. and Price K. V.: *Real-parameter optimization with differential evolution,* Proceedings of CEC 2005, 2-3 September, Edinburgh, **1**, pp. 567–574, 2005.
9. Storn R. and Price K.: *Differential Evolution—a simple and efficient adaptive scheme for global optimization over continuous spaces,* International Computer Science Institute, Berkeley, TR-95-012, 1995.
10. Tvrdik J.: *Differential Evolution with Competitive Setting of Control Parameters,* Task Quarterly, **11**(1-2), pp. 169–179, 2007.

11. Zaharie D.: *Critical values for the control parameters of differential evolution algorithms,* in R. Matoušek and P. Ošmera (eds.), Proceedings of 8th International Conference on Soft Computing, Mendel 2002, pp. 62–67, 2002.
12. Zaharie D.: *Control of population diversity and adaptation in differential evolution algorithms,* in R. Matoušek and P. Ošmera (eds.), Proceedings of 9th International Conference on Soft Computing, Mendel 2003, pp. 41–46, 2003.
13. `http://www.icsi.berkeley.edu/~storn/code.html` Differential Evolution web page [last access: july, 2007].

Proceedings of the International Multiconference on Computer Science and Information Technology pp. 183–192
ISSN 1896-7094
© 2007 PIPS

Information Extraction Systems and Nominal Anaphora Analysis Needs

Ireneusz Matysiak

Politechnika Wrocławska
Instytut Informatyki Stosowanej
Wybrzeże Wyspiańskiego 27,
50-370 Wrocław, Poland
ireneusz.matysiak@pwr.wroc.pl

Abstract. This paper presents the needs for anaphora analysis in Information Extraction from practical point of view. It depicts the problem of anaphora phenomenon and its role in texts together with corpus examples. It encloses a short survey of the computational treatment of nominal anaphora. In the end, a uniform approach for nominal anaphora analysis is proposed. Moreover, the paper describes requirements for corpus preparation and framework implementation.

1 Introduction

1.1 Background

An increasing demand for *Information Extraction* (*IE*) systems and the tendency for improving their performance (precision and recall) call for context and coherence analysis. The coherence can be obtained by different linguistic phenomena among which *anaphora*[1] plays a crucial role. Anaphora analysis for information extraction purposes was taken into consideration during the Message Understanding Conferences (MUC)[2], the conferences were dedicated to information extraction systems, and finally formulated as *coreference task* during information extraction at the 7'th MUC ([14]). The task was restricted only to *coreferences*[3] and consists in collecting of all mentions of a given entity. The

[1] This paper concerns *anaphora* as a language phenomenon, which introduces the connection between pointing back expression *anaphor* (called *referent* as well) and *antecedent*; the present article discuses only *anaphors* that can take shape of a pronoun – *pronominal anaphora*, e.g. ,,If I have *the dog*, I'll walk *him*" (the pronoun *him* refers to the noun *the dog*; both point out the same object from a context) , or a non-pronominal noun phrase – *lexical noun phrase*, e.g. ,,I got *a small dog*. *The puppy* is great." (the anaphor is the noun *the puppy* and points back to the noun phrase *a small dog*). Both types are contained in wider term *nominal anaphora*.

[2] Sometimes called the Message Understanding Competition after its competition character.

[3] Coreference is regarded as type of anaphora where an identity relation, between the anaphor and the antecedent, is preserved.

present linguistic research, corpus evidences and practice combined with new sources of information like *WordNet* lead to the need for the redefinition of the coreference task.

1.2 Previous work on anaphora analysis

The first effort to define anaphora and create methods for analysing them were made in the late 70s by Sidner ([24]) and Hobbs ([15])[4]. Automatic methods from that time based on knowledge engineering – rules which mainly used *syntactic information*. Some of them where supported by late *binding theory* ([6]). Following modifications applied *cognitive techniques* i.e. *focus* (Sidner [24]), *short memory model* ([11]). Certain possibilities boundary of those algorithms can be observed in a frequently applied (and modified) Lappin & Leass' algorithm for pronominal anaphora resolution (Lappin and Leass [17]) (often used as baseline for comparison).

The 90s contributed to a better discourse and coherence understanding, which triggered the emergence of algorithms based on *centering theory* (Grosz [10]). In those times, one made an effort to use corpora for anaphora analysis and machine learning (ML) approach. Almost all kinds of ML methods were, more or less, investigated. However, for the sake of conciseness, only one species of a selected kind is quoted: *case base reasoning* in relative pronouns disambiguation ([4]), *decision trees* for coreference resolution ([18]), *unsupervised learning* (clustering) for noun phrases ([5]) (coreference resolution between noun phrases was treated as the grouping task, where algorithm found suitable similarity metrics determining the resolution). Even a *genetic algorithm* for finding salience weights in pronoun resolution system was implemented ([1]).

In an alternative corpus-based approach a large number of documents is processed for *statistical analysis* (those algorithms required a priori pre-selection of suitable features and a huge corpus size), e.g. personal pronoun resolution ([9]), non-pronominal anaphora ([12]) and coreferences (Hartrumpf [13]).

Recent scientific research tend to apply *knowledge-poor analysis* like Mitkov's approach to personal pronouns ([20]) (shallow parsing and rules which operate on empirical *antecedent indicators* independent of language; it achieved almost 90% *success rate*) or proper names coreferences ([2]) (hand-crafted orthographic rules for coreferences specific to text genre) and the machine learning methods: a decision tree for coreferences ([25]) (definite descriptions and some bridging anaphoras). There are studies on integration semantics and lexical information from various knowledge sources like *WordNet* with machine learning ([23]) or searching the Web for a particular type of lexico–syntactic pattern's statistics for resolving definite descriptions ([3]).

[4] Some heuristics for personal pronouns were implemented in micro worlds, e.g. Winograd's SHRDLU in late 60s.

2 Anaphora in IE

There are many varieties of anaphoras, which should be resolved if one wants to understand the meaning of the text. An interesting review of the issue is done by Krahmer & Piwek ([16]). Apart from the problem of definition and denotation undertaken in their studies, characteristic properties of anaphora are worth emphasising. Especially the two of them are interesting – a type of non-coreferential relation (*bridges*)[5] between an anaphor and the antecedent, and constraints on the relation which may have an impact on the scope. The authors enumerate a set of relations among nominal anaphora participants (though the article mentions relation for other types of anaphoras): *set membership*, *part-of* (necessary part, probable part, inducible part), which with relations from ontology (like *specialization/generalisation*), need to be investigated during analysis.

Since extraction information is driven by recognised entities (during *named entity task*), the coreference task become important part of processing, extending scope of extraction. The wider scope system has, the better results it gets, so capturing information about all mentions of a given entity (including non-coreferential relations) should be regarded as nominal anaphora resolution. Consequently, according to previous suggestions, the MUC task formulation should be expanded by adding described relations (with an effect on annotation scheme) as it is suggested by Deemter & Kibble ([8]).

In a modern approach to IE system creation machine learning is used to automate customisation to the text genre with the smallest effort. The system possibilities are limited by delivered data. To sum up, IE sets the *anaphora resolution* (*AR*) in the following context:

1. should cover a wide range of anaphora (at least the nominal anaphora),
2. has to have the best possible efficiency,
3. should be open for system delivered knowledge and domain (easy adjustable),
4. should be data driven (*data-enriched* approach),
5. does not need to be a general solution.

3 An uniform approach for nominal anaphora analysis

Many of the mentioned algorithms were restricted to a certain kind of anaphoras (with a few exceptions using machine learning techniques), i.e. personal pronouns, lexical noun phrases, and coreferences. However, the ways of their analysis through language levels are similar. Somehow, different kinds of anaphora require various types of knowledge, which determines the accuracy of resolving. The Figure 1 presents anaphora analysis methods at levels of the language analysis (the preliminary levels are omitted). The first and the simplest method (especially important for pronouns) usually described as *co-indexing* (after assignment of the same index to a referent and the anaphor) is based on syntactic

[5] Sometimes are referred to as *associative* and *associative anaphora*.

knowledge like a parsing tree and morphological attributes to validate assignment. A more complicated *binding*[6] can reflect more compound dependences coming from formalism, i.e. the binding theory, the centering theory, and sometimes looks up to the dictionary for lexical details needed for verification (the lexical knowledge could be built in formalism like in HPSG grammar).

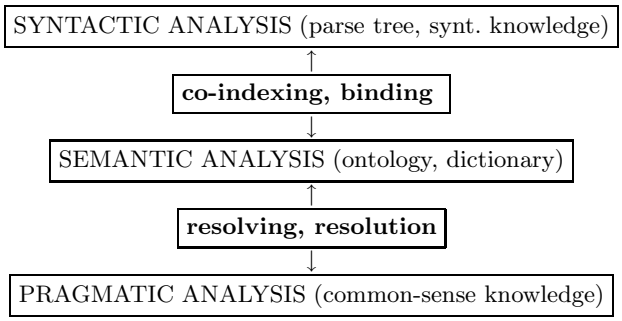


Fig. 1. Anaphora processing in the flow of natural language analysis

The terms *resolving* or *resolution* describe methods which perform a compounded analysis. They can use additional knowledge sources like ontologies, semantic dictionaries for examining potential antecedents. The method could access rules/heuristics involving a common-sense knowledge, and can be genre-specific (i.e. associative anaphora cannot do without the information about a relation between investigated entities). In some systems based on a *sense representation* (e.g. DRT) a reasoning can be carried out.

Approaches presented in Sect. 1.2 use, more or less, the knowledge sources in two ways. In the first one as *restrictions* (constraints) preventing from wrong binding. They can have various forms, starting from syntactic rules like *agreement of morphological attributes*, i.e. gender, number (a simple pronoun and his antecedent have to match number and gender, otherwise the antecedent is discounted), *semantic lexical features* like animacy, etc. (a semantic restriction from a context placed on an antecedent, must be met by an anaphor), ending with *syntactic rule* constraint taken from a selected theory like *c-command* (centering theory). The other strategy is *preference*, which is mandatory and its importance is weighted. Some examples based on syntactic knowledge are *lexical reiteration* (if a noun phrase was mentioned before, it gives more salience), *similar syntactic role* in the sentence, and *collocation* matching. Semantic preferences can be *similar semantic role*, *metonymy/holonymy/hyponymy/hypernymy* relations between an anaphor and the antecedent. Tracing the utterance object can cause preference of a *focused* object.

[6] The distinction between co-indexing and binding is not obvious in the literature, but helps to organise terminology.

A uniform approach requires treating nominal anaphora as a whole, and should allow to integrate any type of knowledge (which determines the quality of the solution and causes that the solution is driven by data). Also, it need to be easily adapted for the target use (to the delivered knowledge and application domain).

These requirements inspired searching a predictive model that could perform mapping from observations about anaphor features to conclusions about its antecedent. The machine learning technique like a decision tree seems to be a good solution. Moreover, it could support integration of rule based algorithms as special kind of test nodes – *computational nodes*. The nodes are not ordinary attribute test, but the result from the rule/algorithm (e.g. focus tracing algorithm or even complete anaphora algorithm like co-indexation). Moreover, it might give a chance to recover from wrong rule/algorithm result (in case of the involved node is not final). The salience carried by the test (attribute or computational node) can be automatically weighed during the tree creation. In addition, the automatic tree creation can be a useful way of adjusting an algorithm to a specific genre. In that approach, there is no distinction between preferences and restrictions[7], but the negative and final test result might be interpreted as the restriction.

4 Corpus

The corpus consists of documents published by stock companies quoted at the Warsaw Stock Exchange. The documents contain reports concerning current companies activities and events, which (according to the law) must be announced, i.e. significant agreement contraction, regulations established during a meeting of shareholders, changes of members of the board, etc[8]. The initial corpus has 4175 parsed documents (average file size is 2,5 KB). Some statistics about the corpus are presented in Table 1[9]. The collection is a starting point for creating the information extraction system. The preliminary investigation reveals that for those type of documents a formal language is used, which increases the role of nominal phrases (especially definite descriptions and proper names, see comparison in Table 1). Such situation requires more accurate valuation of a relation between an anaphor and the antecedent.

In the corpus document one can notice a rich set of relations among entities, which could be used anaphorically like part-of relation: członek zarządu (member of the board) – Zarząd (the board); part/whole relation: Zarząd (company board) – Spółka Akcyjna (registered company); set/subset: Grupa Kapitałowa (capital group) – spółka zależna (controlled company), synonymy: firma (firm) –

[7] With exception of the case where decision tree is only preference selector.

[8] Detailed list of the events can be obtained from *Rozporządzenie Rady Ministrów w sprawie informacji bieżących i okresowych przekazywanych przez emitentów papierów wartościowych.*

[9] Since corpus is still in preparation the statistics were collected in a full automatic way and should be considered as rough estimation.

Table 1. Corpus in numbers

	Total	Mean per file	Mean to tokens	%
Tokens	846000	203	1	100
Noun phrases	401745	96	0.47	47
Proper names (found by parser)	18931	4.5	0.02	2
Personal pronouns	3064	0.7	0.003	0.3
Possessive pronouns	1689	0.4	0.002	0.2
Relative pronouns	1188	0.3	0.001	0.1
Reciprocal pronouns	383	0.09	0.0006	0.06

przedsiębiorstwo (company); identity of proper names: Zarządca Warszawa S.A. – Zarządca S.A.; generalization/specialization: Spółka Akcyjna (registered company) – Emitent (share drawer) – Swarzęc S.A.; Akcjonariat (shareholders) – Akcjonariusz (shareholder).

4.1 Annotation scheme

For anaphora it is important to mark both a relation type, where it is possible, and additional information about the other antecedents – *anaphoric chain*. Remarks made by Deemter & Kibble ([8]) reject the MUC annotation scheme, because of the corpus evidence of relation (which are not always the identity type) between entities (anaphor – referent) and the issue of marking the same entity in different manner (e.g. separating/joining entities from compound relations). The adapted *MATE* schema ([22]), then, seems to be the most appropriate. The adaptation consists in using MATE scheme for marking discourse entities (enables compound relations), anaphors (enable an anaphoric chain) and types of anaphoric relations: *ident* (identity), *subset* (set/subset relation), *poss* (part/whole relation), *genrel* (generalisation/specialization (from the previous version of MATE)[10]. In original scheme annotations are added to a text in SGML manner, but it is not the most practical solution from the processing point of view. To make it independent of processing resources, like tokenizers and proper name extractors, and easy to transform into other format, annotation should be saved in XML format, with a connection to the original clean text (like file offset) instead of tokens. That assumption puts GATE[11] annotation scheme (changed TIPSTER scheme) into the game. In the end, these annotations can be read and processed by GATE application (General Architecture for Text Engineering ([7])) for further purposes.

[10] An addition attribute *subtype* are used, but unlike MATE it contains information about referent form – noun/noun phrase/pronoun; its value is deduced from parser output and used for statistical purposes.

[11] GATE proposes a coreference annotation scheme, but it does not meet the proposed requirements.

4.2 Web annotation tool

The web annotation tool enables an easy access to the corpus and work without installation problems in any operating system. Moreover, it is simple to maintain and extend. The application consists of two parts (implemented as J2EE application).

The first part organises an access to the corpus files, saves annotation results and files status, whilst the second one transforms (using XSLT) the corpus document content into a web page with javascripts to anaphora annotation (on the web page some proper names, pronouns are marked out to speed up a selection process). In a typical session, an annotator: 1) selects an anaphor (marking in the text the beginning and the end); 2) chooses an appropriate relation (the default 'antecedent' – in case of uncertainty of the relation, 'equality' in case of the identity-of-relation anaphora, 'part' in case of the 'part/whole' relation, 'set' – 'set/subset' relation', 'similarity' – generalization/specialization etc., or 'none' if there is no antecedent); 3) links it to the antecedent (selecting it in the same way as an anaphor). A typical user session is presented in Fig. 2.

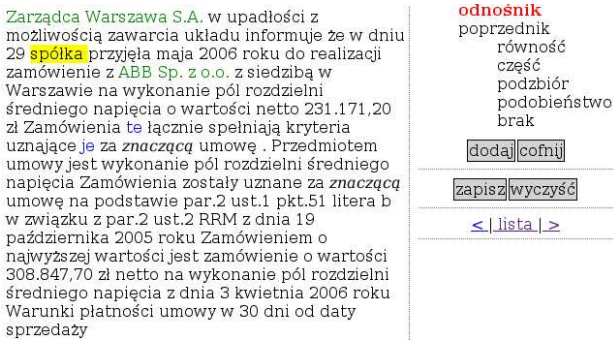


Fig. 2. Screen shot from the web application

The annotation results are kept in separate files for each user (post processed into XML as described in Sect. 4.1) and merged with the original text into one file with annotations.

5 The proposal for framework

The framework is practical realisation of the uniform approach for IE needs. It also utilises the annotation scheme (described in Sect. 4.1). It does not force implementation details, rather than helps to manage processed files and anaphora cases, organises document flow, gives suggestions about an interface for the IE *anaphora resolution module*[12] and a way the module should operate.

[12] The IE system module performing anaphora resolution task.

A typical framework document flow through different forms is depicted in Fig. 3.

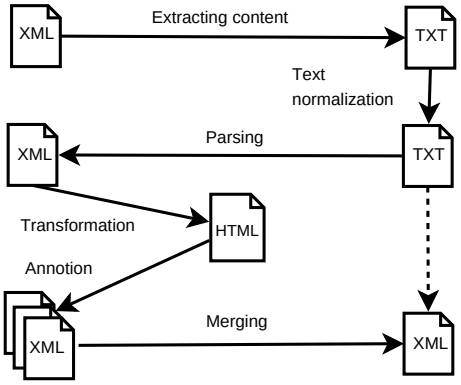


Fig. 3. Document flow

The distinction between document content and its annotated form allows using different document versions (through assignment of XSLT file to the version), so one document can be processed by various tools. Separation clean text files from processed and annotated ones requires document form on demand service from framework.

The anaphora resolution module should operate in four steps:

1. get information about anaphor and all candidate antecedents; according to the approach the system delivers anaphora context (feature vector) – attribute values and the computational nodes execution results (the feature vector should be configurable and can have a simple *CSV* format, e.g. one row contains an anaphor and a candidate features),
2. perform resolution via a decision tree (DT) (the result is a type of anaphora or none, if failed); DT could be treated as an algorithm for anaphora resolution ([18]) or antecedent filter ([21]), in the last case a selection strategy should be used,
3. choose the best antecedent according to the selection strategy (skip this step if DT is an algorithm),
4. return the antecedent – the row number (from CSV input) of selected antecedent (or none if it is a discourse new entity).

A module to compute feature values is dependent on the IE system, so it should be a part of IE system. Moreover, it can use different knowledge sources and algorithms, thus data which it operates rely on implementation and could be in an incompatible form (i.e. different parser output format, ontologies, and a serialization format).

Implementation of the anaphora resolution module consists of: 1) creation of associated XSLT file, which is depended on IE system processing tools like tagger, parser, tokenizer; 2) annotation of the corpus with anaphora annotations; 3) implementation of the feature value computation module (and computational nodes); 4) DT training (implementation of DT and strategy selection, if appropriate); 5) anaphora resolution module implementation (implementing access to DT and selection strategy).

The framework supporting varieties of salience features can be a useful tool for investigating various types of nominal anaphora. Furthermore, it can be a handy test bed for checking impact of different knowledge sources, processing tools on the resolution process.

6 Future work

The research is still at the conception stage and a lot of work needs to be done. First, preparation of the corpus with anaphora annotations. Second, implementation of well-known algorithms for Polish language (algorithm for personal pronouns ([19]) is promising; it was tested on few Polish hand prepared documents and obtained a success rate 93.3%; but for comparison a most-recent strategy is desirable as well), which could be used by the approach as computational nodes. Next, implementation of the IE anaphora resolution module (preparation learning cases set for automatic creation of a decision tree, using the following knowledge sources: the Polish WordNet, stock exchange ontology, parser output, selected focus and preferences techniques from the previous works). Having working framework, it would be useful to investigate some aspects, i.e. core anaphora features, which can be adjusted to a specific text genre; computational nodes; recovering from algorithm errors; an impact of different knowledge sources and errors from the previous stages (working in a fully automatic environment).

Acknowledgement. Work financed by the Polish Ministry of Education and Science, project No. 3 T11C 018 29.

References

1. Byron D., Allen J.: Applying genetic algorithms to pronoun resolution. In *AAAI/IAAI*, page 957, 1999.
2. Bontcheva K., Dimitrov M., Maynard D., Tablan V., Cunningham H.: Shallow methods for named entity coreference resolution. In *Proceedings of TALN*. TALN, 2002.
3. Bunescu R.: Associative anaphora resolution: A web-based approach, 2002.
4. Cardie C.: Corpus-based acquisition of relative pronoun disambiguation heuristics. In *Meeting of the Association for Computational Linguistics*, pages 216–223, 1992.
5. Cardie C., Wagstaff K.: Noun phrase corefernce as clustering. In *Proceedings of the Join SIGDAT Conference on Empirical Methods in NLP and Very Large Corpora*, pages 82 – 89, University of Maryland, USA, 1999.
6. Chomsky N.: *Lectures on Government and Binding: The Pisa Lectures*. Mouton de Gruyter, 1981.

7. Cunningham H.: GATE, a General Architecture for Text Engineering. *Computers and the Humanities*, 36:223–254, 2002.
8. Deemter van K., Kibble R.: On coreferring: Coreference in MUC and related annotation schemes. *Computational Linguistics*, 26(4):629–637, 2000.
9. Ge N., Hale J., Charniak E.: A statistical approach to anaphora resolution. In *Proceedings of the Workshop on Very Large Corpora*, pages 161–170, 1998.
10. Grosz B., Joshi A., Weinstein S.: Centering: a framework for modelling the local coherence of discourse. *Computational Linguistics*, 21(2):203–226, 1995.
11. Guindon R.: Anaphora resolution: Short-term memory and focusing, 1985.
12. Hall K.: A statistical model of nominal anaphora. Technical report, Brown University, 2001.
13. Hartrumpf S.: Coreference resolution with syntactico-semantic rules and corpus statistics. In *Proceedings of the Fifth Computational Natural Language Learning Workshop (CoNLL-2001)*, pages 137–144, 2001.
14. Hirschman L., Chinchor N.: MUC-7 Coreference Task Definition (version 3.0). In *Proceedings of the 7th Message Understanding Conference (MUC-7)*, Fairfax, Virginia, 1998.
15. Hobbs J.: Resolving pronoun references. *Lingua*, 44(8):311–33, 1978.
16. Krahmer E., Piwek P.: *Varieties of Anaphora: Introduction*, chapter Varieties of Anaphora, pages 1–15. Reader ESSLLI, Birmingham, 2000.
17. Lappin S., Leass H.: An algorithm for pronominal anaphora resolution. *Computational Linguistics*, 20(4):535–561, 1994.
18. McCarthy F., Lehnert W.: Using decision trees for coreference resolution. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence (IJCAI '95)*, 1995.
19. Mitkov R., Evans R., Orasan C., Barbu C., Jones L., Sotirova V.: Coreference and anaphora: developing annotating tools, annotated resources and annotation strategies, 2000. DAARC.
20. Mitkov R.: *Anaphora resolution*. Pearson Education, 2002.
21. Paul M., Yamamoto, K. and Sumita, E.: Corpus-based anaphora resolution towards antecedent preference, 1999.
22. Poesio M.: The mate/gnome proposals for anaphoric annotation, revisited. In *Proceedings of the 5th SIGdial Workshop on Discourse and Dialogue*, pages 154–162, Cambridge, Massachusetts, USA, April 30 - May 1 2004. Association for Computational Linguistics.
23. Poesio M., Ishikawa T., Schulte im Walde S., and Viera R.: Acquiring Lexical Knowledge for Anaphora Resolution. In *Proceedings of the 3rd Conference on Language Resources and Evaluation*, volume IV, pages 1220–1224, Las Palmas, Spain, 2002.
24. Sidner C.: Towards a computational theory of definite anaphora comprehension in english discourse. Technical report, MIT, 1979.
25. Vieira R.: Applying inductive decision trees in co-reference resolution of definite NPs, 1999. ASAI-99.

Proceedings of the International Multiconference on Computer Science and Information Technology pp. 193–202
ISSN 1896-7094
© 2007 PIPS

Adaptive Differential Evolution: Application to Nonlinear Regression

Josef Tvrdík

University of Ostrava, Department of Computer Science *
30. dubna 22, 701 03 Ostrava, Czech Republic
Josef.Tvrdik@osu.cz
http://albert.osu.cz/tvrdik

Abstract. Adaptation of control parameters in differential evolution is considered. Five adaptive variants of differential evolution are applied to the estimation of parameters in nonlinear regression models and their performance is compared experimentally with the adaptive controlled random search algorithm tailored especially for these problems. The NIST nonlinear regression datasets are used as a benchmark. Two of five tested variants of adaptive differential evolution perform almost as reliable as the adaptive controlled random search algorithm and one of these two variants converges only slightly slower and its time requirements are almost comparable with the adaptive controlled random search.

1 Introduction

The global optimization problem in this paper follows the form:

$$\text{minimize } f(\boldsymbol{x}) \text{ subject to } \boldsymbol{x} \in D\,,$$

where $\boldsymbol{x}$ is a continuous variable with the domain $D \subset \mathbb{R}^d$, and $f(\boldsymbol{x}) : D \to \mathbb{R}$ is a continuous function. The domain D is defined by specifying lower (a_j) and upper (b_j) limits of each component x_j, $D = \prod_{j=1}^{d}[a_j, b_j]$, $a_j < b_j$, $j = 1, 2, \ldots, d$. This specification of D is also called the *box constrains*. The global minimum point $\boldsymbol{x}^* = \arg\min_{\boldsymbol{x}\in D} f(\boldsymbol{x})$ is the solution of the problem.

Differential evolution (DE) introduced by Storn and Price [10] is simple but powerful evolutionary algorithm for global optimization over the box-constrained search space. The differential evolution has become one of the most frequently used algorithms for solving the continuous global optimization problems in recent years [8].

The aim of this paper is to compare numerically some self-adaptive variants of differential evolution with the specialized adaptive controlled random search algorithm [15] in the estimation of nonlinear-regression parameters. Section 2 concerns with the differential evolution and the setting of its control parameters. Principles of adaptive variants of differential evolution that have been used

* Supported by the grant 201/05/0284 of the Czech Grant Agency and by MSM 6198898701 of the Institute for Research and Applications of Fuzzy Modeling.

in experimental comparison are described in Section 3. In Section 4 the aditive nonlinear regression model is formed and the problem of parameter estimation is mentioned briefly. Numerical experiments on NIST nonlinear-regression benchmark tests are presented in Section 5. Some concluding remarks are made in Section 6.

2 Differential Evolution and its Control Parameters

The algorithm of differential evolution in pseudo-code is written as Algorithm 1.

Algorithm 1. Differential evolution

```
1  generate an initial population P = (x_1, x_2, ..., x_N), x_i ∈ D
2  repeat
3      for i := 1 to N do
4          generate a new trial vector y
5          if f(y) < f(x_i) then  insert y into new generation Q
6                          else  insert x_i into new generation Q
7          endif
8      endfor
9      P := Q
10 until stopping condition
```

A new trial point $\boldsymbol{y}$ is generated by using mutation and crossover. There are various strategies how to generate the mutant point $\boldsymbol{u}$. The most popular strategy called DE/rand/1/ generates the point $\boldsymbol{u}$ by adding the weighted difference of two points

$$\boldsymbol{u} = \boldsymbol{r}_1 + F\left(\boldsymbol{r}_2 - \boldsymbol{r}_3\right), \tag{1}$$

where $\boldsymbol{r}_1, \boldsymbol{r}_2$ and $\boldsymbol{r}_3$ are three mutually distinct points taken randomly from P not coinciding with the current $\boldsymbol{x}_i$, and $F > 0$ is an input parameter. Another strategy called DE/best/2/ generates the point $\boldsymbol{u}$ according to formula

$$\boldsymbol{u} = \boldsymbol{x}_{\min} + F\left(\boldsymbol{r}_1 + \boldsymbol{r}_2 - \boldsymbol{r}_3 - \boldsymbol{r}_4\right), \tag{2}$$

where $\boldsymbol{x}_{\min}$ is the point of P with minimal function value, $\boldsymbol{r}_1, \boldsymbol{r}_2, \boldsymbol{r}_3, \boldsymbol{r}_4$ are four mutually distinct points taken randomly from P not coinciding with the current $\boldsymbol{x}_i$ or $\boldsymbol{x}_{\min}$, and $F > 0$ is an input parameter.

Kaelo and Ali [4] proposed a slightly different attempt to generating a mutant point $\boldsymbol{u}$ by (1). They called it random localization. The point $\boldsymbol{r}_1$ is not selected randomly but it is tournament best among $\boldsymbol{r}_1$, $\boldsymbol{r}_2$, and $\boldsymbol{r}_3$, therefore $\boldsymbol{r}_1 = \arg\min_{i\in\{1,2,3\}} f(\boldsymbol{r}_i)$. Significant improvement in convergence (about 35% of function evaluations) with preserving the reliability of the search is reported for 50 test problems in [4].

The elements y_j, $j = 1, 2, \ldots, d$ of the trial point $\boldsymbol{y}$ are built up by the crossover of the current point $\boldsymbol{x}_i$ and the mutant point $\boldsymbol{u}$ using the following rule

$$y_j = \begin{cases} u_j & \text{if} \quad U_j \leq CR \quad \text{or} \quad j = l \\ x_{ij} & \text{if} \quad U_j > CR \quad \text{and} \quad j \neq l\,, \end{cases} \tag{3}$$

where l is a randomly chosen integer from $\{1, 2, \ldots, d\}$, $U_1, U_2, \ldots, U_d$ are independent random variables uniformly distributed in $[0,\ 1)$, and $CR \in [0, 1]$ is an input parameter influencing the number of elements to be exchanged by crossover. Eq. (3) ensures that at least one element of $\boldsymbol{x}_i$ is changed even if $CR = 0$. This kind of crossover according to (3) is called DE/$\cdot$/$\cdot$/bin.

The efficiency of differential evolution is very sensitive to the setting of values F and CR. The values recommended in literature are $F = 0.8$ and $CR = 0.5$, but even Storn and Price in their principal paper [10] use $0.5 \leq F \leq 1$ and $0 \leq CR \leq 1$ depending on the results of preliminary tuning. Several papers deal with the setting of control parameters. Ali and Törn [1] suggested to adapt the value of the scaling factor F within the search process. Zaharie [20] derived the critical interval for the control parameters. Teo [11, 12] suggested a self-adaptive mechanism changing population size during the search process. Some other attempts to the adaptation of DE control parameters are summarized in Liu and Lampinen [6].

3 Adaptive Variants of DE for Experiments

Evolutionary self-adaptation of control parameters F and CR was proposed by Brest et al. [3]. The values of F and CR are initialized randomly for each point in population and survive together with the individuals, but they can be changed randomly in each generation with respective probabilities τ_1, τ_2. New values of F are distributed uniformly in $[\,F_l,\, F_u]$ and new values of CR are also uniform random values $\in [\,0,\, 1]$.

The competitive setting of the control parameters was introduced by Tvrdík [14]. Let us have H different settings of F and CR used in (3), (1), and (2) and choose among them at random with probabilities $q_h,\ h = 1, 2, \ldots, H$. The probabilities can be adjusted according to the success rate of the settings in the preceding steps of search process. The h-th setting is successful, if it generates such a trial point $\boldsymbol{y}$ that $f(\boldsymbol{y}) < f(\boldsymbol{x}_i)$. When n_h is the current number of the h-th setting successes, probability q_h is evaluated as the relative frequency

$$q_h = \frac{n_h + n_0}{\sum_{j=1}^{H}(n_j + n_0)}\ , \tag{4}$$

where $n_0 > 0$ is a constant. The setting of $n_0 > 1$ prevents a dramatic change in q_h by one random successful use of the h-th parameter setting. To avoid process's degeneration, the current values of q_h are reset to their starting values $q_h = 1/H$, if any probability q_h decreases bellow a given limit $\delta > 0$. Several variants of such competitive differential evolution were implemented and tested on a benchmark in [16] and two best performing variants are included into the Matlab program library [17] for free use.

The mutation according to (1) or (2) does not guarantee that a new trial point $\boldsymbol{y} \in D$. In such a case either the point $\boldsymbol{y}$ can be skipped and a new one generated or if any $y_j < a_j$ or $y_j > b_j$, all such coordinates y_j can be reversed via flopping into D over the a_j or b_j by mirroring. The latter method is used in algorithms implemented for numerical tests.

4 Nonlinear Regression Model and Estimation of its Parameters

The additive nonlinear regression model is formed as follows:

$$Y_i = g(\boldsymbol{z}_i, \boldsymbol{\beta}) + \varepsilon_i, \quad i = 1, 2, \ldots, n\,, \tag{5}$$

where Y_i are random variables, $\boldsymbol{z}_i^T = (z_1, z_2, \ldots, z_k)$ are rows of regressor matrix $\boldsymbol{Z}$, $\boldsymbol{\beta}$ is vector of unknown parameters, g is a given function nonlinear in parameters, and all ε_i are independent and identically distributed (i.i.d.) random variables with zero mean values. The estimation of parameters by the least squares method means to find such estimates $\hat{\boldsymbol{\beta}} = (\hat{\beta}_1, \hat{\beta}_2, \ldots, \hat{\beta}_d)^T$ that minimize the residual sum of squares $f(\hat{\boldsymbol{\beta}})$ given by the equation

$$f(\hat{\boldsymbol{\beta}}) = \sum_{i=1}^{n} \left[Y_i - g(\boldsymbol{z}_i, \hat{\boldsymbol{\beta}}) \right]^2 . \tag{6}$$

Unlike the residual sum of squares in linear regression models, the function $f(\hat{\boldsymbol{\beta}})$ is not convex in general. Due to this fact the estimation of the parameters in nonlinear regression models is the global optimization problem. Iterative deterministic algorithms used for the estimation of parameters in standard statistical packages like Levenberg-Marquardt or Gauss-Newton methods searching for a local minimum fail sometimes, see e.g. [13]. A specially tailored controlled random search algorithm with an adaptive stopping condition was proposed for the estimation of the parameters in nonlinear regression models [15]. Four local-search heuristics compete in this controlled random search algorithm, and the stopping condition is changed adaptively according to the value of determination index (often called R-squared), as achieved in the search. For details see [15], the implementation of this algorithm in Matlab is included into the Program Library [17] as well.

5 Experiments and Results

One of the criteria for examining the reliability of algorithm is the comparison of the computed values with the "certified values" from sources that are considered to be reliable. NIST collection of datasets [7] contains 27 nonlinear regression tasks categorized according to their level of difficulty (lower – 8 tasks, average – 11 tasks, and higher – 8 tasks). The nonlinear regression models in the collection are of exponential or rational types, the number of parameters ranging from 2 to 9. The total number of observations varies in a wide range, from 6 to 250. The details about the models are available at NIST web site [7]. The certified values are reported to 11 decimal places for each task. The accuracy of the experimental results obtained by the algorithms is evaluated according to the number of duplicated digits when compared with the certified results. The

number of duplicated digits λ can be calculated via log relative error [9] as

$$\lambda = \begin{cases} 0 & \text{if } \frac{|m-c|}{|c|} \geq 1 \\ 11 & \text{if } \frac{|m-c|}{|c|} < 1 \times 10^{-11} \\ -\log_{10}\left(\frac{|m-c|}{|c|}\right) & \text{otherwise}, \end{cases} \tag{7}$$

where $c \neq 0$ denotes the certified value and m denotes the computed value. According to [7], excluding the cases where the certified value is essentially zero (e.g. for the three Lanczos problems), a good nonlinear least squares procedure should be able to duplicate the certified results to at least 4 digits.

Six algorithms were compared in numerical experiments, namely the controlled random search with competing local-search heuristics and adaptive stopping condition tailored for the estimation of regression parameters [15], denoted CRS4e in the text, and five self-adaptive variants of differential evolution. The same adaptive stopping condition was used in all algorithms and tests that were carried out on all 27 NIST nonlinear regression tasks [7]. The algorithms were tested in 100 repeated runs for each task. Such number of repetitions ensures that the asymptotical confidence intervals for the probability of stopping at a good approximation of the global minimum point are acceptably narrow (about 20% or less in the case of 95% confidence interval). Domains D for all the tasks are given in [15], as well as the values of control parameters for CRS4e.

Population size for all the DE variants in experiments was set to $N = \max(20, 5\,d)$. Among adaptive variants of DE in testing there were four variants with the competition of various (F, CR) settings. Parameters for the control of competition were set to $n_0 = 2$, and $\delta = 1/(5\,H)$ in all tasks. These competitive variants of DE are denoted as follows:

- DER9 uses 9 settings from all combinations of values ($F = 0.5$, $F = 0.8$, $F = 1$) and ($CR = 0$, $CR = 0.5$, $CR = 1$). The vector $\boldsymbol{y}$ is generated by the standard DE/rand/1/bin strategy according to (1) and (3),
- DER9rl employs the same 9 settings of F and CR like DER9, but the mutant vector $\boldsymbol{u}$ is generated by using random localization after [4] described in Section 2,
- DEBR18 with 18 settings (a pair of nine combinations F and CR used in DER9). For nine settings the trial vector $\boldsymbol{y}$ is generated by the standard DE/rand/1/bin strategy, and for the other nine settings by the DE/best/2/bin strategy according to (2) and (3),
- DEBR18rl uses 18 settings of F and CR like DEBR18, but with random localization in DE/rand/1/bin strategy.

The last variant of adaptive DE denoted BREST was implemented after [3] with control parameters set up to the values recommended by authors: $\tau_1 = 0.1$, $\tau_2 = 0.1$, $F_l = 0.1$, and $F_u = 0.9$.

The results of the numerical comparison for each of 27 tasks are presented in Table 1. The time requirement of the task is expressed by the average number of the objective function evaluations (ne) needed to reach the stopping condition.

Table 1. Comparison of algorithms in estimation of nonlinear-regression parameters (tasks are ordered in groups from low to higher level of difficulty)

Algorithm	CRS4e		DEBR18		DEBR18rl		DER9		DER9rl		BREST	
Task	*ne*	*RP*	*rne*	*RP*	*rne*	*RP*	*rne*	*RP*	*rne*	*RP*	*rne*	*RP*
Chwirut1	1945	100	80	100	41	100	90	100	27	100	207	100
Chwirut2	1934	100	73	100	41	100	90	100	25	100	201	100
DanWood	1173	100	98	100	68	100	89	100	41	100	242	100
Gauss1	9133	100	104	100	46	100	82	100	18	100	87	100
Gauss2	9452	98	134	79	55	78	98	79	20	76	144	88
Lanczos3	30347	100	95	99	60	100	378	99	277	93	675	5
Misra1a	1794	100	110	99	71	99	139	100	96	97	463	95
Misra1b	1512	100	104	100	72	100	171	99	90	99	379	100
ENSO	13375	86	144	100	83	100	111	100	38	99	83	99
Gauss3	10391	99	124	100	39	100	104	100	21	99	273	100
Hahn1	12267	93	39	90	-2	83	49	96	1	85	548	92
Kirby2	6532	100	34	100	-5	100	52	100	-9	100	350	100
Lanczos1	209448	100	-57	99	-68	100	-21	100	-36	100	14	1
Lanczos2	30463	100	96	100	54	100	357	100	253	100	679	7
MGH17	9045	100	40	100	14	100	135	100	77	100	486	100
Misra1c	1868	100	110	100	72	100	284	97	125	97	833	88
Misra1d	1794	100	109	100	69	100	284	99	118	99	736	91
Nelson	4876	100	100	100	67	100	320	94	172	96	536	100
Roszman1	3379	100	22	100	-9	100	13	100	-23	100	163	100
Bennett5	36585	100	103	96	104	91	174	2	135	1	216	1
BoxBOD	828	100	120	100	89	100	99	100	58	100	172	100
Eckerle4	1705	100	47	100	25	100	34	100	5	100	29	100
MGH09	8823	100	38	100	16	100	120	100	71	100	429	100
MGH10	21025	100	90	100	79	100	436	74	340	100	458	9
Rat42	1921	100	72	100	32	100	56	100	8	100	176	100
Rat43	2941	100	28	100	-6	100	18	100	-21	100	151	100
Thurber	9671	100	50	100	2	100	53	100	0	100	475	100
Average		99.1	78	98.6	41	98.2	141	94.0	71	94.1	341	80.6

For easier comparison and better readability, columns denoted *rne* contain the relative changes of *ne* in percents when compared with CRS4e. Therefore, the negative value of *rne* means less time consumption with respect to CRS4e, the positive value means a higher one, the value $rne = 100$ means twice greater time demand in comparison with CRS4e. The reliability of the search in each run was evaluated by the number of duplicated digits in residual sum of squares (6). The percentage of runs with $\lambda > 4$ (except for Lanczos1 task, where the value of objective function in the global minimum point is essentially zero, the percentage of runs with $\lambda > 2.4$ is reported) is given in the columns *RP*. As it is shown in Table 1, the tested variants of DE do not outperform the CRS4e algorithm either in the average reliability or in the average convergence. However, the average reliability of DEBR18 and DEBR18rl does not differ significantly from

the reliability of CRS4e. The convergence of DEBR18rl is only slightly slower than CRS4e, with increase of time demands 41% in average. All DE variants used in the tests outperform the specialized CRS4e algorithm in the reliability on the ENSO task, but this better reliability is paid by slower convergence.

BREST variant of DE shows the worst performance in the estimation of nonlinear regression parameters among all the algorithms in tests. This is unexpected finding because in numerical comparison of the same algorithms on the six composite functions [5], BREST variant achieved very good performance [18].

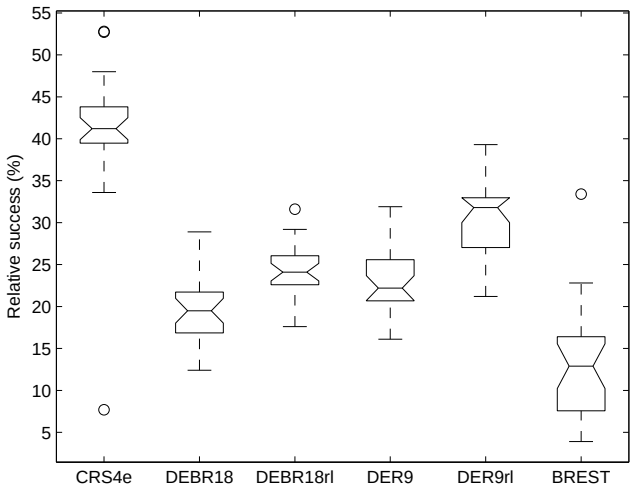


Fig. 1. Boxplots of average relative success in generating the new trial point $\boldsymbol{y}$

The best convergence of CRS4e can be explained by its highest relative success in generating the new trial point $\boldsymbol{y}$. Boxplots of average relative success are displayed for all tested algorithms in Fig. 1. The boxplots are based on 27 averages of relative success computed from 100 runs for each task. The values of the relative success for all the variants of DE are significantly less than for CRS4e. Median of the relative success for CRS4e is approximately 42%, which is higher value than maximum of all the DE variants. When comparing rl-variants of DE with their standard counterparts, random localization increases the relative success and in consequence improves convergence as well, but the average relative success of rl-variants remains significantly smaller than the relative success of CRS4e.

Variabilities in number of function evaluations needed to reach the stopping condition are compared in Fig. 2. The variability is evaluated by the coefficient of variance vc, defined by expression $vc = 100 \times stdev\,/\,mean$. The boxplots are also based on average coefficients of variability for the tasks like in the case of

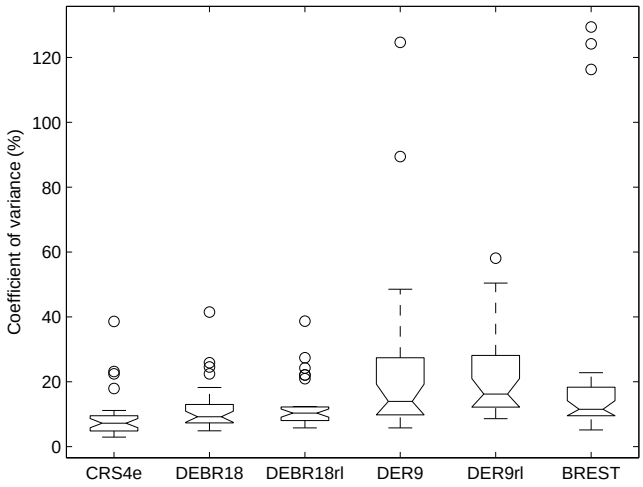


Fig. 2. Variability in number of function evaluation – coefficients of variability

previous figure. As it is seen in the boxplots, the less reliable algorithms have also higher variability of time demands, mostly with several heavy outliers.

6 Conclusions

Adaptive variants of differential evolution intended as general heuristics for the box-constrained optimization problems were compared experimentally with CRS4e algorithm tailored for the estimation of parameters in nonlinear regression models. We can conclude from the numerical results that the algorithm tailored for a certain class of problems can outperform the other algorithms.

Good performance of the CRS4e algorithm is achieved due to high success rate in generating of the new trial point $\boldsymbol{y}$, about 40% or higher for most tasks. This success rate is approximately twice greater than 20% used for correction of control parameters in evolutionary strategy ("the rule of one fifth", for details see e.g. [2]). This high success rate achieved by CRS4e is a bit risky and can result in premature convergence, particularly if the algorithm is used for the functions of different shape than the algorithm was tailored. General adaptive heuristics like adaptive variants of differential evolution are supposed to perform better on such functions.

The worst preforming BREST algorithm in the estimation of nonlinear-regression parameters is the best on the composite functions [5] and vice versa, the CRS4e is the worst on these composite functions, see the experimental results in [18]. Although such finding seems paradoxical, it is in agreement with the implications resulting from the No Free Lunch Theorems [19] stating that no stochastic search algorithm can outperform others for all objective functions.

References

1. Ali M. M., Törn A.: *Population set based global optimization algorithms: Some modifications and numerical studies,* Computers and Operations Research **31** (2004) 1703–1725.
2. Bäck T.: *Evolutionary Algorithms in Theory and Practice,* Oxford University Press, New York (1996).
3. Brest J., Greiner S., Boškovič B., Mernik M., Žumer V.: *Self-Adapting Control Parameters in Differential Evolution: A Comparative Study on Numerical Benchmark Problems,* IEEE Transactions on Evolutionary Computation **10** (2006) 646–657.
4. Kaelo P., Ali M. M.: *A numerical study of some modified differential evolution algorithms,* European J. Operational Research, **169** (2006) 1176–1184.
5. Liang J. J., Suganthan P. N., Deb K.: *Novel Composition Test Functions for Numerical Global Optimization,* IEEE Swarm Intelligence Symposium, (2005) 68–75. Matlab codes available at `http://www.ntu.edu.sg/home/EPNSugan/index_files/comp-functions.htm`
6. Liu J., Lampinen J.: *A Fuzzy Adaptive Differential Evolution Algortithm,* Soft Computing **9** (2005) 448–462.
7. *NIST. Statistical Reference Datasets. Nonlinear regression,* NIST Information Technology Laboratory. `http://www.itl.nist.gov/div898/strd/` (2001).
8. Price K. V., Storn R., Lampinen J.: *Differential Evolution: A Practical Approach to Global Optimization,* Springer-Verlag (2005).
9. McCullough B. D., Wilson B.: *On the accuracy of statistical procedures in Microsoft Excel 2003,* Comput. Statist. and Data Anal. **49** (2005) 1244 – 1252.
10. Storn R., Price K. V.: *Differential Evolution—a Simple and Efficient Heuristic for Global Optimization over Continuous Spaces,* J. Global Optimization **11** (1997) 341–359.
11. Teo J.: *Differential evolution with self-adaptive populations. Knowledge-Based Intelligent Information and Engineering Systems,* 9th International Conference, KES 2005, Melbourne, Australia, September 14-16, 2005, Proceedings, Part I, Lecture Notes in Computer Science **3681** (2005) 1284–1290.
12. Teo J.: *Exploring Dynamic Self-adaptive Populations in Differential Evolution,* Soft Comput. **10** (2006) 673–686.
13. Tvrdík J., Křivý I.: *Comparison of algorithms for nonlinear regression estimates,* In: Antoch J. (ed) COMPSTAT 2004. Physica-Verlag, Heidelberg New York, (2004) 1917–1924.
14. Tvrdík J.: *Competitive Differential Evolution,* MENDEL 2006, 12th International Conference on Soft Computing (Matoušek R. and Ošmera P. eds). University of Technology, Brno (2006) 7–12.
15. Tvrdík J., Křivý I., Mišík L.: *Adaptive Population-based Search: Application to Estimation of Nonlinear Regression Parameters,* Computational Statistics and Data Analysis (2007) doi 10.1016/j.csda.2006.10.014 (in print).
16. Tvrdík J.: *Differential Evolution with Competitive Setting of its Control Parameters,* TASK Quarterly **11** (2007) 169–179.
17. Tvrdík J., Habiballa H., Pavliska V.: *Matlab Program Library for Box-Costrained Global Optimization,* APLIMAT 2007, Part I, 6th International Conference on Applied Math. Slovak University of Technology, Bratislava (2007) 463–470, Program Library available on `http://albert.osu.cz/oukip/optimization/`
18. Tvrdík J.: *Adaptation in Differential Evolution: A Comparison on Composite Test Functions,* MENDEL 2007, 13th International Conference on Soft Computing. (Matoušek R. ed.). University of Technology, Brno (2007) 1–6.

19. Wolpert D. H., Macready W. G.: *No Free Lunch Theorems for Optimization,* IEEE Transactions on Evolutionary Computation **1** (1997) 67 – 82.
20. Zaharie D.: *Critical Values for the Control Parameter of the Differential Evolution Algorithms,* MENDEL 2002, 8th International Conference on Soft Computing (Matoušek R. and Ošmera P. eds). University of Technology, Brno (2002) 62–67.

Proceedings of the International Multiconference on Computer Science and Information Technology pp. 203–212
ISSN 1896-7094
© 2007 PIPS

Color Mining of Images Based on Clustering*

Łukasz Kobyliński and Krzysztof Walczak

Institute of Computer Science, Warsaw University of Technology
ul. Nowowiejska 15/19, 00-665 Warszawa, Poland

Abstract. The increasing size of multimedia databases and the ease of accessing them by a large number of users through the Internet carries a problem of efficient and semantically adequate querying of such content. A metadatabase may be used to shorten query resolution time by trying to limit the number of images being thoroughly analyzed to a smaller subset, having a high probability of finding the query image. In the article we propose a simple but fast and effective method of indexing such image metadatabases. The index is created by describing the images according to their color characteristics, with compact feature vectors, that represent typical color distributions. We present experiment results of typical search schemes by querying the metadatabase index created using a few different approaches.

Keywords: image retrieval, image clustering, multimedia indexing, data mining

1 Introduction

Querying of images by content is an important part of any multimedia mining system and it has been thoroughly studied in the literature. As the datasets available simultaneously to thousands of people in the world grow, a problem of not only accurate but also very efficient methods arises. It is now not only a question how to find an image most probably similar to the given example, but also how to perform this query in the global network, in enormous number of databases containing enormous numbers of images.

A possible approach to this problem is to divide it to smaller subproblems of indexing the individual databases and introduce a global metaindex, which enables to perform a preliminary selection between the databases, while resolving a given query. In the case of answering a query by example, the features of the query image are firstly calculated and the metaindex is queried to receive a list of databases most probably containing images similar to the one given. Then, the databases are queried in the given order to find the particular images, most similar to the example. This way, having an efficient method of indexing the databases and querying the resulting metaindex, the cost of performing a more thorough search may be reduced significantly by reducing the problem domain.

* The research has been partially supported by grant No 3 T11C 002 29 received from Polish Ministry of Education and Science.

In this paper we propose to use the Binary Thresholded Histogram (BTH) to calculate the image features while creating the metadatabase index. It is a color mining based approach to performing image queries, which has a very low cost of computation and adequate effectiveness for distinguishing databases having varied content. The BTH was created having the most compact representation of an image's color features in mind, while retaining enough information to be sufficient for differentiating of images. It then seems to be a very good candidate as a basis for creating the metadatabase index, enabling to perform very efficient, preliminary search in a set of databases.

The rest of the paper is organized as follows: in Section 2 we present previous work regarding the creation and indexing of metadatabases and content-based image retrieval. In Section 3 we describe the database setup that will be used in further discussion. In Section 4 we give the details of our approach to image representation, which is then used in query processing. In Section 5 we describe the employed method of processing a given query. Section 6 presents experimental results of database selection and Section 7 closes with a conclusion and discussion on possible enhancements.

2 Previous Work

The concept of employing a metadatabase index to facilitate querying of image databases has been proposed in [2]. The authors presented the use of a metadatabase to efficiently select the most appropriate databases while processing a query. Two selection strategies have been proposed, which maximize the probability of finding an image similar to a template associated with the query image.

The idea of Binary Thresholded Histogram has been presented in [6]. It is an efficient method of color-based retrieving of images, which uses 40 very compact, less then 32-bits long color feature vectors to find similarities between images.

Recent image mining and image retrieval surveys, relevant to the discussed subject, include [3] and [5]. Our previous work towards data mining in image databases, where the representation of images is based on color and texture features of regularly partitioned photographs has been presented in [4].

3 Database organization

There are fundamentally two different cases where a metadatabase index can be applied. Whenever a data can be thought of as a set of databases—individual collections of objects, which can be independently queried, these collections may contain semantically distinct or mixed content. In the first case objects of one semantic type exist only in one of the databases, whereas in the second they may appear in any number of datasets. On the other hand, a single database may consist of objects of only one semantic type or multiple types, without influencing the proposed method.

A video security monitoring system, consisting of multiple cameras pointed at different objects, each recording data separately, may be an example of the

first possible type of metadatabase application. Here, a problem may be stated as finding still images from any of the cameras, most similar to the given video frame. In this case only one of the databases containing photographs from distinct cameras should be thoroughly analyzed to find frames similar to the given query image, while the rest may be ignored. The aim of the metadatabase index is then to eliminate all irrelevant databases from the query and direct at the only one containing appropriate images.

In the web search engine research area there is a well known problem of routing queries to a distributed index of documents [1]. The problem, which may be given to illustrate the second possible case of data organization and the application of a metaindex structure, emerges from the fact, that distributed search engines need to have their index partitioned into multiple machines, using a method that guarantees the best possible load balancing across the servers. Many methods have been proposed for routing text search queries, but they can not be directly used in the case of performing image queries. Contrary to the first example, here the individual parts of the partitioned index concern subsets of images found on the Web by the search engine crawler and each of the subsets may contain images of different semantic types.

We have performed experiments with data simulating both stated possibilities, but the initial metaindex creation procedure is performed similarly, regardless of the type of data being processed. In our approach the metaindex consists of a very compact representation of each of the individual databases, created on the basis of color feature values of the images contained in the databases. A given database is firstly processed to calculate the features of each of the images and then clustered to obtain the set of the most significant feature values. The database representation consists of features of cluster centroids found in the database, along with the information of the number of images contained in each of the clusters and the mean and variance values of their similarity to the cluster centroids.

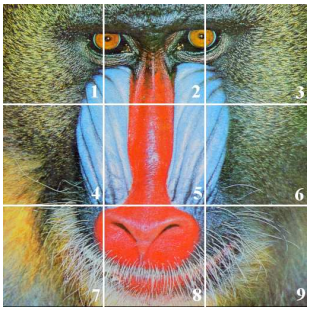
4 Image representation

We have used the Binary Thresholded Histogram approach, described in [6], to calculate and compare the color features of the images. The choice was made on the premise that a very efficient method, both in calculation and performing comparisons, must be used when creating the metadatabase index, to allow quick preliminary differentiation between available subsets of data.

4.1 Calculating the Binary Thresholded Histogram

As the BTH is calculated in the HSV color space, the first step in obtaining feature values is to convert a given image from RGB to HSV color representation. Secondly, the image is uniformly divided into rectangular blocks, by applying a $m \times n$ grid, where m is the number of rows and n is the number of columns. Next, color histograms of resulting blocks are calculated, with the chosen number

of bins set for each of the HSV components. This way each of the images is represented by $m \times n$ feature vectors consisting of $h \times s \times v$ values. Finally, the values are converted to binary by thresholding, with a given minimum $t\%$ threshold. Thus any color range appearing in the image subblock on more than $t\%$ of the pixels is denoted with a logical 1, while other colors are denoted with a 0. A logical 1 in the feature vector is meant to indicate that a particular color range is sufficiently represented in the image region to be noticed by a human. Consequently, the setting of the threshold value should be made on the basis of a human reception of the photographs and we have experimentally chosen a value of 4%. It is possible to tune the BTH to a given "quality ratio" by changing the representation bit length.



i	BTH	dist(1, i)
1	1 1 0 1 1 0 0 1 0 0 0 0	0
2	1 1 1 1 1 0 1 1 0 0 0 0	2
3	1 1 0 1 1 0 1 1 0 0 0 0	1
4	1 1 0 1 1 0 0 1 0 0 0 0	0
5	0 1 1 0 0 0 1 1 0 0 1 0	6
6	1 1 0 1 1 0 0 1 0 0 0 0	0
7	1 1 0 1 1 0 1 1 0 0 0 0	1
8	1 1 1 0 0 0 1 0 0 1 1 0	7
9	1 1 0 1 1 0 1 1 0 0 0 0	1

Fig. 1. A BTH representation of the *chimp* photograph and a few sample distance measure values between particular image tiles. The BTH threshold is set at $t = 4\%$ and $h = 4$, $s = 3$, $v = 1$ in this example.

This color feature representation approach is insensitive to small image translations, rotations, color shifts and brightness/contrast changes. The sensitivity depends on the set threshold value, chosen size of image tiles and the number of histogram color bins. For example in the case of translation, the representation will remain unchanged unless too many pixels are to be moved between neighboring tiles.

The binary representation allows for very quick comparisons, by performing a bitwise sum modulo 2 of any two feature vectors. Since each of the images is divided into $m \times n$ parts, a comparison of two images is accomplished by summing $m \times n \times h \times s \times v$ bits describing them. Formula 1 presents the calculation of the distance function between two images, for which histograms $x^{1 \ldots mn}$ and $y^{1 \ldots mn}$ are given.

$$\mathrm{d_{BTH}}(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{mn} \sum_{j=1}^{hsv} (x_j^i \oplus y_j^i). \tag{1}$$

Based on the distance, we can introduce a normalized similarity function between images $\mathbf{x}$ and $\mathbf{y}$: $\mathrm{sim}(\mathbf{x}, \mathbf{y}) = 1 - \frac{\mathrm{d_{BTH}}(\mathbf{x},\mathbf{y})}{mnhsv}$

Figure 1 presents an example of a BTH representation of an image divided into nine parts. A simple assessment of the method may be performed by comparing the distance measure values between image tiles (given as the rightmost table column) and the visual similarity of the tiles.

4.2 Database clustering

Since the image representation has a form of a binary vector, an adequate clustering algorithm must be employed to process such data, taking into account its relatively high dimensionality and computation time constraints. We have chosen to compare the CLOPE algorithm [7], which has been developed for effective clustering of transactional data and the well-known k-Means method with different distance measures.

The CLOPE algorithm relies on a concept of calculating simple parameters of cluster histograms and finding such clusterings, which optimize these parameters. This is accomplished by maximizing a global criterion function:

$$\text{Profit}(\mathbf{C}, r) = \frac{\sum_{i=1}^{k} \frac{S(C_i)}{W(C_i)^r} |C_i|}{\sum_{i=1}^{k} |C_i|}, \tag{2}$$

where $\mathbf{C} = \{C_1, \ldots, C_k\}$ is a given clustering, $\mathrm{S}(C)$ is the sum of item occurrences and $W(C)$ is the number of distinct items in the cluster C. The number of transactions in a cluster is denoted by $|C|$. By defining $S(C)$ as the size, $W(C)$ as width and $H(C) = S(C)/W(C)$ as the height of a cluster C, we see that $H(C)/W(C) = S(C)/W(C)^2$. The idea of the algorithm is then to maximize the ,,height" to ,,width" ratio of the clusters' histograms, with a given input parameter r called repulsion, which allows to change the level of intra-cluster similarity.

Apart from the CLOPE algorithm we have also performed experiments with the k-Means clustering method, using several different distance measures, namely the BTH distance given by Formula 1, the city-block distance and the Euclidean distance. The city-block distance, given by the formula:

$$\mathrm{d_M}(\mathbf{h}, \mathbf{g}) = \sum_{i=0}^{N-1} |h[i] - g[i]| \tag{3}$$

is normally equal to the distance measure proposed for BTH. The results are only different when we allow for non-binary values in the feature vector in intermediate calculations, as is the case in the vector averaging step performed in k-Means algorithm. We use only logical 0's and 1's in the case of BTH distance measure, which uses a bitwise exclusive-OR operation.

4.3 Cluster description

A few statistical parameters are calculated in the clustered database, to roughly assess the similarity distributions of images in the databases to the cluster centroids. A mean and variance of similarities to a given centroid are calculated and,

along with centroid's feature values and the number of images in the cluster, are included in the metadatabase index.

$$\mathrm{mean}(db, t, l_t) = \frac{\sum_{i \in db \wedge \mathrm{sim}(i,t) > l_t} \mathrm{sim}(i, t)}{\mathrm{num}(db, t, l_t)}, \tag{4}$$

$$\mathrm{var}(db, t, l_t) = \frac{\sum_{i \in db \wedge \mathrm{sim}(i,t) > l_t} (\mathrm{sim}(i, t) - \mathrm{mean}(db, t, l_t))^2}{\mathrm{num}(db, t, l_t) - 1}, \tag{5}$$

where $\mathrm{sim}(i, j)$ is a similarity measure between images i and j, $0 \leq \mathrm{sim}(i, j) \leq 1$ and $\mathrm{num}(db, t, l_t)$ is the number of images in database db with similarity greater than l_t to cluster centroid t.

Figure 2 presents the overall flow of database processing and the content of metadatabase index. At first the BTH feature values of the images in each of the databases are calculated. Having the representation in the form of binary vectors, a selected method of clustering is employed to form a chosen number of clusters. The feature values of cluster centroids are then used as the description of respective databases. The size of each cluster and the mean and variance of similarities between cluster content and cluster centroids are also included in the index.

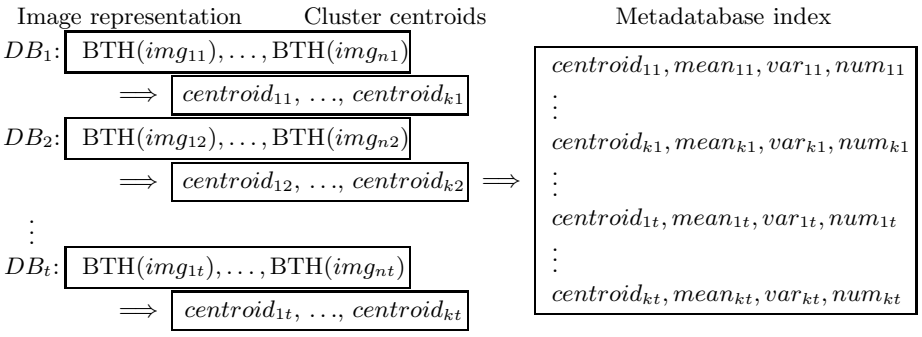


Fig. 2. An illustration of database processing and metadatabase index creation. The index is built on the basis of t databases, each containing a number of images belonging to different (or the same) semantic categories, for which BTH representation is calculated. The clustering step results in k cluster centroids for each of the databases. The index contains $k \times t$ centroid feature values, sizes of each cluster, as well as mean and variance values of distance measures of cluster elements to respective centroids.

5 Query execution

In a typical query by example scenario, a given image is firstly analyzed to calculate its features. This step is performed similarly, as described earlier in the case of database indexing. The feature values are then used to find the most

similar cluster centroids in the metadatabase index and then the corresponding databases, which satisfy certain conditions, depending on an employed database selection approach. Two such methods have been proposed in [2], of which we have used the mean-based approach in our experiments. The set D_i of relevant databases for the query q_i is chosen on the following criteria:

$$D_i = \{db \in DB \mid \text{num}(db, t, l_t) \geq v \ \wedge \ [x_t^{db}, y_t^{db}] \subseteq [a_{qi,t}, b_{qi,t}]\}, \tag{6}$$

where DB is the set of all databases, $[a_{qi,t}, b_{qi,t}]$ is a chosen similarity range of the query q_i to centroid cluster t within which the search is being conducted, $[x_t^{db}, y_t^{db}]$ is a calculated confidence interval, within which the mean similarity of images to centroid t is estimated to fall with a 99 percent chance and v is the minimum number of images similar to t in a database db for the database to be chosen.

The similarity range is defined by applying a given offset δ_q to the similarity measure value, such that:

$$a_{q_i,t} = \text{sim}(q_i, t) - \delta_q, \tag{7}$$

$$b_{q_i,t} = \text{sim}(q_i, t) + \delta_q. \tag{8}$$

The estimation of similarity mean value in a database db is performed by calculating a confidence interval at the 99% level ($\lambda = 0.01$): $x_t^{db} = \text{mean}(db, t, l_t) - E$, $y_t^{db} = \text{mean}(db, t, l_t) + E$, where:

$$E = z_{\lambda/2} \sqrt{\frac{\text{var}(db, t, l_t)}{\text{num}(db, t, l_t)}}. \tag{9}$$

6 Experimental Results

Given a set of ca. one thousand images of such categories as mountains, trees, flowers, people and buildings we have generated two series of datasets, one with images of a single original category included only in one of the possible databases and the second with the categories mixed randomly between the databases. The number of generated databases has been limited by the number of categories present in the test set.

The experiments have been conducted by taking one of the images from the set randomly and performing a query by example. In the case of applying the method to a set with original categories in only one of the databases, the quality of selection can be measured directly - by comparing the category label of the query image and the category of the selected database. In the second case of having a mixed content in each of the databases the quality is measured by precision and recall factors. This allows to assess the accuracy of the returned list of databases most probably containing images similar to the given sample. By denoting the number of relevant databases by A, the number of retrieved databases by B and the number of retrieved relevant databases by C we calculate precision as $P = C/B$ and recall as $R = C/A$.

The following parameter values have been set when performing the experiments: images partitioned into 3×3 blocks ($m = n = 3$), BHT with 64 color bins ($h = 8$, $s = 4$, $v = 2$), BHT threshold $t = 4\%$, similarity threshold $l_t = 0.5$, minimum number of similar images per database $v = 10$.

Table 1 presents a comparison of performance of the clustering methods during the preliminary database indexing phase. The results of clustering using the k-Means method are the averages of ten repeated experiments, as there is a random factor involved in the algorithm when selecting the starting cluster centers. The aim of the clustering is to find representative subgroups in each of the databases, from which distribution parameters are to be extracted and used in the metadatabase index. The clusters should thus be ,,pure" – ideally containing only one category of images. To assess the clustering performance we have used an entropy measure, calculated for each of the created clusters:

$$E_i = \sum_j -\frac{c(i,j)}{n_i} \log \frac{c(i,j)}{n_i}, \tag{10}$$

where $c(i, j)$ is the number of images in cluster i having a class label j and n_i is the number of all images in cluster i. The overall clustering entropy is calculated as the weighted sum of the individual cluster entropies: $E = \sum_i E_i n_i / \sum_i n_i$.

The CLOPE algorithm is an efficient clustering method, but not as accurate in performing BTH image representation clustering as the k-Means algorithm. It creates clusters with high mean similarity of the feature vectors, but also higher variance, which results in higher overall entropy of the clustering. The higher entropy of clustering using the BTH distance over the city-block distance metric is caused by loss of information while rounding non-integer values to perform the exclusive-or operation.

Table 1. Clustering quality of a mixed dataset with images of a) 3, b) 6 and c) 9 semantic categories

clustering method	number of clusters	similarity mean	similarity variance	clustering entropy
a) CLOPE ($r = 1.2$)	4	0.9528	0.047222	1.0627
k-Means, BTH distance	3	0.9177	0.006013	0.6593
k-Means, Euclidean distance	3	0.9174	0.008626	0.5690
k-Means, city-block distance	3	0.9172	0.006264	0.5579
b) CLOPE ($r = 1.3$)	7	0.9473	0.039157	1.6706
k-Means, BTH distance	6	0.9263	0.009208	1.0513
k-Means, Euclidean distance	6	0.9256	0.010810	0.8342
k-Means, city-block distance	6	0.9220	0.007342	0.8853
c) CLOPE ($r = 1.6$)	9	0.9675	0.039062	1.8656
k-Means, BTH distance	9	0.9291	0.012842	1.2935
k-Means, Euclidean distance	9	0.9268	0.015042	0.8890
k-Means, city-block distance	9	0.9208	0.007643	1.1133

The results of the query by example queries are presented in Table 2 and Table 3. In the case of databases with uniform content the purpose of the system is to return a ranked list of possible matches for specified query. To illustrate the performance of the proposed method two parameters have been included in the comparison of results for multiple similarity ranges (δ_q values, see Formula 7 and 8) of queries. The m_r value is the mean rank in the sorted list of results returned by the algorithm of the database that contains images of the same category label as the query image. Ideally, this value should be equal to 1 (correct database returned as the first in the list) to eliminate all irrelevant databases from further search. The p value, given as a percentage of all queries, is the part of queries that didn't match any database in the metaindex. The accuracy of selecting an appropriate database proved to increase when increasing the number of subdivisions of the databases from $k = 3$ to $k = 6$ clusters. This results in increasing the amount of information available in the metaindex and facilitates the query resolution.

Table 2. Metadatabase query by example results. Uniform databases case. Clustering with a) $k = 3$ clusters ($r = 1.2$ for CLOPE) and b) $k = 6$ clusters ($r = 1.5$).

clustering	$\delta_q = 0.04$		$\delta_q = 0.05$		$\delta_q = 0.06$		$\delta_q = 0.07$		$\delta_q = 0.08$	
method	m_r	$p\%$	m_r	$p\%$	m_r	$p\%$	m_r	$p\%$	m_r	$p\%$
a) CLOPE	2.88	21.71	3.74	7.34	3.66	1.94	3.62	2.88	3.88	-
BTH	2.23	9.00	2.48	6.48	2.3	0.98	3.54	-	3.13	-
Euclidean	2.40	16.53	2.90	9.01	2.51	0.98	3.03	-	4.12	-
City-block	2.65	10.62	2.46	1.94	2.81	2.88	3.31	-	3.75	-
b) CLOPE	2.67	38.42	3.02	32.22	3.45	10.62	3.59	5.61	3.48	1.94
BTH	2.06	26.81	1.95	12.17	2.27	4.72	2.48	0.98	2.69	-
Euclidean	2.45	17.89	2.20	12.93	2.48	4.72	2.84	1.94	2.74	-
City-block	1.96	12.93	1.98	-	2.27	-	2.11	-	2.71	-

The mean results of queries performed in the databases of a mixed content are presented as precision and recall values for three different δ_q values. While narrowing the similarity ranges of the queries results in a better precision value, the recall indicates that at most two-thirds of relevant databases are returned. In most applications maximizing the recall value is a priority and it may be achieved by increasing δ_q to at least 0.08 with only a small loss of precision.

7 Conclusions

In this article we have proposed an application of Binary Thresholded Histogram (BTH), a color feature description method, to the creation of a metadatabase index of multiple image databases and conducted experiments comparing several clustering approaches for this application. The BTH, despite being a very rough and compact representation of image colors, proved to be an adequate method

Table 3. Metadatabase query by example results. Mixed databases case. Clustering with a) $k = 3$ clusters ($r = 1.2$ for CLOPE) and b) $k = 6$ clusters ($r = 1.3$).

clustering method	$\delta_q = 0.04$		$\delta_q = 0.06$		$\delta_q = 0.08$	
	precision	recall	precision	recall	precision	recall
a) CLOPE	0.6157	0.4201	0.6456	0.6145	0.6376	0.8408
BTH	0.6160	0.6057	0.6117	0.8877	0.5936	0.9793
Euclidean	0.5892	0.6731	0.5694	0.9237	0.5990	0.9933
City-block	0.6369	0.6425	0.5883	0.8943	0.5912	0.9843
b) CLOPE	0.5671	0.5319	0.5730	0.7348	0.5902	0.9256
BTH	0.6257	0.5208	0.6271	0.7913	0.6122	0.9713
Euclidean	0.6287	0.4798	0.6367	0.8418	0.5924	0.9600
City-block	0.6543	0.5671	0.6210	0.8073	0.6047	0.9627

of describing the characteristics of image databases and creating a metadatabase index for querying large amounts of data. While the approach does not provide very high precision in returning the set of relevant databases, it still significantly reduces the final query execution time by limiting the number of databases to analyze. The parameters may be set to achieve 100% recall value, which guarantees that all relevant databases will be retrieved for each query.

References

1. Baeza-Yates R., Castillo C., Junqueira F., Plachouras V., Silvestri F.: *Challenges on Distributed Information Retrieval,* International Conference on Data Engineering (ICDE). Istanbul, Turkey, April 2007. ICEE CS Press.
2. Chang W., Sheikholeslami G., Wang J., Zhang A.: *Data Resource Selection in Distributed Visual Information Systems,* IEEE Transactions on Knowledge and Data Engineering, 10 (6): 926–946, 2001.
3. Datta R., Joshi D., Li J., Wang J. Z.: *Image Retrieval: Ideas, Influences, and Trends of the New Age.* ACM Computing Surveys, 2007.
4. Kobyliński Ł., Walczak K.: *Image Classification with Customized Associative Classifiers,* Proceedings of the International Multiconference on Computer Science and Information Technology. November 6–10, 2006. Wisła, Poland.
5. Lew M., Sebe N., Djeraba C., Jain R.: *Content-based Multimedia Information Retrieval: State-of-the-art and Challenges,* ACM Transactions on Multimedia Computing, Communication, and Applications, Vol. 2, No. 1, pp. 1–19, February 2006.
6. Walczak K.: *Image Retrieval Using Spatial Color Information,* Computer Analysis of Images and Patterns: 9th International Conference, CAIP 2001 Warsaw, 53–60.
7. Yang Y., Guan X., You J.: *CLOPE: a fast and effective clustering algorithm for transactional data,* ACM SIGKDD '02. July 23–26, 2002, Edmonton, Alberta, Canada.

Proceedings of the International Multiconference on
Computer Science and Information Technology pp. 213–223
ISSN 1896-7094
© 2007 PIPS

Using of a Graph of Action Dependencies for Plans Optimization

Lukáš Chrpa

Department of Theoretical Computer Science and Mathematical Logic
Faculty of Mathematics and Physics
Charles University in Prague
chrpa@kti.mff.cuni.cz

Abstract. There are a lot of planning techniques which prefer faster computation of plans instead of quality of these plans. However, in many cases we do not only need to have a plan which is computed quickly, but we need to have the plan of good quality as well. This paper presents a Graph of Action Dependencies as a tool which can be useful for plans optimization. This approach is very good for detecting pairs of inverse actions in plans or actions not needed to acquire a goal etc. Combination of this approach with existing planners should bring an improvement in acquiring more quality plans in a short time.

1 Introduction

Planning problems are an important branch of AI and they are very useful in many practical applications. However, many planning problems are hard to solve and many planners need to use heuristics for faster computation, but unfortunately at the cost of loss of the optimality of plans.

An early planner, called STRIPS [2], was very popular, because it generated plans very quickly. However, it was showed later that STRIPS did not always generate optimal plans [15]. Later, there were proposed many planning techniques for solving planning problems. In 1997, there was proposed an algorithm based on planning graph [1] which was much more efficient than other planners and brought a little revolution into planning. In 1998, there was introduced an International Planning Competition [10] which evoked an expansion of many planning techniques. Lot of these planning techniques were suboptimal which means that plans generated by these techniques were not optimal (for example FF [8] and LPG [5]). The main advantage of suboptimal planning systems rests in much faster generations of plans than in optimal planning systems. However, these plans are far from being optimal. This is the reason why it is needed to study the area of plans optimization, but there is not much work done yet. Results, published in [17], are very interesting, but the technique for the plans optimization described there seems to consume a lot of time.

A possible approach which can help with plans optimizations rests in plans structure analysis, especially in analysis or description of actions dependencies.

There are many of these approaches. One of them [4] refers about action languages as a formalism for talking about the effects of actions. This work describes a couple of action languages which are divided into two main classes (action description languages for describing the system and action query languages for making queries to the system). Another work [12] defines a language for expressing causal knowledge (previously studied at [3, 11]) and gives an approach of formalizing actions. The contribution of both of the works is mainly theoretical. More practical work [9] comes with landmarks, facts that must be true in every valid plan. This approach can evoke an improvement to the planners giving them more constraints in searching. One of the newest approaches [16], which is based on plan space planning techniques (more in [6]), gives us very good results in parallel planning. The importance of these results rests in a fact that an undermentioned structure of a Graph of Action Dependencies is partly related to plan space planning.

This paper presents a Graph of Action Dependencies which provides an information about plans structures. Usage of graphs in many kinds of optimization tasks seems to be very useful, because many problems can be solved much faster [14]. This paper also describes algorithms of plans optimization which run in a polynomial time and which can easily detect and remove many kinds of unnecessary actions (more precisely described in section 4). These algorithms can be very helpful with making plans more optimal without spending a lot of time. However, according to the complexity results [7], it is not possible to make the plan optimal in a polynomial time. This paper also presents a possibility of plans optimization through optimization of their subplans with a small length which in spite of hardness of this problem should not consume a lot of time (if the length on the subplans is 2 that the optimization can be computed in a polynomial time).

This paper is organized as follows. Next section will present some basic notions from planning problems. Section 3 will introduce a Graph of Action Dependencies, an algorithm of its construction and some basic theoretical aspects of it. Section 4 will present algorithms for plans optimization, especially removing of actions which are not needed to acquire a goal and pairs of inverse actions, and finally the ideas how to make plans optimization in general. Section 5 will present my future research plans and the last section will conclude.

2 Preliminaries

This section presents definitions and notions which are needed for better reading and understanding of this paper.

2.1 Planning problems

In this subsection, we introduce some basic definitions which are needed to understand basic notions frequently used in a connection to planning problems. To get deeper insight about planning problems, see [6].

Definition 1. *Assume that $L = \{p_1, \ldots, p_n\}$ is a finite set of predicates. Planning domain Σ over L is a 3-tuple (S, A, γ) where:*

- *$S \subseteq 2^L$ is a set of states. $s \in S$ is a state. If $p \in s$ then p is true in s and if $p \notin s$ then p is not true in s.*
- *A is a set of actions. Action $a \in A$ is a 3-tuple $(p(a), e^-(a), e^+(a))$ where $p(a)$ is a precondition of the action a, $e^-(a)$ is a set of negative effects of the action a and $e^+(a)$ is a set of positive effects of the action a and $e^-(a) \cap e^+(a) = \oslash$.*
- *$\gamma : S \times A \to S$ is a transition function. $\gamma(s, a) = (s - e^-(a)) \cup e^+(a)$ if $p(a) \subseteq s$.*

Definition 2. Planning problem P *is a 3-tuple (Σ, s_0, g) such that:*

- *$\Sigma = (S, A, \gamma)$ is a planning domain over L.*
- *$s_0 \in S$ is an initial state.*
- *$g \subseteq L$ is a set of goal predicates.*

Definition 3. Plan π *is an ordered sequence of actions $\langle a_1, \ldots, a_k \rangle$ such that, the plan π* solves *the planning problem P if and only if exist appropriate sequence of states $\langle s_0, \ldots, s_k \rangle$ such that $s_i = \gamma(s_{i-1}, a_i)$ and $g \subseteq s_k$. Plan π is* optimal *if and only if for each $\pi' \mid \pi \mid \leq \mid \pi' \mid$ is valid.*

2.2 Graph theory

In this paper, there are used some notions from the theory of graphs. These notions are usually well known and due to space reason the notions will not be defined here. To get deeper insight about the graph theory and graph algorithms, see [13].

3 Graph of action dependencies

This section describes what is the Graph of Action Dependencies and how can be constructed from the existing plan. There are also presented some theoretical aspects of the Graph of Action Dependencies.

3.1 Basic definitions

Before we describe the algorithm which constructs the Graph of Action Dependencies we must define notions which are needed to know what is the Graph of Action Dependencies.

Definition 4. *Let $\langle a_1, \ldots, a_n \rangle$ is an ordered sequence of actions. The action a_j is* straightly dependent on the action a_i *(denoted as $a_i \prec a_j$) if and only if $i < j$, $(e^+(a_i) \cup (p(a_i) \setminus e^-(a_i))) \cap p(a_j) \neq \emptyset$ and does not exist any $k_1, \ldots, k_l$ such that $i < k_1, \ldots, k_l < j$ and $(e^+(a_i) \cup (p(a_i) \setminus e^-(a_i))) \cap p(a_j) \subseteq \bigcup_{t=1}^{l} (e^+(a_{k_t}) \cup (p(a_{k_t}) \setminus e^-(a_{k_t})))$.*

Definition 4 told us that some action a_j is straightly dependent on some action a_i when the action a_i is performed before the action a_j and the action a_i is the last action which operates with some predicate (or predicates) which are in the precondition of a_j (the predicate or predicates must be true after performing the action a_1). It is clear that some action can be straightly dependent on more actions.

Definition 5. *Let $\pi = \langle a_1, \ldots, a_n \rangle$ is a plan which solves the planning problem $P = (\Sigma, s_0, g)$. Let $a_0 = (\emptyset, \emptyset, s_0)$ and $a_{n+1} = (g, \emptyset, \emptyset)$ are actions. Let $\prec$ is the relation of straight dependency over the ordered sequence of actions $\langle a_0, a_1, \ldots, a_n, a_{n+1} \rangle$.* Graph of action dependencies with respect to π and P *is a directed graph $G = (V, E)$ such that $V = \{v_0, v_1, \ldots, v_n, v_{n+1}\}$ and $(v_i, v_j) \in E$ if and only if $a_i \prec a_j$.*

The Graph of Action Dependencies can give us more information about structures of plans which will be explained later.

Definition 6. *Let $\langle a_1, \ldots, a_n \rangle$ is an ordered sequence of actions and a relation $\prec^*$ is a transitive closure of the relation of straight dependency $\prec$. The action a_j is* dependent on the action *a_i if and only if $a_i \prec^* a_j$.*

In the following text we will use the special actions a_0 and a_{n+1} defined in the same way as in definition 5. For illustration, the action a_0 is performed before the plan and the action a_{n+1} is performed after the plan.

3.2 Construction of the Graph of Action Dependencies

There will be presented an algorithm which constructs from some plan he Graph of Action Dependencies. Each predicate p gains an attribute $d(p)$ which refers to the last action which operates with them. Each edge (v_i, v_j), should contain an assignment of a set of such predicates p where $d(p) = i$ and $p \in p(a_j)$.

Algorithm 1:
`INPUT:` A plan $\pi = \langle a_1, \ldots, a_n \rangle$ which solves the planning problem $P = (\Sigma, s_0, g)$
`OUTPUT:` A Graph of action dependencies with respect to π and P

foreach predicate $p \in s_0$ **do** $d(p) = 0$ **endforeach**
$V = \{v_0\}$; $E = \emptyset$
for $i = 1$ **to** n **do**
  $V = V \cup \{v_i\}$
  compute a set $X = \{d(p) | \forall p \in s_{i-1} \cap p(a_i)\}$
  foreach $x \in X$ **do**
    $E = E \cup \{(v_x, v_i)\}$
    create an assignment for (v_x, v_i) which contains all predicates p where $d(p) = x$ and $p \in p(a_i)$
  endforeach
  $s_i = (s_{i-1} \setminus e^-(a_i)) \cup e^+(a_i)$
  foreach predicate $p \in s_i \cap (e^+(a_i) \cup (p(a_i) \setminus e^-(a_i)))$ **do** $d(p) = i$ **endforeach**
endfor

$V = V \cup \{v_{n+1}\}$
compute a set $X = \{d(p) | \forall p \in s_n \cap g\}$
foreach $x \in X$ **do**
  $E = E \cup \{(v_x, v_{n+1})\}$
  create an assignment for (v_x, v_{n+1}) which contains all predicates p where $d(p) = x$ and $p \in g$
endforeach

Theorem 1 (correctness of algorithm 1). *Let $\pi = \langle a_1, \ldots, a_n \rangle$ is a plan which solves the planning problem $P = (\Sigma, s_0, g)$. Algorithm 1 always constructs a graph $G = (V, E)$ which is the Graph of Action Dependencies with respect to π and P.*

Proof. It is clear that $V = \{v_0, v_1, \ldots, v_n, v_{n+1}\}$. Now we must prove that $(v_j, v_i) \in E$ if and only if $a_j \prec a_i$. From the algorithm 1 can be seen that $(v_j, v_i) \in E$ if and only if there exists some predicate $p \in s_{i-1} \cap p(a_i)$ whose attribute $d(p) = j$. It is clear that $0 \leq j < i$ because in i-th step of cycle there is no such predicate p' where $0 \leq d(p') < i$. If exist such predicate $q \in (s_{i-1} \cap p(a_i))$ whose attribute $d(q) = j$ then this attribute must be set to j in j-th step of cycle and $q \in (s_j \cap (e^+(a_j) \cup (p(a_j) \setminus e^-(a_j))))$. Now it is clear that $(e^+(a_j) \cup (p(a_j) \setminus e^-(a_j))) \cap p(a_i)$ is not empty and contains at least one predicate. It is also clear that does not exist any k such that $j < k < i$ and $q \in (e^+(a_k) \cup (p(a_k) \setminus e^-(a_k)))$. Now we can see that $a_j \prec a_i$. □

It is clear that the algorithm 1 runs at $O(n)$ steps. The algorithm 1 can also run simultaneously with a planner based on forward planning techniques.

3.3 Theoretical aspects of the Graph of Action Dependencies

This subsection is dedicated to basic theoretical aspects of the Graph of Action Dependencies which can help us finding out possible advantages of this approach.

Proposition 1. *A Graph of action dependencies $G = (V, E)$ with respect to $\pi = \langle a_1, \ldots, a_n \rangle$ and $P = (\Sigma, s_0, g)$ is acyclic and has a topological sort ord : $V \to \{0, \ldots, n+1\}$ such that $ord(v_i) = i$, $\forall i \in \{0, \ldots, n+1\}$.*

Proof. From the definition of topological sort we know that for all edges $(v_i, v_j) \in E$ we have $ord(v_i) < ord(v_j)$. From the assumption we know that $ord(v_i) < ord(v_j)$ if and only if $i < j$. From the definition 5 we know that $(v_i, v_j) \in E$ if and only if $a_i \prec a_j$ for some actions a_i and a_j. From the definition 4 we know that if $a_i \prec a_j$ than $i < j$. Now it is clear that *ord* is the topological sort of G and from this can be immediately seen that G is acyclic. □

Proposition 2. *Let $G = (V, E)$ is a Graph of Action Dependencies with respect to $\pi = \langle a_1, \ldots, a_n \rangle$ and $P = (\Sigma, s_0, g)$. There is a path from v_i to v_j in G if and only if $a_i \prec^* a_j$.*

Proof. It can be seen immediately from the definition 5 (i.e. $(v_k, v_l) \in E$ if and only in $a_k \prec a_l$) and from the fact that $\prec^*$ is a transitive closure of $\prec$. □

Proposition 3. *Let $G = (V, E)$ is a Graph of Action Dependencies with respect to $\pi = \langle a_1, \ldots, a_n \rangle$ and $P = (\Sigma, s_0, g)$. If there does not exits any action $a_i \in \pi$ such that $p(a_i) = \emptyset$ and also $g \neq \emptyset$ than $a_0 \prec^* a_j$ where $0 < j \leq n + 1$.*

Proof. From the definition 4 we know that if $p(a_i) \neq \emptyset$ than there must exist an action a_j such that $a_j \prec a_i$ and $j < i$. The proof can be done inductively. It is clear that a_1 must be straightly dependent (and also dependent) on a_0. Assume that $a_1, a_2, \ldots a_k$ are dependent on a_0. It is clear a_{k+1} must be straightly dependent on at least one action $a_0, a_1, \ldots a_k$. Then it is clear that a_{k+1} is also dependent on a_0. □

4 Optimizing plans using Graph of Action Dependencies

This section shows how can be the Graph of Action Dependencies used for the optimization of plans.

4.1 Removing of actions which are not needed to acquire goal

Any action a where $a \not\prec^* a_{n+1}$ is not needed to acquire the goal. According to the proposition 2 $a_i \not\prec^* a_{n+1}$ is satisfied if and only if there is no path from v_i to v_{n+1} in the Graph of Action Dependencies. To detect and remove these actions, we must find and remove all vertices (and corresponding actions) which have not any successors until there are no such vertices without successors. Other kind of unnecessary actions are such actions on that is a_{n+1} dependent, but does not exist any action which is straightly dependent on them by any predicate from the set of positive effects. The following algorithm detects and removes all of previously mentioned kinds of 'useless' actions and also modifies the corresponding Graph of Action Dependencies.

Algorithm 2:
`INPUT:` Plan $\pi = \langle a_1, \ldots, a_n \rangle$ which solves the planning problem $P = (\Sigma, s_0, g)$ and a Graph of Action Dependencies $G = (V, E)$ with respect to π and P.
`OUTPUT:` Modified plan π' where does not exist any action on which the goal is not dependent. Modified Graph of Action Dependencies with respect to π' and P

while exists $v_i \neq v_{n+1}$ such that no edge (v_i, v_j) for any v_j contains an assignment containing a predicate from $e^+(a_i)$ **do**
  remove a_i from π
  foreach $(v_j, v_i) \in E$ **do**
    foreach predicate p from the assignment of (v_j, v_i) **set** $d(p) = j$
  endforeach
  foreach $(v_i, v_k) \in E$ **do**
    foreach predicate p from the assignment of (v_i, v_k) **do** $E = E \cup (v_{d(p)}, v_k)$ if $(v_{d(p)}, v_k) \notin E$
  endforeach
  remove v_i and all its input and output edges from G
endwhile

Theorem 2. *[correctness of algorithm 2] Let* $\pi = \langle a_1, \ldots, a_n \rangle$ *is a plan which solves the planning problem* $P = (\Sigma, s_0, g)$ *and* $G = (V, E)$ *is the a Graph of Action Dependencies with respect to* π *and* P*. The algorithm 2 always computes modified plan* π' *which also solves the planning problem* P *and where* $\mid \pi' \mid \leq \mid \pi \mid$*. The algorithm 2 also computes the modified Graph of Action Dependencies with respect to* π' *and* P*.*

Proof. It is clear that the algorithm 2 always terminates, because there is a finite number of vertices in the Graph of Action Dependencies and in each step of while cycle is one of them deleted. It is also clear that $\mid \pi' \mid \leq \mid \pi \mid$ must be satisfied, because the algorithm 2 does not add actions to the plan. Let v_i is a vertex from V and a_i is a corresponding action from π that satisfy the condition in the while cycle. Let s_i is a state which is obtained by performing of the sequence of actions $\langle a_1, \ldots, a_i \rangle$ on s_0. It is clear that $e^+(a_i) \subseteq s_i$. If there exists any action a_j for $i + 1 \leq j \leq n + 1$ such that $e^+(a_i) \cap p(a_j) \neq \emptyset$ than the while condition must be broken (it can be seen from algorithm 1). If a_i is removed from π, the sequence of actions $\langle a_{i+1}, \ldots, a_{n+1} \rangle$ can be performed on the state $s_{i-1} = (s_i \setminus e^+(a_i)) \cup e^-(a_i)$, because $e^+(a_i) \cap p(a_j) = \emptyset$ for every $j \in \{i + 1, \ldots, n + 1\}$ (remember that a_{n+1} represents the goal state), so the plan $\langle a_1, \ldots, a_{i-1}, a_{i+1}, \ldots, a_n \rangle$ is valid and solves the planning problem P. Now it is needed to prove that the algorithm 2 also modifies the Graph of Action Dependencies. The first foreach cycle computes attributes of each predicate which is in $p(a_i)$ in such a way that the attributes have the same values as in algorithm 1 before i-th step of the for cycle. The second foreach cycle adds edges to the Graph of Action Dependencies in the same way as in the algorithm 1. □

The algorithm 2 removes all actions which is not needed to acquire the goal. The complexity of the algorithm 2 depends on a fact how many steps we need to find a vertex satisfying the condition of the while cycle and how many times the while cycle will run. The while cycle can run at most $O(n)$ times, because after n runs there will remain only the special actions a_0 and a_{n+1}. To find a vertex satisfying the condition of the while cycle is needed $O(n)$ steps in a worst case (when using a naive search algorithm). This estimation should be improved by creating a list of successors of each vertex such that each edge between the vertex and its successors will contain at least one predicate from a set of positive effects from the action which corresponds with the vertex. It is clear that every vertex whose list is empty satisfies the condition of the while cycle. These lists can be created during the construction of the Graph of Action Dependencies and modified during any modification of the Graph of Action Dependencies.

4.2 Removing of inverse actions

Inverse actions are a couple of actions such that their performance consequently on a state leads to reach the same state as before the performance. The inverse actions are usually pairs of actions a,b such that $e^+(a) = e^-(b)$ and $e^-(a) = e^+(b)$. To detect and remove pairs of inverse actions we can also use the Graph of

Action Dependencies, because the pairs of inverse actions which can be omitted do not appear usually in plans consecutive. The idea of following algorithm is quite similar to the idea of the algorithm 2 and is based on detecting and removing of pairs of inverse actions which can be omitted.

Algorithm 3:
INPUT: Plan $\pi = \langle a_1, \ldots, a_n \rangle$ which solves the planning problem $P = (\Sigma, s_0, g)$ and a Graph of Action Dependencies $G = (V, E)$ with respect to π and P.
OUTPUT: Modified plan π' where does not exist any unnecessary pair of inverse actions. Modified Graph of Action Dependencies with respect to π' and P

while exists inverse actions a_i and a_j, where $i, j \neq 0, n+1$ such that $(v_i, v_j) \in E$ and no edge (v_i, v_k) for any $v_k \neq v_j$ contains an assignment containing a predicate from $e^+(a_i)$ **do**
  remove a_i and a_j from π
  foreach $(v_k, v_i) \in E$ and $(v_l, v_j) \in E$, where $l \neq i$ **do**
    foreach predicate p from the assignment of (v_k, v_i) **set** $d(p) = k$
    foreach predicate p from the assignment of (v_l, v_j) **set** $d(p) = l$
  endforeach
  foreach $(v_i, v_k) \in E$, where $k \neq j$, and $(v_j, v_l) \in E$ **do**
    foreach predicate p from the assignment of (v_i, v_k) **do** $E = E \cup (v_{d(p)}, v_k)$ if $(v_{d(p)}, v_k) \notin E$
    foreach predicate p from the assignment of (v_j, v_l) **do** $E = E \cup (v_{d(p)}, v_l)$ if $(v_{d(p)}, v_l) \notin E$
  endforeach
  remove v_i,v_j and all their input and output edges from G
endwhile

Theorem 3 (correctness of algorithm 3). *Let $\pi = \langle a_1, \ldots, a_n \rangle$ is a plan which solves the planning problem $P = (\Sigma, s_0, g)$ and $G = (V, E)$ is a Graph of Action Dependencies with respect to π and P. The algorithm 3 always computes modified plan π' which also solves the planning problem P and where $\mid \pi' \mid \leq \mid \pi \mid$. The algorithm 3 also computes the modified Graph of Action Dependencies with respect to π' and P.*

Proof. The proof of termination of the algorithm 3 and the proof of $\mid \pi' \mid \leq \mid \pi \mid$ can be done in a same way as the proof of theorem 2. Let a_i and a_j from π are inverse actions that satisfy the condition of the while cycle. Let v_i and v_j are corresponding vertices to the actions a_i and a_j. Let s_{i-1} is a state which is obtained by performing of the sequence of actions $\langle a_1, \ldots, a_{i-1} \rangle$ on s_0. It is clear that performance of the actions a_i and a_j consecutively on the state s_{i-1} will result in the same state s_{i-1}. If there exists any action a_k for $i + 1 \leq k \leq j - 1$ such that $e^+(a_i) \cap p(a_k) \neq \emptyset$ than the while condition must be broken (it can be seen from algorithm 1). If a_i and a_j is removed from π, the plan $\langle a_1, \ldots, a_{i-1}, a_{i+1}, \ldots, a_{j-1}, a_{j+1}, \ldots, a_{n+1} \rangle$ is also valid, because all actions which are straightly dependent on a_j can be straightly dependent on actions performed before a_i (it is clear that all predicates from $e^+(a_j)$ must

exist before a_i is performed). If there exists some action which is straightly dependent on a_i, it is dependent only on predicates from $p(a_i) \setminus e^-(a_i)$, these predicates are also available before a_i is performed. Proof that the algorithm 3 modifies the Graph of Action Dependencies correctly can be done in the same way as in proof of theorem 2. □

The algorithm 3 removes all pairs of inverse actions which can be omitted. The complexity of the algorithm 3 (likewise as on the algorithm 2) depends on a fact how many steps we need to find a pair of inverse actions satisfying the condition of the while cycle and how many times the while cycle will run. The while cycle runs at most $\frac{n}{2}$ times, because in each run of the while cycle 2 actions are removed. To find a pair of inverse actions satisfying the condition of the while cycle is needed $O(m)$ steps in a worst case, where m is number of edges in Graph of Action Dependencies and $m \leq n^2$. This estimation should be improved by creation of a list of edges which were not tested.

4.3 Looking for plans optimization in general

Previous subsections showed us two main possibilities of making plans more optimal in polynomial time. However, the complexity results [7] showed that there are some planning domains where is much harder to find an optimal plan than a satisfying (not optimal) plan. Next lines will show possibilities of usage Graph of Action Dependencies for plans optimization in general.

Definition 7. *Let $G = (V, E)$ is a Graph of Action Dependencies. $G' = (V', E')$ is* Subgraph of Action Dependencies *of G if and only if the next conditions are satisfied. V' is a subset of V. There is at most one vertex $v \in V'$ such that there exists an edge $(v', v) \in E$, where $v' \notin V'$. There is at most one vertex $w \in V'$ such that there exists an edge $(w, w') \in E$, where $w' \notin V'$. $(v_k, v_l) \in E'$ if and only if $v_k, v_l \in V'$ and $(v_k, v_l) \in E$.*

Previous definition gives a description how can be a given plan split into subplans which are satisfying planning subproblems of a given planning problem.

Proposition 4. *Let $G = (V, E)$ is a Graph of Action Dependencies with respect to π and P, where π is a plan which solves the planning problem P. If every Subgraph of Action Dependencies of G cannot be replaced by another Graph of Action Dependencies representing the solution of the same planning subproblem and having less of vertices than π is an optimal plan for the planning problem P.*

Proof. It is clear that if some part of a plan can be replaced by a shorter one than the plan is not optimal. The meaning of definition 7 is in a fact that every Subgraph of Action Dependencies represents a part of a plan (or whole plan) and if the Subgraph of Action Dependencies can be replaced by anther Graph of Action Dependencies representing the solution of the same planning subproblem and which has less of vertices, the part of the plan (or the whole plan) represented by the Subgraph of Action Dependencies is not optimal. □

Proposition 4 gives an idea how to make an algorithm based on searching for the Subgraphs of Action Dependencies and their corresponding subplans which always computes from a plan an optimal plan. However, in many planning problems the algorithm would have to replan the whole plan which means that in these case the usage of an optimal planned will be better. Luckily, the algorithm can be used for searching for subplans of length 2 and replacing them by plans contains one single action. In this case, the algorithm can run in polynomial time, the idea of searching for subplans of length 2 is very similar to searching for pairs of inverse actions (described previous subsection). When the algorithm is used for searching for subplans of length bigger than 2, the time needed to run of the algorithm can rise exponentially which means that the algorithm can be used for subplans of small length (not much greater than 2).

5 Future research

My plans of future research in this area are quite large, because the Graph of Action Dependencies seems to be a useful feature in many areas of planning. In the following lines, I will briefly introduce main possibilities of the other usage of Graph of Action Dependencies.

The Graph of Action Dependencies seems to be a very useful tool which can be good for a plan analysis. The plan analysis can be very useful in looking for heuristics and for gathering knowledge of planning domains which can help with computation of solutions of more larger planning problems.

At last, it seems to be useful to extend the Graph of Action Dependencies into a temporal planning. This extension can be very helpful in the connection between planning and scheduling and also can be helpful in (temporal) plan optimization and in achieving better plans quality by matching more constraint preferences.

6 Conclusion

This paper presented the Graph of Action Dependencies which showed us that it is an useful tool for plans optimization. There were presented algorithms for removing unnecessary actions like actions which are not needed to acquire a goal or pairs of inverse actions in a polynomial time which means that these actions can be removed from a plan quickly and make this plan more optimal. However, due to previous complexity results is not possible to get an optimal plan in a polynomial time, but this paper showed a possibility of making more optimal plans through optimizing their subplans which have a small length (if the length is 2 the optimization can be computed in a polynomial time). Despite the fact that plans cannot be fully optimized in polynomial time, presented algorithms seems to be very useful in combination with existing planners, because it should provide an improvement in plans quality in a short time.

7 Acknowledgements

I thank the reviewer for the comments. The research is supported by the Czech Science Foundation under the contract no. 201/07/0205 and by the Grant Agency of Charles University (GAUK) under the contract no. 326/2006/A-INF/MFF.

References

1. Blum A., Furst M. Fast planning through planning graph analysis. *Artificial inteligence*. 90:281–300. 1997.
2. Fikes R., Nilsson L. STRIPS: A new approach to the application of theorem proving to problem solving. *Artificial Inteligence* 2(3-4): 189-208. 1971.
3. Geffner H. Causal theories of nonmonotonic reasoning. *In proceedings of AAAI*. 524-530. 1990.
4. Gelfond M., Lifschitz V. Action Languages. *Electronic Transactions on Artificial Intelligence* 2:193-210. 1998.
5. Gerevini A., Serina I. LPG: a planner based on local search for planning graphs. *In proceedings of IPC*. 2002.
6. Ghallab M., Nau D., Traverso P. *Automated planning, theory and practice*. Morgan Kaufmann Publishers. 2004.
7. Helmert M. New Complexity Results for Classical Planning Benchmarks. *In proceedings of ICAPS*. 52-61. 2006.
8. Hoffmann J., Nebel B. The FF planning system: Fast plan generation through heuristics search. *Journal of Artifical Inteligence Research* 14:253-302. 2001.
9. Hoffmann J., Porteous J., Sebastia L. Ordered Landmarks in Planning. *Journal of Artifical Inteligence Research* 22:215-278. 2004.
10. `http://ipc.icaps-conference.org`
11. Lin F. Embracing causality in specifiing the indirect effects of actions. *In proceedings of IJCAI*. 1985-1991 1995.
12. McCain N., Turner H. Causal Theories of Action and Change. *In proceedings of AAAI*. 460-465. 1997.
13. Mehlhorn K. *Data Structures and Algorithms 2: Graph Algorithms and NP-Completeness*. Springer-Verlag. 1984.
14. Surynek P.; Chrpa L.; and Vyskocil J. Solving difficult problems by viewing them as structured dense graphs. *To appear at IICAI*. 2007.
15. Sussman G. *A Computational Model of Skill Acquision* New York: Elsevier. 1975.
16. Vidal V., Geffner H. Branching and pruning: An optimal temporal POCL planner based on constraint programming. *Artificial Inteligence* 170(3): 298-335. 2006.
17. Westerberg C. H., Levine J. Optimising plans using genetic programming *In proceedings of ECP* 2001.

Proceedings of the International Multiconference on Computer Science and Information Technology pp. 225–236
ISSN 1896-7094
© 2007 PIPS

Manifestation of selective attention in Sigma-if Neural Network

Maciej Huk[1]

Wroclaw University of Technology
Institute of Applied Informatics
Wroclaw, Poland
Maciej.Huk@pwr.wroc.pl, http://www.iis.pwr.wroc.pl/~huk

Abstract. Artificial neural networks are a very well known biologically-inspired machine learning technique. This technique has been widely applied in many domains, such as real-time signal filtering, modelling and synthesis, process control and classification (e.g. of images, diseases or star spectra). Other examples of use of neural networks include generation of rules for expert systems and knowledge discovery. However, effective analysis of multidimensional data sets still causes particular difficulties. It has been shown that processing too many data features is costly and has adverse effects on the classification properties of resulting models. Thus, while development of techniques for selecting features from very large data sets is not trivial, it is nonetheless very important. It can be observed that analogous problems are effectively solved by human low-level distributed selective attention mechanisms. For this reason, building a general (e.g. neural) model of such functionality would provide great benefits for the machine learning domain. This work addresses selective attention functionality found in the recently developed Sigma-if neural network. Experiments show how this selective attention model can reduce data acquisition and processing costs as well as the probability of classification errors.

1 Introduction

Artificial neural networks are one of the most valuable biologically-inspired machine learning techniques. They play a significant role in many data processing applications - from real-time signal filtering, modelling and synthesis, through process control to classification (e.g. of images, diseases or star spectra). Other important examples of neural network use include generation of rules for expert systems and knowledge discovery in data [1–4]. Typical examples of artificial neural networks are multilayer perceptrons (MLP), radial basis function networks (RBF) and asynchronous spiking neural networks. A wider description of artificial neural network types can be found in many recent surveys [5–8].

In turn, biological inspirations of artificial neural networks suggest that such systems can effectively realise feature selection tasks easily observed in humans and other primates [9, 10]. Artificial models of this phenomenon could reduce

problems with proper data selection and integration still observed in most machine learning systems. This could be of great benefit to all domains in which effective analysis of multidimensional data sets is crucial [11–13]. The most important question in this field is what basic neural mechanisms implement the functionality called selective attention [14–16].

Despite the great variety of known neural network models, very few realize any aspect of selective attention functionality [15, 17, 18]. This is probably due to the fact that such models do not copy some important processes which evidently take place during input signal aggregation in real neurons [17, 19–25]. In this context it is very interesting that selective attention functionality, which seems to effectively mimic low-level attentional processes observed in humans, was found in the recently developed simple generalization of the well-known MLP network called Sigma-if [4, 26].

The remainder of this paper is organized as follows. Section 2 presents the concept and basic properties of selective attention systems as well as main results of to-date research in the field of selective attention modelling. Section 3 describes the Sigma-if neural network model, which realizes selective attention functionality. Section 4 illustrates how selective attention manifests itself in properties of the Sigma-if neural network on the example of selected UCI Machine Learning benchmark data sets. Finally, conclusions are presented in Section 5.

2 Previous work

In nature, selective attention is a mechanism which provides living organisms with the possibility to sift incoming data, to extract information which is most important at a given moment and which should be processed in detail [14, 27]. This mechanism is necessary due to limited processing capabilities of the nervous system which does not allow rapid analysis of the whole scene of visual and other senses [17, 20]. Selective attention can be viewed as a strategy of dynamical input space selection for gaining predefined goals by an organism interacting with a very complicated environment.

Potential benefits from adopting the selective attention mechanisms in domains such as data analysis and feature selection have led many researchers to search for appropriate models of this feature of the human brain [9, 10, 15, 16, 20, 28, 29]. The first widely known result was Broadbent's bottleneck model [30], but even after improvements by Treisman [31] it was found to be too inaccurate. Another important development was the late selection model by Deutsch and Norman [32] and its two-step version by Snyder and Posner [33]. These models were founded on strong evidence, through research of Lewis [34], Corteen [35], Neely [36] and Pynte [37]. Although late selection models were computationally very costly, they showed that fast, low-level automatic data selection is crucial for achieving effectiveness of slow contextual data analysis realized by higher organizational levels of neural structures [38–41]. On this basis, the most important achievement of neuropsychological studies arose - namely, the development of earlier works by Noton and Stark on saccadic eye movements [42–44]. Element

of automatic, cyclic attentional field selection, according to learned scan paths, turned out to be common in almost all input channels of a human brain [45, 46].

Many models of scan path selection strategies were subsequently proposed - this included hierarchical neural routing and shifting circuits [10, 16, 47] as well as oscillating and adaptive neural networks [29, 48, 49]. They made it possible to build specialised tools, e.g. for face recognition, object tracking or indicating interesting objects for observation by space probes [50–53]. However, none of these solutions yielded a general model of low-level selective attention. In fact, most of them were very complicated and realised only centralised or predefined (i.e. not learned or evolved) feature selection strategies [15, 17, 18]. The degree of their specialisation and neurobiologically unfounded solutions induced researchers to look for basic selective attention mechanisms on the level of single synapses and neurons [54–57].

Neuropsychological and neurobiological studies of selective attention phenomena have led researchers to the conclusion that low-level selective attention functionality is realized in a distributed manner by serial processes that are carried out by one to three layers of neurons [19, 17, 20]. Simultaneously, selective attention at higher levels of brain structure organization seems to emerge as an effect of synergy between elementary structures at lower levels (e.g. neurons or groups of neurons). Unfortunately, the exact mechanisms that lie at the base of selective attention are still unknown.

This is why networks that use higher-order neuron models, such as Sigma-Pi [58–60], Power Unit [61] or Clusteron [62], realize only a very limited set of attentional mechanisms. Thus it can be very interesting that selective attention functionality which seems to effectively mimic low-level attentional processes observed in humans, was found in a recently-developed simple generalization of the well-known MLP network called Sigma-if [4, 26, 63].

3 The Sigma-if neural network

The proposed Sigma-if neural network is a type of fully connected, synchronous multilayer perceptron neural network (MLP) which possess selective attention abilities [26, 63]. Such a neural network does not use separate centralized attention guidance modules. Its ability to realize selective attention functionality emerges as an effect of synergy between its hidden, Sigma-if neurons. Each Sigma-if neuron is a special direct generalization of a sigmoidal perceptron which implements basic selective attention functionality via input connections grouping and stepwise conditional input signal accumulation. This is due to the new neuron's aggregation function [5, 26].

3.1 The Sigma-if neuron

The Sigma-if neuron, in contrast to a typical perceptron, aggregates input signals for the given data vector not in one but in a series of given K steps according to the corresponding state graph. Its input connections are divided into K discrete

subsets during the training process. Subsequently, when the neuron's aggregation function value is computed, in every k-th step of this process, the subset of input signals X_k is taken from the environment and processed to determine the current value of the partial activation level φ_k of the neuron. The process continues until the value of φ_k exceeds a given aggregation threshold φ^*. When that condition is met, signals which were not analyzed are ignored, and φ_k is considered the input value for the neuron's activation function F. As a result, signal level information coding and even the use of a non-local activation function (e.g. sigmoid) do not degrade the neuron's selective attention abilities. A sample scheme of a state graph for a Sigma-if neuron is presented in Fig. 1.

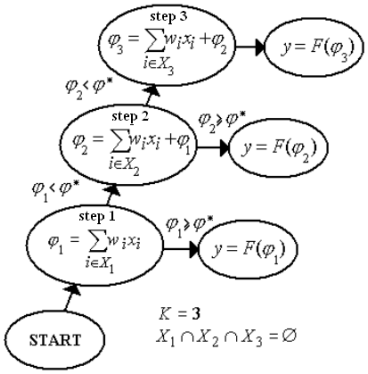


Fig. 1. Sample process of a three-step input signal aggregation in the Sigma-if neuron

Formally speaking, M dendrites of the Sigma-if neuron are divided into K distinct groups, by complementing each i-th input connection with an additional integer parameter $\theta_i \in \{0,1,..,K-1\}$, determining membership in one of the groups. This allows us to divide the process of signal accumulation into K steps, where K is a function of the neuron's grouping vector $\theta^T = [\theta_1, \theta_2, ..., \theta_M]$:

$$K(\theta) = \max_{i=1}^{M}(\theta_i). \tag{1}$$

During each step k (from 0 to K-1), the neuron accumulates data belonging to one selected group, such that

$$\theta_i = k. \tag{2}$$

Within each k-th group, partial activation $\Delta\varphi(k)$ is determined as a weighted sum of input signals and the appropriate Kronecker's delta:

$$\Delta\varphi_k(w, x, \theta) = \sum_{i=1}^{M} w_i x_i \delta(k, \theta_i), \tag{3}$$

where w_i and x_i are coefficients of the neuron's weight vector w and an input vector x. This process is repeated until the activation derived from respective groups exceeds a preselected aggregation threshold φ^*. It can be described by the following recursive formula (vectors w, x and θ are omitted for clarity):

$$\varphi_k = \begin{cases} \Delta\varphi_k H(\varphi^* - \varphi_{k-1}) + \varphi_{k-1} : k \geq 0 \\ 0 : k < 0 \end{cases} \tag{4}$$

where H is Heaviside's function. This sum is then treated as the neuronal activation value. The input from remaining (heretofore unconsidered) groups is neglected. Thus, the proposed form of the aggregation function A is:

$$A(w, x, \theta) = \varphi_K(w, x, \theta). \tag{5}$$

In the final stages of determining the output value Y of the neuron, function (5) serves as a parameter of the nonlinear threshold (e.g. sigmoidal) function F:

$$Y(w, x, \theta) = F(A(w, x, \theta)). \tag{6}$$

It is worth noting that the described model assumes that the state graph used during signal aggregation is always a simple directed path of nonterminal nodes corresponding with the neural activation accumulation procedure. In a general case, the Sigma-if neuron, besides the weights vector w, includes one continuous valued parameter for aggregation threshold φ^*, and an additional connections grouping vector θ with only one nominal valued coefficient for each neuronal input connection.

3.2 Sigma-if network training

In comparison to MLP neural network training, searching for a globally optimal set of Sigma-if network parameters would be very computationally challenging. This is due to the noncontinuous character of Sigma-if neuron grouping vectors.

While there is no quick and effective method for global searching of network weights and grouping vectors, the proposed solution assumes that at each Sigma-if neuron, coefficients of the grouping vector θ are in fact direct functions of that weight vector. As a result, network connection weights are established by the well known error backpropagation algorithm, but for every ω training epoch, actual grouping vectors are computed. This reflects the application of the self-consistency idea widely used in physics.

In this work, the grouping vector computation procedure (as well as the predefined value of the aggregation threshold φ^*) is common for all Sigma-if neurons. It simply divides input connections into a given number of groups, according to "the greater the connection weight, the smaller the connection group number" principle. Such a search problem reduction leads us to very interesting results, and to practical elimination of the additional θ parameters vector. However, during further analysis of the general Sigma-if model, it is still very helpful to use the grouping vector concept.

4 Properties of the Sigma-if neural network

During examination of the proposed model properties, Sigma-if neural networks were compared to MLP networks with the same architectures. All networks were fully connected and had one hidden layer with the number of neurons for which the MLP network gained highest test data classification accuracy. As the sigmoidal perceptron is a special case of a Sigma-if neuron, MLP networks were in fact simulated by Sigma-if networks with the number of inputs groups K of all Sigma-if neurons set to one. This was consistent with the additional assumption that K and aggregation threshold φ^* values of all hidden Sigma-if neurons in the given network were equal. In all cases, standard input signal coding was used, and answers of the network were computed in the winner-takes-all manner.

Beside classification accuracies for training (u) and test (γ) data, properties such as network data processing time (τ), as well as hidden connections and network input activity (*hca* and *nia* respectively) were considered. The data processing time τ for all trained networks was measured on the same computer (P4 2.6GHz) to check relative data processing costs for MLP and Sigma-if networks. Regardless of the very precise time measurement procedure used (following measurement of real processor core frequency, the number of processor clock cycles was counted, starting from the end of loading data into network inputs and finishing with the end of the network output value generation), actual timings on other hardware setups may vary considerably.

Hidden connection activity (*hca*) and network input activity (*nia*) were used to representing the percentage ratio of the number of hidden and input connections used during data processing, compared to all of the network's hidden and input connections respectively. These parameters allowed us to check if hidden Sigma-if neurons used their selective attention ability in practice. For the completeness of analysis, for each given problem and trained network, the percent of all inputs used to classify all test vectors (*niu*) was calculated. This procedure was important in order to determine if selective attention functionality is also realized at the level of the whole Sigma-if network. All presented values for all considered UCI Machine Learning Repository benchmark problems were calculated as averaged outcomes of ten independent 10-fold cross validations.

The conducted experiments can be divided into two groups: examination of Sigma-if network classification properties and verification of the hypotheses that Sigma-if networks realize selective attention at the level of single neurons and of the whole network. In both cases all of the considered properties were analyzed in relation to the number of Sigma-if neuron inputs groups K. This was because the K value has the highest influence on the properties of the proposed network. Other parameters, such as the aggregation threshold φ^* and the grouping vector actualization interval ω, were set to 0.6 and 25 respectively, following preliminary tests. It is also important to note that parameters such as *hca*, *nia* and *niu* are informative only for Sigma-if networks; for MLP networks ($K = 1$) they are, by default, equal to 100%.

The obtained results indicate that increasing the number of Sigma-if neuron input groups K to more than one results in an increase of test data classification

accuracy γ as well as in simultaneous decrease of data processing time τ. The drawback here is a decrease of training data classification accuracy u. A typical example of such a dependency can be observed for the HeartC problem, which is presented in Fig. 2. The observed decrease in training data classification ac-

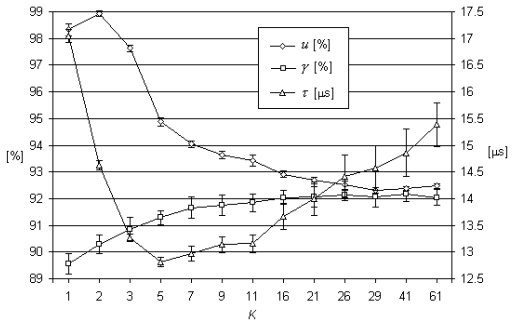


Fig. 2. The time of Sigma-if network output signal generation (τ), the classification accuracy of training (u) and test (γ) data for the HeartC problem versus the number of hidden neuron input connections groups K (networks architecture: 28 inputs, 10 hidden neurons, 5 outputs)

curacy u was most probably due to the fact that Sigma-if neurons process data in the same way after and during training. It is harder to learn when the neuron's input space is changed every ω epoch. However, and more importantly, the obtained increase in test data classification accuracy γ is a result of rejecting redundant or noisy signals from processed data and the consequence of effective reduction of problem dimensionality. This thesis is confirmed by the observed simultaneous decrease in data processing time τ. Reduction of τ can only be caused by reduction of the network's hidden and input connections activities. However, regardless of the reasons, these results shows that the Sigma-if neural network has better classification properties than MLP.

The visible increase of HeartC data processing time τ for K above 7 inputs groups is the effect of a linear increase of time cost, connected with the existence of additional instructions for grouping vector θ information processing. This factor can be easily seen for the number of groups K greater than the given number of network inputs. Without it, data processing time would semi-logarithmically decrease with rising K. This indicates the character of the changes of Sigma-if network hidden (hca) and input connection activities (nia) as a function of K, which can be observed in Fig. 3 (for the Sonar problem) and in Fig. 4 (for the Votes problem).

It can be easily seen that in the case of the Sonar problem the increase in K causes an increase of the test data classification accuracy, with a corresponding decrease of the training data classification accuracy. These changes are accompa-

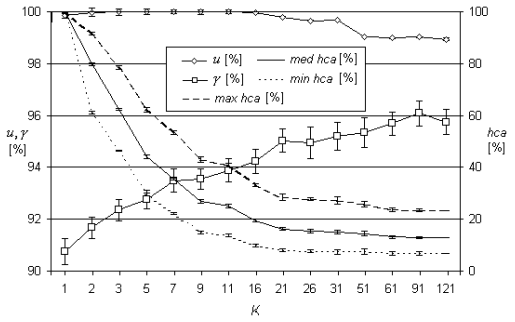


Fig. 3. The Sigma-if network hidden connection activity (*hca*), the classification accuracy of training (u) and test (γ) data for the Sonar problem versus the number of hidden neuron input connections groups K (network architecture: 60 inputs, 30 hidden neurons, 2 outputs)

nied with a much stronger reduction of hidden connection activities. The shape of the *hca(K)* function confirms earlier conclusions that the data processing time reduction is connected with Sigma-if neurons' selective attention abilities. All this is clear evidence that Sigma-if neurons use selective attention, and that this can reduce the generalization error level as well as data processing costs.

The last example concerns how Sigma-if selective attention abilities manifest themselves on the level of the whole network. The analysis of results for the Votes problem (Fig. 4) shows that when a significant decrease of network input activity (*nia*) occurs, one can expect a simultaneous reduction in the number of Sigma-if network inputs used to classify data (*niu*) without a notable decrease of classification accuracy in comparison to the MLP network. While the Sigma-if

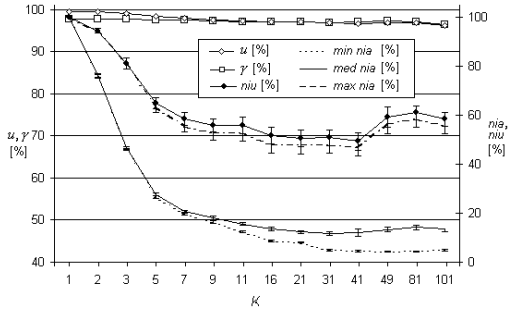


Fig. 4. The Sigma-if network input activity (*nia*), the number of network input used (*niu*), and the classification accuracy of training (u) and test (γ) data for the Votes problem versus the number of hidden neuron input connections groups K (network architecture: 48 inputs, 2 hidden neurons, 2 outputs)

network has no specialized or separate attention guiding unit, all such activities can emerge only as an effect of synergy between individual neurons. Thus, the observed selective attention behaviour of the proposed network, treated as a black box, is a significant indication that the Sigma-if model effectively mimics low level attentional processes observed in nature. This can, in turn, make the model an interesting tool for feature selection and other data processing purposes.

5 Conclusion

In this paper the Sigma-if neuron model and benefits from using the Sigma-if neural network instead of the MLP network were considered. The model's selective attention ability for medium-size test problems manifests itself in an increase of classification accuracy and in a simultaneous decrease of data processing costs. Additional reduction of the number of network inputs used to classify data hints at the possibility of further reduction of data collection costs, during as well as after the training process.

The Sigma-if selective attention feature also introduces new possibilities in the area of analyzing the network decision process via its input activity interpretation. This can point at features of given data sets that are most important for classification, and help identify features that are irrelevant, redundant or contaminated by noise.

What is more important, the Sigma-if neural network seems to be a promising model of low-level human selective attention. This functionality is realized in a distributed manner by sequential input signal accumulation, carried out by Sigma-if neurons, and emerges as an effect of synergy among the network's hidden neurons.

All this makes the Sigma-if neural network a very useful tool for the data analysis domain, despite the fact that the proposed method requires further tests on benchmark and real-life data. New Sigma-if network training methods as well as new selective attention neurons with aggregation functions other than those presented in this paper will be the subject of further research.

References

1. Anthony, M., Bartlett, P.: Neural networks learning: Theoretical foundations. 1st ed. edn. Cambridge University PressCambridge University Press, Cambridge (1999)
2. Srinivasan, T., Chandrasekhar, A., Seshadri, J., Siddharth, J.: Knowledge discovery in clinical databases with neural network evidence combination. In: Second IEEE International Conference on Intelligent Sensing and Information Processing - ICISIP 2005, Chennai, India, IEEE Press, (2005) 512–518
3. LiMin, F.: Knowledge discovery based on neural networks. Commun. ACM **42** (1999) 47–50

4. Huk, M.: The sigma-if neural network as a method of dynamic selection of decision subspaces for medical reasoning systems. Journal of Medical Informatics & Technologies **7** (2004) 65–73
5. Duch, W., Jankowski, N.: Survey of neural transfer functions. Neural Computing Surveys **2** (1999) 163–212
6. Duch, W., Adamczak, R., Diercksen, G.: Neural networks in non-euclidean metric spaces. Neural Processing Letters **10** (1999) 201–210
7. Chowdhury, F., Wahi, P., Raina, R., Kaminedi, S.: A survey of neural networks applications in automatic control. In: Southeastern Symposium on System Theory. (2001) 349–353
8. Zhao, Y., Gao, J., Yang, X.: A survey of neural network ensembles. In: ICNN&B'05. International Conference on Neural Networks and Brain. Volume 1. (2005) 438–442
9. Desimone, R., Duncan, J.: Neural mechanisms of selective visual-attention. Annual Review of Neuroscience **18** (1995) 193–222
10. Olshausen, B., Anderson, C., Van Essen, D.: A neurobiological model of visual attention and invariant pattern recognition based on dynamic routing of information. J. Neurosc. **13** (1993) 4700–4719
11. Fayyad, U., Piatetsky-Shapiro, G., Smyth, P.: Knowledge discovery and data mining: Towards a unifying framework. In: 2nd Intl. Conf. on Knowledge Discovery and Data Mining (KDD96), Portland, Oregon (1996) 82–88
12. Yu, H., Yang, J., Han, J.: Classifying large data sets using svm with hierarchical clusters. In: 2003 ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining (KDD'03), Wash., D.C. (2003)
13. Fayyad, U., Djorgovski, S., Weir, N.: Automating the analysis and cataloging of sky surveys. In: Advances in Knowledge Discovery and Data Mining. AAAI/MIT Press (1996) 471–493
14. Broadbent, D.: Task combination and selective intake of information. Acta Psychologica **50** (1982) 253–290
15. Heinke, D., Humphreys, G.: Computational Models of Visual Selective Attention: A Review. In: Connectionist Models in Psychology. Psychology Press (2002)
16. Koch, C., Ullman, S.: Shifts in selective visual attention: towards the underlying neural circuitry. Hum. Neurobiology **4** (1985) 219–227
17. Tsotsos, J., Culhane, S., Cutzu, S.: From foundational principles to a hierarchical selection circuit for attention. In: Visual Attention and Cortical Circuits. MIT Press, Cambridge, MA (2001) 285–306
18. Privitera, C., Stark, L.: Algorithms for defining visual region-of-interest: Comparison with eye fixations. IEEE Transactions on Pattern Analysis and Machine Intelligence **22** (2000) 970–982
19. Spratling, M., Hayes, G.: Learning synaptic clusters for nonlinear dendritic processing. Neural Processing Letters **11** (2000) 17–27
20. VanRullen, R., Koch, C.: Visual selective behavior can be triggered by a feedforward process. Journal of Cognitive Neuroscience **15** (2003) 209–217
21. Anderson, R., Essich, G., Siegal, R.: Encoding of spatial location by posterior parietal neurons. Science **230** (1985) 456–458
22. Karlholm, J.: Associative memories with short-range, higher order couplings. Neural Networks **6** (1993) 409–421
23. Stuart, G., Spruston, N.: Determinants of voltage attenuation in neocortical pyramidal neuron dendrites. Journal of Neuroscience **18** (1980) 3501–3510
24. Larkum, M., Zhu, J., Sakmann, B.: A new cellular mechanism for coupling inputs arriving at different cortical layers. Nature **398** (1999) 338–341

25. Körding, K., König, P.: Neurons with two sites of synaptic integration learn invariant representations. Neural Computation **13** (2001)) 2823–2849
26. Huk, M., Kwasnicka, H.: The concept and properties of sigma-if neural network. In Ribeiro, B., Albrecht, R., Dobnikar, A., Pearson, D., Steele, N., eds.: Adaptive and Natural Computing Algorithms, Springer Computer Science (2005) 13–17
27. Treisman, A.: Contextual cues in selective listening. Quarterly Journal of Experimental Psychology **12** (1960) 242–248
28. Xiao-Ping, Z.: Thresholding neural network for adaptive noise reduction. IEEE Transactions on Neural Networks **12** (2001) 567–584
29. Houghton, G., Tipper, S.: Inhibitory mechanisms of neural and cognitive control: Applications to selective attention and sequential action. Brain and Cognition **30** (1996) 20–43
30. Broadbent, D.: Perception and Communication. Pergamon Press, Oxford (1958)
31. Treisman, A.: Selective attention in man. British Medical Bulletin **20** (1964) 12–16
32. Norman, D.: Toward a theory of memory and attention. Psychological Review **75** (1968) 522–536
33. Posner, M., Snyder, C.: Facilitation and inhibition in the processing of signals. In: Attention and Performance. Academic Press, London (1975)
34. Lewis, J.: Semantic processing of unattended messages using dichotic listening. Journal of Experimental Psychology **85** (1970) 220–227
35. Corteen, R., Wood, B.: Autonomic responses to shock-associated words in an unattended channel. Journal of Experimental Psychology **94** (1972) 308–313
36. Neely, J.: Semantic priming and retrieval from lexical memory: Roles of inhibitionless spreading activation and limited capacity attention. Journal of Experimental Psychology: General **106** (1977) 226–254
37. Pynte, J., Do, P., Scampa, P.: Lexical decisions during the reading of sentences containing polysemous words. In: Preparatory States and Processes. Erlbaum, Hillsdale NJ (1984)
38. Sperling, G.: A unified theory of attention and signal detection. In: Varieties of attention. Academic Press, New York (1984) 103–181
39. Dosher, B., Sperling, G.: Strategy optimization in human information processing. In: Handbook of perception and human performance: Sensory processes and perception. Volume 1. John Wiley and Sons, New York (1986) 1–65
40. Neumann, O.: Beyond capacity: A functional view of attention. In: Perspectives on perception and action. Lawrence Erlbaum Associates, Hillsdale, NJ (1987)
41. Posner, M., S.E., P.: The attention system of the human brain. Ann. Rev. Neuro. **13** (1990) 25–42
42. Noton, D., Stark, L.: Scanpaths in saccadic eye movements while viewing and recognizing patterns. Vision Research **11** (1971) 929–942
43. Findlay, J.: Global visual processing for saccadic eye movements. Vision Research **22** (1982) 1033–1046
44. Chelazzi, L., Miller, E., Duncan, J., Desimone, R.: A neural basis for visual search in inferior temporal cortex. Nature **363** (1993) 345–347
45. Tsal, Y.: Movements of attention across the visual field. Journal of Experimental Psychology: Human Perception and Performance **9** (1983) 523–530
46. Kastner, S., De Weerd, P., Desimone, R., Ungerleider, L.: Mechanisms of directed attention in the human extrastriate cortex as revealed by functional mri. Science **282** (1998) 108–111
47. Anderson, C., Van Essen, D.: Shifter circuits: a computational strategy for dynamic aspects of visual processing. Proc. Natl. Academy Sci. USA **84** (1987) 6297–6301

48. Niebur, E., Koch, C., Rosin, C.: An oscillation-based model for the neural basis of attention. Vision Research **33** (1993) 2789–2802
49. Tsotsos, J., Culhane, S., Wai, W., Lai, Y., Davis, N., Nuflo, F.: Modeling visual attention via selective tuning. Artificial Intelligence **78** (1995) 507–547
50. Yamada, K., Cottrell, G.: A model of scan paths applied to face recognition. In: 17th Ann. Cog. Sci. Conf., Pittsburgh (1995)
51. Rybak, I., Gusakova, V., Golovan, A., Podladchikova, L., Shevtsova, N.: A model of attention-guided visual perception and recognition. Vision Research **38** (1998) 2387–2400
52. Hager, G., Toyama, K.: Incremental focus of attention for robust visual tracking. International Journal of Computer Vision **35** (1999) 45–63
53. Privitera, C., Azzariti, M., Stark, L.: Locating regions-of-interest for the mars rover expedition. JRS **21** (2000) 3327–3347
54. Mel, B.: Why have dendrites? A computational perspective. In: Dendrities. Oxford University Press (1999) 271–289
55. Baldassi, S., Verghese, P.: Comparing integration rules in visual search. Journal of Vision **2** (2002) 559–570
56. Shimozaki, S., Eckstein, M., Abbey, C.: Comparison of two weighted integration models for the cueing task: linear and likelihood. Journal of Vision **3** (2003) 209–229
57. Hamker, F.: Predictions of a model of spatial attention using sum- and max-pooling functions. Neurocomputing **56** (2004) 329–343
58. Feldman, J., Ballard, D.: Connectionist models and their properties. Cognitive Science **6** (1982) 205–254
59. Rumelhart, D., Hinton, G., McClelland, J.: A general framework for parallel distributed processing. In: Parallel Distributed Processing: Explorations in the Microstructure of Cognition: Foundations. Volume 1. The MIT Press (1986) 45–76
60. Mel, B.: The sigma-pi column: a model of associative learning in cerebral cortex. Technical report, CNS Memo 6, Computation and Neural Systems Program, California Institute of Technology (1990)
61. Durbin, R., Rumelhart, D.: Product units: A computationally powerful and biologically plausible extension to backpropagation networks. Neural Computation **1** (1990) 133–142
62. Mel, B.: The clusteron: toward a simple abstraction for a complex neuron. In: Advances in Neural Information Processing Systems. Volume 4. Morgan Kaufmann (1992) 35–42
63. Huk, M.: Sigma-if neural network as a use of selective attention technique in classification and knowledge discovery problems solving. Annales UMCS Informatica AI **5** (2006) 121–131

Proceedings of the International Multiconference on
Computer Science and Information Technology pp. 237–246
ISSN 1896-7094
© 2007 PIPS

Recognition of Structured Collocations in An Inflective Language

Bartosz Broda, Magdalena Derwojedowa, and Maciej Piasecki

Institute of Applied Informatics, Wrocław University of Technology, Poland
Wybrzeże Wyspiańskiego 27, Wrocław, Poland,
bartosz.broda,maciej.piasecki@pwr.wroc.pl
Institute of Polish, University of Warsaw
derwojed@uw.edu.pl

Abstract. We present a method of the structural collocations extraction for an inflective language (Polish) based on the process divided into two phases: extraction and filtering of the pairs of wordforms reduced to baseforms and structural annotation of the extracted collocations with lexico-syntactic patterns. The parameters of the patterns are specified manually but their instances are generated and tested on the corpus automatically. The extracted collocations were evaluated by applying them as rules in morpho-syntactic disambiguation of Polish and by comparing them with a lists of two-word expressions extracted from two Polish dictionaries.

1 Introduction

According to the generative power of the natural language, humans are able to produce the infinite number of sentences as well as they can flexibly combine words in compliance with syntactic and semantic rules. However, some sequences of words express more fixed structure than others: their constituents co-occur more often and changes in their structure are very restricted (sometimes even impossible). There is no general name for this broad class of 'non-atomic language units' as subsets of the class (varying in the scope of their semantic properties) are called: *collocations*, *fixed expressions*, *terms* or *proper names*. Further on we will call them simply multiword expressions (MEs) or collocations (*sensu largo*).

As collocations introduce a kind of fixed points into the space of possible language expressions, they are very important for the *Natural Language Engineering* (NLE), e.g. identification of collocations can enrich and reduce the description of a document in *Information Retrieval* (IR), improve the accuracy of OCR, or increase the quality of *Machine Translation* (MT). Unfortunately, only a small number of collocations (mostly idioms) is listed in dictionaries, partly because collocation lists are very large and many collocations are domain dependent. That is why the *automatic recognition of collocations* on the basis of large set of text documents—*a corpus*—is very important to the applications in NLE, IR, MT and the similar areas.

There are plenty of methods for recognition of collocations starting with the seminal paper [17]. Most of them are based on the statistical measures of likelihood of the co-occurrence of two *word forms* (WFs) in texts. This general scheme works fine for English but it expresses two significant drawbacks in the case of inflective languages like Polish. Firstly, the fixed order of constituents implicitly assumed in many methods doesn't work for (almost) free word order in Polish; secondly (this is even more important), Polish lexemes are expressed by many WFs. All well known methods treat each WF separately, e.g. two sequences *czerwoną kartkę* (*red card*$_{\text{case=acc, number=sg}}$) and *czerwonych kartek* (*red card*$_{\text{case=gen, number=pl}}$) are analysed as two different collocations regardless of the fact that they are both derived from the expression *czerwona kartka* (*red card*$_{\text{case=nom, number=sg}}$) and differ only in the values of the *case* and the *number*. The syntactic structure and meaning of these expressions are the same ('penalty card', as in football).

The aim of this work is to construct a method of collocations recognition that copes with the large number of WFs for one lexeme and identifies the basic syntactic structure of a collocation i.e. the morpho-syntactic dependencies between words in it.

There is no common definition of a collocation. In this paper we adopt the one by Manning & Schütze (it can be regarded as a mainstream definition; [7, pp. 151]):

> "*A collocation* is an expression consisting of two or more words that correspond to some conventional way of saying things." "[...] are characterized by limited *compositionality*".

Collocations are not compositional in their meaning, i.e. the meaning of a collocation cannot be fully predicted from the meanings of the constituents. It is impossible to exchange one of the collocation constituents to its synonym, e.g. *czerwony arkusz* (*a red sheet*) means something different than *czerwona kartka* (*a red card*), while in many contexts *arkusz* (*a sheet*) is a synonym of *kartka* (*a card*, cf. [3]). Moreover, some types of collocation like *fixed expressions* (cf. [8]) have irregular syntactic structure. Collocations include or at least overlap in large extent with *terminology*, i.e. technical terms and proper names ([7, 5, 10]).

Most methods of collocation recognition are based on the identification of such sequences of WFs that are more frequent than it would be expected from the probabilistic distributions of their constituents. Several statistical and heuristic measures based on statistics have been proposed. An extensive list of 84 measures is surveyed in [10]. In [1], the work on the recognition of collocations in Polish corpus, 16 different measures are tested. Statistical identification of significantly frequent sequences is often accompanied by additional pre- and post-processing, especially in the case of languages of rich inflection. During preprocessing the text is first filtered against *stop lists* of meaningless, too general or unknown WFs and then analysed morphosyntactically in order to annotate them with a PoS and values of the morphosyntactic categories, e.g. case, gender, number, tense. Moreover, *morphological base forms* (or lemmas, BFs) can be also assigned to WFs in text. In the case of Serbian (cf. [9]), the preprocessing was extended

with syntactic filters (implemented as regular expressions) identifying potential terminology.

There is a limited number of works on Slavic languages ([9, 16, 18]) and only one for Polish: Buczyński's *Kolokacje* system (cf. [1]) is based exclusively on statistical recognition of significantly frequent two-word sequences of WFs.

2 Basic Statistical Recognition

Polish is a language of rich inflection, which means, that a lexeme is (typically) a set of many wordforms, e.g. up to 14 WFs for a noun and even up to 119 for a verb (including participles, gerunds etc.). The application *Kolokacje* (cf. [1, 2]) works on texts in Polish and implements 16 different statistical measures for binary collocations on the level of WFs. The properties of statistical measures were subjects of many studies, so we decided to use one of the measures implemented in *Kolokacje*, and concentrate on the problems of the Polish inflection and free word order.

Contrary to [9], we wanted to keep the first phase of processing, i.e. statistical recognition, as simple as possible. To do that, we apply linguistic filtering in the post-processing, when the possible collocations are already identified. The cost of syntactic analysis of occurrences of selected potential collocations is much lower than the syntactic analysis of the entire corpus. Moreover, we wanted to make the syntactic filtering more automatic. We also wanted to avoid manual construction of the detailed syntactic rules. This is especially difficult because of the free word order in Polish.

The MEs recognition process has been dived into three phases:

1. *reduction of WFs* — all WFs are reduced to BFs,
2. *statistical recognition* — frequent sequences of BFs are identified and are marked *potential collocation*,
3. *statistical syntactic filtering* — frequency of potential collocation matching syntactic constraints is tested and a list of structurally annotated collocations is generated.

Polish wordforms are often ambiguous among several possible BFs, e.g. *mam* can be a WF of the following lexemes (BFs are listed): *mama* ($mummy_{\text{case=gen,num=pl}}$ infml. 'mother'), *mieć* ($to\ have_{\text{person=1st,num=sg,tense=present}}$) and *mamić* ($to\ delude_{\text{imperative}}$). This type of ambiguity can be solved only by analysing the context. For the disambiguation we applied *TaKIPI* — a morpho-syntactic tagger of Polish (cf. [12]). Its accuracy is 93.44%, when measured for all tokens and the complete morpho-syntactic description (86.3% for ambiguous words only). The accuracy of the base form disambiguation has not been measured yet. We can expect it lower, but close to the PoS disambiguation i.e. 98.8% (91.64% for ambiguous words only).

For all experiments, we used the largest corpus of Polish, namely IPI PAN Corpus (henceforth IPIC; [14]). During the reduction phase, all documents of

IPIC, (254 524 624 tokens in total), have been disambiguated by TaKIPI and saved as sequences of BFs. Next, the *Kolokacje* application slightly modified in order to make the processing of so large corpus possible was used for the statistical recognition. A list of potential collocations was produced according to the selected measure. Because of the technical properties of *Kolokacje* we limited ourselves to binary collocations. We tested several measures implemented in *Kolokacje*, achieving the best results (according to the selective manual evaluation) for the *Frequently Biased Symmetric Conditional Probability* (FSCP):

$$R_{FSCP} = \frac{c(w, w')^3}{c(w)c(w')} \tag{1}$$

where w, w' are words, and $c(w)$, $c(w, w')$ are frequencies of a word and a pair, respectively.

FSCP, proposed in [1], produces similar results to *Log Frequency Biased Mutual Dependency*, but is more efficient. As a result, 304,139 binary potential collocations, for which the value of (rounded to third decimal place) FSCP was grater than 0, were identified. The reduction to BFs decreases the complexity of texts by making all different forms of lexemes equal. On the other hand, it can result in accidental association of words that are not syntactically linked, because morpho-syntactic properties of WFs are not expressed on the level of BFs. E.g. after transforming the sentence below to BFs:

- WFs: *Dałem* **długopis czerwony** *koledze.*
  (*I gave a red ballpoint to a colleague.*)
- BFs: *Dać* **długopis czerwony** *kolega.*

there is no information left that *czerwony* (*red*) modifies *długopis* (*a ballpoint*) and not *kolega* (a colleague). But the most unwanted side-effect of this method is that some MEs, which are fixed not only lexically, but also grammatically (mostly verbal and prepositional collocations), can be reduced to unrecognizable BFs. For example three possible noun-noun constructions — both nouns in the same case. i.e. an apposition, e.g. *królowa matka*$_{\text{case=nom}}$ (*queen mother*); noun and subordinate noun in genitive, e.g. *pies*$_{\text{case=nom}}$ *sąsiada*$_{\text{case=gen}}$ (*neighbour's dog*) and noun that has its own requirement, usually inherited from verb in the lexical derivation,e.g. *pomoc*$_{\text{case=nom}}$ *ofiarom*$_{\text{case=dat}}$ *wypadku* (*help for the victims of the accident*) — are presented as the pairs of the same base forms, although the linguistic mechanism of each of those collocations is totally different.This loss of morphosyntactic information can be a problem if such data are used for (theoretical) linguistic purposes.

3 Statistical Syntactic Filtering

The main goal of the filtering phase is to separate accidentally associated pairs of BFs from the ones representing real syntactic units in the corpus. After the manual inspection of potential collocation we identified several classes among them

corresponding to the interesting collocation types, namely: `Adj-Noun`, `Noun-Adj`, `Noun-Noun`, `Prep-Noun`. The last class was introduced experimentally to identify fixed associations of nouns with prepositions (any regularities could be very useful in automatic identification and classification of some types of adjuncts).

Each class is characterised by different syntactic relations between the elements in the pair. It is possible to express these relations by a formal *constraint*, called *constructional constraint* (CC) which must be satisfied for any pair corresponding to the given potential collocation. Let's assume that in the case of the sentence in Sec. 2 two collocations are recognised: *czerwony* (*red*)–*długopis* (*a ballpoint*) of the class `Noun-Adj` and *czerwony* (*red*)–*kolega* (*a colleague*) of the class `Adj-Noun`. In order to find which of the two is really supported by the original sentence, we need to identify the corresponding WFs in text and to check if they agree in number, case and gender (then a CC requires that both have the same value of those three categories). This agreement takes place in the case of the first pair—*długopis czerwony*, but it is absent in the second pair. As each token in IPIC was previously annotated with the morpho-syntactic information by TaKIPI, it is enough to move a *text window* of the size two across IPIC to identify the corresponding pairs of WFs.

The formal tool for expressing the constraints and checking them for pairs of wordforms in IPIC we used is the JOSKIPI language of the syntactic constraints and its implementation in the TaKIPI engine (cf. [11]). The constraints are applied to each position of a text window. Let's take the CC of the class `Noun-Adj` as an example: `agrpp(0,1,nmb,gnd,cas,3)`, where `agrpp` is an operator testing the agreement on number, gender and case between the first and the second WF in the text window.

For each potential collocation $\langle b_i, b_j \rangle$, we need to check if the number of the WF pairs satisfying the appropriate CC, written $CC(\langle b_i, b_j \rangle)$, is significantly large in comparison to some accidental value. We used the standard *t-score test*:

$$\frac{|CC(\langle b_i, b_j \rangle)| - \frac{n}{V}}{\sqrt{\frac{n}{V}}} \tag{2}$$

where n is the number of WFs corresponding to $\langle b_i, b_j \rangle$, and V is the number of possible combinations of values tested by the given CC, e.g. in the case of the CC presented above we have 2 possible numbers, 5 genders and 7 cases, that gives $V = 4900$ combinations—$1/4900$ probability of the CC equals `true` in the null hypothesis; V is specified for each CC separately.

There can be several CCs defined for a class of potential collocations, because pairs of BFs can result from different types of syntactic relations, e.g. in the case of the `Noun-Noun` we distinguished three possible types of syntactic constructions:

1. the second noun is in genitive and modifies the first one, e.g. *służba zdrowia* ($V = 7$):
   `and(equal(cas[1],gen),not(equal(cas[0],cas[1])));`
2. a symmetric construction—the first noun is in genitive ($V = 7$):
   `and( equal(cas[0],gen), not( equal(cas[0],cas[1]) ) );`

3. both nouns are in the same case—(typically these are) proper names, e.g. *Jan Paweł* (*John Paul*) ($V = 49$): `equal(cas[0],cas[1])`.

The constructional properties of a collocation are necessary, but not the only ones features, e.g. *Gwiezdne Wojny* (*Star Wars*) occurs in text only in plural—this is not necessary in the syntax, but results from the semantics. In order to identify such properties we introduced additional set of constraints for each class of potential collocations, called *specifying constraints* (SCs). Each SCs is defined as a template of all possible significant syntactic regularities of WF pairs corresponding to potential collocation. The template is written in the form of a sequence of JOSKIPI operators—$o_1, \ldots, o_k$, and the regularities are statistically significant patterns of values of the operators, i.e. $o_1 = v_{1,i}, \ldots, o_k = v_{k,j}$, e.g. for the `Noun-Noun` class and the second CC above, the following SC is defined: `and(nmb[0],nmb[1])`—we check whether there is a statistically significant pattern of occurrences constraining values of numbers of the two word forms. For example for *Dynamo Kijów* (*Dynamo Kyiv*) such pattern was found automatically—when those word forms co-occure forming ME then both are in singular.

In order to distinguish significant patterns we use the *t-score test* again. The test is limited to WF pairs of a given potential collocation. Such pairs that match the given CC—we look for syntactic regularities only across instances of the given collocation. The null hypothesis is that all patterns of operator values are equally possible. Besides the sequence of operators, each SC is specified with the list of the numbers of possible values of operators— $val(o_1)*\ldots*val(o_k)$, that the corresponding JOSKIPI operator can produce. This list is the parameter of the null hypothesis:

$$\frac{|SC(o_1,\ldots,o_k)| - \frac{|CC(\langle b_i,b_j\rangle)|}{val(o_1)*\ldots*val(o_k)}}{\sqrt{\frac{|CC(\langle b_i,b_j\rangle)|}{val(o_1)*\ldots*val(o_k)}}} \tag{3}$$

where $SC(o_1, \ldots, o_k)$ is an instance of SC, i.e. $o_1 = v_{1,i}, \ldots, o_k = v_{k,j}$, and $|SC(o_1, \ldots, o_k)|$ is the size of the set of WF pairs satisfying this SC instance.

All possible instances of SC (of any subsequence of operators) for the given CC and potential collocation are generated and tested. The instances satisfying the test (with 99.5% confidence) are saved to a file as significant regularities, e.g. for the collocation *mistrzowstwa*$_{\text{num=pl}}$ *świata*$_{\text{num=sg}}$ (*championship*) one significant instance of the SC template: `and(nmb[0],nmb[1])`, was found: `nmb[0]=pl,nmb[1]=sg`.

4 Evaluation

A proper evaluation of the collocations extraction is a permanent problem (cf. [7]), because dictionaries of collocations are created rarely and their coverage is selective and limited. There is no available electronic dictionary of collocations

for Polish. Thus, a sound evaluation process based on the precision and recall calculated in relation to some manually created pattern set is not possible for Polish. Additionally to a limited manual assessment we decided to perform two tests:

1. applying the extracted collocations as a knowledge source in the morphosyntactic disambiguation of Polish — the improved accuracy was expected,
2. comparing the extracted collocations with two lists of two-word lexical units extracted from the electronic source, i.e. [13], and from [15] (by queries on the WWW interface formulated on the basis of WFs from IPIC).

For the needs of the first test, we used a statistical morphosyntactic tagger of Polish (cf. [6]) based on the basic *bi-gram Markov Model* (cf. [7]). The accuracy of this tagger is 91.3% on all words. During the test 102,286 instances of collocations were found. Tagger has accuracy of 94.8% on them. We transformed the extracted collocations and their CCs into rules of morpho-syntactic tag elimination, which removed all tags not fulfilling the CC for found collocations. The rules removed the proper description only in 0.5% of WFs and the number of tags was reduced by 44.4%, the ambiguity was not resolved completely. The accuracy of the tagger was increased to 91.8% on the whole while measured only for the words in collocations to 95.7%. It means that the application of the rules had a positive impact on the work of the whole statistical tagger.

In the second test we used the joint list of two-word lexical units extracted from both dictionaries, i.e. on 8,601 pairs. Next we compared the joint list with the list of extracted collocations. In the case of all classes of collocations, the recall is 46.7%, see Tab. 1. In the case of the classes `Adj-Noun` and `Noun-Adj`, it is difficult to calculate the exact value of the recall because for WFs in dictionaries no Part of Speech is assigned. However, on the basis of manual inspection, it is much higher than in the first case. In both cases precision is low — quite expectable result for the small general dictionary as a source of collocations.

Table 1. Comparison of the extracted collocations with the two dictionaries.

	collocations	common	missed
all classes	682,454	4,015	4,586
`Adj-Noun` and `Noun-Adj`	338,467	3,360	—

We decided to take a look into extracted ME, but because their number is very large we have selected a random sample for each class of ME for evaluation by a qualified linguist (one of the co-authors). Size of the samples has been determined using tables from [4]. Population size was rounded up to values chosen by Israel. Using assumption of 95% confidence level he used the following formula: $n = \frac{N}{1+N(e)^2}$ (n is sample size, N is population size and e is desired level of precision). Together 3,149 out of total 94,558 ME were rated (see table 2).

All selected collocations were analysed manually and assigned to six types:

Table 2. Size of samples in relation to multiword expression types. Precision level is 5% and confidence level is 95%. Sum_N is number of merged test cases within broader class.

Class	Adj-Verb	Noun-Adj	Noun-Noun	Noun-Verb	Verb-Adj	Verb-Noun
Sum_N	512	23914	55278	5310	628	8916
Sample size	308	394	1122	552	323	450
N	21.75%	11.93%	30.48%	41.12%	6.81%	23.33%

Table 3. Results [%] of evalutaion.

Class	Adj-Verb	Noun-Adj	Noun-Noun	Noun-Verb	Verb-Adj	Verb-Noun
N	21.75	11.93	30.48	41.12	6.81	23.33
B	52.27	12.44	24.15	23.55	78.33	6.67
NW	0.00	5.33	11.05	0.18	0.00	0.00
Nwb	0.00	0.25	0.89	0.00	0.00	0.00
K	7.47	55.84	0.53	21.20	3.72	58.00
Kb	16.88	1.78	30.57	13.04	9.60	9.11
F	0.65	12.44	1.52	0.18	0.31	1.78
Fb	0.97	0.00	0.80	0.72	0.93	0.89

- B — error, (e.g. "Al-Kaida" separated in two words),
- K — real collocations,
- Kb — real collocations but with some grammatical properties not described,
- NW — proper names (collocations, too),
- NWb — proper names but with some grammatical properties not described,
- N — insignificant or accidental association (originating in the unbalance of the corpus),
- F — phraseology,
- Fb — phraseology with some grammatical properties not described.

ME of the type `Prep-Noun` were excluded form the manual evaluation, as we had not expected to find any significant real collocations. Our aim here was to identify some more significant associations of prepositions and nouns that can be used during tagging in order to disambiguate the case in both constituents. ME of this type were applied as rules during tests with the tagger.

In these ME in which both constituents are associated by morpho-syntactic agreement (involving adjectives and nouns), the number of wrongly associated word forms is very low. However, if an extracted ME is only the result of co-occurrence in sequence in the text, the number of errors and insignificant associations is high. It is important that on average, a significant percentage of the extracted 'collocations' are just stronger syntactic-semantic associations of the type N, e.g. *złamane ramię* (*a broken arm*), *wymóg religii* (*a religion requirement*), or *gwałtowna fala* (*an instantaneous rapid wave*). Such pairs are not real collocations according to the definitions, but are very useful in text processing,

Table 4. Results [%] summed in groups.

Class	Adj-Verb	Noun-Adj	Noun-Noun	Noun-Verb	Verb-Adj	Verb-Noun
K+Kb	24.35	57.61	31.11	34.24	13.31	67.11
F+Fb	1.62	12.44	2.32	0.91	1.24	2.67
NW+Nwb	0.00	5.58	11.94	0.18	0.00	0.00
N+B	74.03	24.37	54.63	64.67	85.14	30.00
K+Kb+NW+Nwb	24.35	63.20	43.05	34.42	13.31	67.11
K+Kb+F+Fb	25.97	70.05	33.42	35.14	14.55	69.78
K+Kb+NW+Nwb+F+Fb	25.97	75.63	45.37	35.33	14.55	69.78

e.g. in the morpho-syntactic disambiguation. Moreover, many pairs of the type N are accidental associations of an adjective describing a colour, geographic origin, time or those, which have very general meaning like *mały* (*small*), *nowy (new)*, etc. Such pairs can be eliminated by simple additional post-processing.

5 Conclusions

If we take into account the raw numbers of precision and recall we can say that the approach failed. However, we have to consider that the used dictionaries are general and quite small. Discovering general collocations in a large general corpus is very difficult, especially in the case of an inflective language like Polish. Application of this method to some domain corpus could result in better figures. Errors caused by the *TaKIPI* are very seldom, one of them is *Al-Kaida*, that was mistakenly separated and interpreted as the association of two nouns.

The application of all extracted collocations in morpho-syntactic disambiguation was quite successful. Moreover, the manual inspection of the extracted collocations showed that in spite of the substantial number of false collocations observed it is still relatively easy to notice the real ones and separate them by editing. Thus, the created tool can be used for semi-automatic collocation extraction.

The syntactic was extracted so the manual work was reduced significantly in comparison to other approaches, e.g. [9]. We did not have to create detailed syntactic rules. Especially, the automatic check of SCs brought interesting results with the minimal human effort. Moreover, the processing is quite efficient — results for very large corpus are processed on an average conteporary PC in less than a day.

In further research we will concentrate on the reduction of the nominal pairs. This goal can be achieved by elimination of the associations with too general adjectives based on information theory and by the application of a semantic stop list including adjectives expressing time or geographical origin. We want to introduce the additional measures based on the contexts in which the given pair is used.

The extracted ME and the method can be also applied in OCR correction of handwriting or in Speech Recognition in a similar manner to morpho-syntatic

tagging task. In postprocessing phase one can directly use collocations and their syntatic descriptions to correct errors in recogniotion of multiword expressions. Our long-term goal is the extraction of lexical units for the needs of semi-automatic extension of a lexicon.

Acknowledgement. Work financed by the Polish Ministry of Education and Science, project No. 3 T11C 018 29.

References

1. Buczyński A.: *Pozyskiwanie z internetu tekstów do badań lingwistycznych,* Msc thesis, Wydz. Mat., Inform. i Mech., Uniwersytet Warszawski (2004).
2. Buczyński A., Okniński T.: *Program Kolokacje,* `http://www.mimuw.edu.pl/polszczyzna/kolokacje/` (2006).
3. Derwojedowa M., Piasecki M., Szpakowicz S., Zawisławska M.: *plWordNet—the Polish Wordnet,* WWW: `http://plwordnet.pwr.wroc.pl` (2007).
4. Israel G.: *Determining Sample Size,* University of Florida Tech. Rep., 1992.
5. Jacquemin C.: *Spotting and Discovering Terms through Natural Language Processing,* The MIT Press (2001).
6. Kukła P.: *Tager dla języka polskiego oparty na kombinacji metod statystycznych,* Msc thesis, Wydz. Inf. i Zarządz., Politechnika Wrocławska (2007) In preparation.
7. Manning C. D., Schütze, H.: *Foundations of Statistical Natural Language Processing,* The MIT Press (2001).
8. Moirón V. M. B.: *Data-driven identification of fixed expressions and their modifiability,* PhD thesis, Rijksuniversiteit Groningen (2005).
9. Nenadić G., Spasić I., Ananiadou S.: *Morpho-syntactic clues for terminological processing in Serbian,* In: Proceedings of Workshop on Morphological Processing of Slavic Languages, EACL 2003, Budapest, Hungary (2003).
10. Pecina P.: *An extensive empirical study of collocation extraction methods,* In: Proceedings of the ACL Student Research Workshop, Ann Arbor, Michigan, Association for Computational Linguistics (2005) 13–18.
11. Piasecki M.: *Hand-written and Automatically Extracted Rules for Polish Tagger,* In Sojka, P. et. al. (ed.) Proc. of the Text, Speech and Dialog 2006 LNAI, Springer (2006).
12. Piasecki M., Godlewski G.: *Effective architecture of the Polish tagger,* In Sojka, P. et. al. (ed.) Proc. of the Text, Speech and Dialog 2006 LNAI, Springer (2006).
13. Piotrowski T., Saloni Z.: *Kieszonkowy słownik angielsko-polski i polsko-angielski,* Wyd. Wilga, Wrszawa (1999).
14. Przepiórkowski A.: *The IPI PAN Corpus Preliminary Version,* Institute of Computer Science PAS (2004).
15. PWN: *Słownik języka polskiego,* Published on WWW: `http://sjp.pwn.pl/` (2007).
16. Sharoff S.: *What is at stake: a case study of Russian expressions starting with a preposition,* In Tanaka T., Villavicencio A., Bond F., Korhonen A., eds.: Second ACL Workshop on Multiword Expressions: Integrating Processing, Barcelona, Spain, Association for Computational Linguistics (2004) 17–23.
17. Smadja F.: *Retrieving collocations from text: Xtract,* Computational Linguistics **19**(1) (1993) 143–177.
18. Spasic I.: *A Machine Learning Approach to Term Classification,* PhD thesis, Information Systems Research Centre School of Computing, Science and Engineering University of Salford, Salford, UK (2004).

Proceedings of the International Multiconference on Computer Science and Information Technology pp. 247–256
ISSN 1896-7094
© 2007 PIPS

Polish Morphological Guesser Based on a Statistical *A Tergo* Index

Maciej Piasecki and Adam Radziszewski

Institute of Applied Informatics, Wrocław University of Technology,Poland
Wybrzeże Wyspiańskiego 27, Wrocław, Poland,
maciej.piasecki,adam.radziszewski@pwr.wroc.pl,

Abstract. We present a direct method of construction of a morpho-syntactic guesser for Polish, which is a program producing morpho-syntactic descriptions for word forms unknown to the morphological analyser. The core of the method is the construction of a statistical *a tergo* index, in which pseudo-suffixes (endings) extracted by a statistical tree define morpho-syntactic properties of corresponding word forms. The secondary aim was to investigate to what extent it is possible to develop the morphological analyses exclusively on the basis of endings. Experiments in the extraction of a guesser for a domain of texts are also presented. The method can be applied to any other inflectional language with only minor technical changes.

Keywords: morphological guesser, Polish, automatic extraction, corpus linguistics, statistical a tergo index

1 Introduction

Morfeusz [1], which is one of the most advanced morphological analysers for Polish, recognises about 1 700 000 word forms. However, in the case of a small domain corpus of medical texts *KorMedIIS* [2], it covers only about 75% tokens occurring there. This due to the following reasons: there are many misspelled words and domain-specific terms. Tokens from both of these categories are not worth to be included in a general tool like *Morfeusz*, but assigning them to one single class of *unknown words* blur the description for language tools of subsequent levels of processing e.g. decreases the accuracy of OCR correction done with the help of a tagger [3].

In the case of English, the initial approaches to guessing a morpho-syntactic tag for an unknown word solely on the basis of its ending (trailing) and also some leading segments, e.g. [4] were next exchanged by extraction of rules expressing patterns of a morphological construction of word forms, e.g. [5–7], or on the basis of *Inductive Logic Programming* [8].

Methods based on extraction of rules, as long as they do not use probabilistic criteria inside the rules (statistics is always present in the algorithms of extraction), are in their expressive power equivalent to finite state automaton

construction proposed in [9], where several heuristic rules of automaton reduction are proposed in order to get a generalised description of the characteristic features of word form construction.

However, the rule-based approaches originate from the works done for English which displays limited inflection. In the case of an English word form ending one can say mostly a little about its morpho-syntactic properties. The situation is completely different in the case of an inflectional language like Polish. The morphological analyser SAM-95 [10] has been constructed on the basis of a linguistic index *a tergo* of Polish verbs of Tokarski [11] (recently published). Tokarski's index consists of manually defined suffixes of verbs which identifies particular morphological descriptions. On this basis one can assign a description to an unknown verb. SAM-95 covering all parts of speech is able to do this for any word, but it assigns tags to unknown word forms too eagerly.

Our main idea is to correct this over-generation by extracting a kind of statistical index *a tergo* from a large corpus, and as a consequence to take into account only the tags supported by the data during guessing. Moreover, we can use the collected statistics concerning pseudo-suffixes (elements of the statistical index *a tergo*) and assign tags to them. This work also originates from the need to construct a guesser for a narrow domain by means of limited man power.

It is hard to find works on morphological guessers for Polish, especially based on a statistical index *a tergo*. A Polish guesser based on the automatic extraction of endings was presented in [12], but endings of the maximum fixed size were used and the coverage was limited to a subset of word forms covered by the extracted endings. Concerning similar inflectional languages, [13] deals with a different problem of the lemmatisation combined with the morpho-syntactic disambiguation. In [14] suffixes of a definite end are used and there is no information on the accuracy in relation to tags; only the coverage of unknown words is given. Hungarian, the target language of [7], according to its agglutinative character is significantly different from Polish.

The goal of this work was to construct a robust morphological guesser for Polish on the basis of a statistical extraction of an index similar to the index *a tergo* of Tokarski [11]. The guesser should be easily portable to different domains of text by using as simple and as direct methods as possible. Moreover, we wanted to investigate experimentally to what extent it is possible to reconstruct a morphological description of a word form exclusively on the basis of a statistically extracted index *a tergo*.

2 Construction of Pseudo-suffix Tree

The main idea is to identify endings of word forms associated with particular morpho-syntactic descriptions. As a model of such descriptions we assume the set of tags of the IPI PAN Corpus (IPIC) [15]—the largest corpus of Polish, containing about 254 millions of words in the version 2.0. The model was implemented in the morphological analyser *Morfeusz* [1], which was also employed

in the training phase[1]. In IPIC format, a word form is assigned a set of pairs consisting of morphological *base form* and a *tag* expressing morpho-syntactic properties.

The construction of guesser for a domain of texts is performed in three steps:

1. Construction of a raw *a tergo tree*
2. Pruning orphaned branches (introduces generalisation)
3. Assigning morphological information to tree nodes

To gain higher degree of data generalisation, additional pruning techniques can be applied after the last step. Some experiments with different heuristics and parameter values have been carried out, yet no significant improvement in the overall results could be observed.

2.1 Construction of a raw tree

Word forms needed for the training process are acquired along with their frequencies from the IPIC corpus. These forms are passed through Morfeusz. Unrecognised forms are immediately removed from the training set.

In IPIC a notion of *grammatical class* is introduced [15] for more fine grained division than into parts of speech. There 32 grammatical classes defined in the IPIC format. The analyses for the extracted word forms are filtered according to the predefined list of 'open' grammatical classes. The list includes (the mnemonics are taken from the IPIC format): `subst` (noun), `depr` (depreciative form of noun), `adj` (adjective), `adjp` (post-prepositional adjective), `adv` (adverb), `fin` (non-past form of verb), `praet` ("l-participle", 'past form'), `impt` (imperative form), `imps` (impersonal form), `inf` (infinitive), `pcon` (contemporary adverbial participle), `pant` (anterior adverbial participle), `ger` (gerund), `pact` (active adjectival participle), and `ppas` (passive adjectival participle). The `adja` (ad-adjectival adjective) was omitted, as word forms of this class are recognised by *Morfeusz* only in context, not on the basis of a single word form alone. We assume full description of other grammatical classes by *Morfeusz* or a lexicon.

The training process leads to the construction of an *a tergo tree*, i.e. a tree-shaped automaton of reversed word forms whose edges are marked with letters. Fig. 1 shows an *a tergo tree* built with these forms:
mężczyzna ($men_{case=nom,num=sg}$), *mężczyzny* ($men_{case=gen,num=sg}$),
zna ($know_{tense=pres,per=1st,num=sg}$), *oka* ($eye_{case=gen,num=sg}$), *kózka* (*young female goat*$_{case=nom,num=sg}$). The '!' marks a node where a form ends without further branching.

Every word form is reversed and along with its frequency passed to the tree. This way the branches are successively built, resulting in a tree-like representation of the filtered training data. To enable generalisation in the guessing process we need to introduce pruning.

[1] We would like to thank dr. Marcin Woliński and dr. Adam Przepiórkowski for their kind acceptance for our experiments done on IPIC with the help of *Morfeusz*.

```
(5)
   -a-> (4)
           -k-> (2)
                   -o-> (1)
                   -z-> (1) -ó-> (1) -k-> (1)
           -n-> (2)
                   -z-> (2!) -yzcżęm-> (1)
   -ynzyzcżęm-> (1)
```

Fig. 1. An example *a tergo* tree.

2.2 Pruning orphaned branches

The described method enables generalisation of the classification process without precision loss on training data. In inflectional languages it is the word ending that conveys most of the morphological information. Thus, we can abstract from particular prefixes when they seem irrelevant to determine the morphological analysis.

This is achieved by a recursive tree traversal. The leaves are pruned as long as they belong to non-branching paths (i.e. a leaf is subjected to pruning when it is the only child of its father). This operation corresponds to removing prefixes of stored word forms where the prefixes are not needed to distinguish between different analyses.

2.3 Assigning morphological information to tree nodes

The training set is employed again in order to assign morphological analyses (tags) to the nodes of the pruned tree. Those nodes are found by following the paths corresponding to consecutive letters of the inverted form (this process is identical with the guessing stage and will be described in the next section). The set of tags assigned to processed form (coming from Morfeusz) is being added to the node. Every tag is associated with its frequency; it is calculated by summing the frequencies of the processed forms having this tag.

We also need to reconstruct the base form generation rule using the encountered form. We attach such a rule to every tag in the node. It consists of the encountered form ending and the length of base form ending. The following example explains the heuristics:

Encountered form: *przepustowościach* ($capacity_{case=loc,num=pl}$)
Base form: *przepustowość* ($capacity_{case=nom,num=sg}$)
Base form ending: *ć*
Encountered form ending: *ciach* (5 letters)
Procedure to get the base form: substitute the last 5 letters from the encountered form with "*ć*"

The first experiments produced relatively low accuracy of generated base forms. The analysis of the data showed that most of the incorrectly restored

base forms belonged to negated participles. This was due to the rules used in IPIC: the negated forms of participles beginning with "*nie*" (*not*) are lemmatised to base forms without the prefix. The negation is included as a value `neg` of the attribute `negation` in the tag. For instance the form *niemyty* (*not cleaned*) has base form of *myć* (*to clean*). This approach leads to some problems as basing on the mere ending we cannot tell if the form is negated or not.

The adopted solution uses a heuristic, which tries to convert all negated forms to their affirmative equivalents. If the attribute `negation` of a tag is set to `neg` and the form begins with "*nie*", this prefix is stripped off and the value of `negation` is set to `aff` (affirmative) before storing the tags in a tree node. The correct value of negation is evaluated by a similar heuristic when guessing. By using this approach we can avoid the unnecessary division of some data into affirmative and negative entries which is no longer meaningful to the guesser (as the value of the negation attribute needs to be restored anyway).

3 The Algorithm of Guessing

The form being guessed is inverted first. Then we follow the branches of the tree corresponding to consecutive letters of the inverted form until the tree path or the inverted form ends. In both cases a node is reached. If this node has tags attached, these tags are treated as analyses of the form. If not, the tags of its ancestor are taken instead.

We need to reconstruct base forms as well; this is achieved by using the rules attached to every tag. Let us assume that we have been guessing the form "niezwykłościach". We have obtained the tags from the tree and one of them contains the ending "*ć*" and the length of "*ciach*", i.e. 5. Then:

Encountered form: *niezwykłościach* ($singularity_{case=loc,num=pl}$)
Length of encountered form ending: 5 (i.e. 5 letters to strip off)
Base form ending: *ć*
Reconstructed base form: "*niezwykłoś*" + "*ć*" = "*niezwykłość*"

If the analysed form begins with "*nie*" its tags are checked. The tags containing the attribute `negation` set to `aff` are subjected to negation restoration heuristics: their `negation` is set to `neg` and the "*nie*" prefix is stripped off from the reconstructed base form.

4 Experiments

The experiments were performed on a list of word forms recognised by *Morfeusz* and collected from two corpora: IPIC [15] — 504 320 word forms of the appropriate grammatical classes (see Sec. 2) and KorMedIIS [16] — 26 070. Documents in KorMedIIS are short medical electronic documents — descriptions and contain a limited vocabulary repetitively used across different documents. IPIC is the largest corpus of Polish containing various genres. The word forms were stored together with their frequencies in a corpus.

The test presented in Table 1 was performed following the *ten-fold scheme*, i.e. the list of word forms were randomly divided into 10 parts, nine were used in training, with one kept for tests, and this procedure was repeated ten times. Word forms from the testing part are unknown to the guesser but known to *Morfeusz*. Precision, recall and F-measure were calculated on the level of tags/base forms in relation to tags/base forms assigned to word forms in the test set, i.e.:

$$P = \frac{T_G \cap T_T}{T_G}, R = \frac{T_G \cap T_T}{T_T}, F = \frac{2PR}{P+R} \quad (1)$$

where T_G is the set of tags/base forms returned by the guesser for all word forms in the test part altogether, T_T — the set of tags/base forms collected from word forms in the training parts.

All the results presented in Table 1 are calculated as an average from ten tests, including the reported numbers (i.e. average numbers) of tested word forms (*Frm.*) and non-recognised (*Non-rec.*).

For some applications of guesser only the morpho-syntactic description is important (e.g. language model on the level of tag trigrams); for others only base forms are necessary (e.g. calculation of semantic similarity of documents). We have performed three kinds of tests:

- only morpho-syntactic tags: grammatical class (Part of Speech) plus morpho-syntactic categories, regardless of the base form (the label *Tags* in Table 1),
- only base forms, regardless of the tag (*Bases* in Table 1),
- and tags together with base forms (*Tags+Bases*), i.e. a test of full descriptions in the IPIC format.

During tests across different test parts, the results are stable and the differences lie in the range of $\pm 2\%$

In the row *all* of Table 1, the overall results calculated for all word forms are presented. The result for tags is much lower than reported for English, but the number of tags being recognised is more than ten times larger. It is worth to emphasize that our intention was to analyse only the endings of words (with one exception described in Sec. 2.3) and to investigate the ability of the mere endings to differentiate among word form morpho-syntactic properties. The higher value of recall means that the guesser generates two many descriptions per tested word, as it finds the closest match instead of the exact one in those cases in which there is no exact match in the tree. The result is no doubt better than a baseline of a random choice (1642 possible choices) and manual assessment of the guesser is quite positive as the errors concern mostly grammatical categories other than the basic set: case, gender, number and person. The assignment of grammatical classes to analysed word forms seems to produce a significantly smaller number of errors than the general result for tags. The number of word forms which were not recognised is very small (about 0.6% of all).

The result is also lower than the precision of 91.5% reported in [12], but it is very hard to compare our approach with [12] as there: only words started with a lower case letter were analysed (in our approach, all word forms are converted

Table 1. Average results of the guesser in ten-fold test: *Tags* — only morpho-syntactic description evaluated, *Bases* — only base form evaluated, *Tags+Bases* — full IPIC tags evaluated, *Non-rec.* — the average number of non recognised word forms, *Frm.* — the average number of word forms across test folds.

PoS	Tags [%]			Bases [%]			Tags+Bases [%]			Non-rec.	Frm.
	R	P	F	R	P	F	R	P	F		
all	81.75	70.32	75.61	88.93	81.49	85.05	79.77	68.61	73.77	282.5	50697.0
subst	74.96	58.21	65.53	83.61	70.72	76.63	73.87	57.37	64.58	1070.7	20497.3
depr	52.34	34.88	41.82	51.12	34.07	40.85	51.12	34.07	40.85	106.3	224.7
adj	88.63	82.61	85.51	87.28	79.87	83.41	87.7	81.74	84.61	1042.8	10594.6
adjp	93.26	80.47	86.13	93.26	80.47	86.13	93.26	80.47	86.13	2.2	32.4
adv	74.64	64.52	69.19	77.69	67.53	72.24	73.04	63.14	67.72	53.0	401.2
fin	69.49	55.01	61.40	89.65	83.44	86.43	66.03	52.27	58.34	217.6	3576.3
praet	71.54	56.39	63.07	96.43	95.94	96.19	69.8	55.02	61.54	58.9	4373.9
impt	69.41	53.69	60.53	85.68	77.65	81.46	63.43	49.07	55.32	52.3	765.2
imps	72.70	58.56	64.86	97.77	97.59	97.68	71.46	57.55	63.75	4.6	598.8
inf	68.66	55.58	61.42	98.74	98.94	98.84	68.62	55.55	61.39	13.0	1150.3
pcon	97.43	97.13	97.27	89.37	90.15	89.76	87.93	87.67	87.80	1.3	451.0
pant	98.00	97.93	97.96	96.53	97.03	96.77	96.11	96.04	96.07	2.2	161.5
ger	73.27	61.58	66.92	94.32	94.00	94.16	71.66	60.23	65.45	70.8	3371.3
pact	95.68	95.58	95.63	88.80	89.52	89.16	86.33	86.23	86.28	16.8	2777.0
ppas	75.69	61.89	68.10	96.25	95.41	95.83	74.87	61.22	67.36	144.8	5642.5

to lower case first), the precision for words starting with an upper case letter was only 43.6%, the precision was calculated only for 71% of unknown words recognised by the guesser of [12], and finaly, the tagset used in [12] consists of only about 400 tags (four times less than the KIPI tagset).

The result of the base form generation is much better than of tag guessing and it is balanced between precision and recall. It is hard to compare it with other approaches, as it is hard to find data in the literature. This problem is very specific for inflective languages. The error rate of tag plus base form recognition is only slightly higher than the error rate for tags alone. It could be expected, as the wrong choice of tag implicates the wrong identification of the cut point.

In order to analyse the information conveyed by endings in different grammatical classes, we calculated the results for the classes, as well, see Table 1. The rows are named with IPIC class mnemonics described in Sec. 2.1. The numbers of non-recognised word forms are higher in the results of classes, as only tags of given case were taken into account as the expected outcome of recognition.

The worst result was achieved for depreciative forms (`depr`); also generation of base forms works poorly. Probably the number of learning cases is too small and they share endings with nouns too often to be properly distinguished. A similar problem can be noticed in the gerund class (`ger`). The smallest error rate is produced for: post-prepositional adjectives (`adjp`), and adverbial participles (`pcon,pant`) — these classes have a limited set of possible endings.

Table 2. Statistics: the length of an ending (*Len.*) needed for the recognition of a base form; all data are given in percentage [%] of endings of the given class.

Len.	subst	depr	adj	adjp	adv	fin	praet	impt	imps	inf	pcon	pant	ger	pact	ppas
1	0,41							0,02							
2	3,80		2,39		0,80	8,19		0,29							
3	7,39	1,16	7,05	0,01	13,91	2,61	4,95	33,78		10,23	0,77				
4	15,18	5,79	9,40	4,67	14,59	24,69	13,05	34,14	0,60	7,07	7,68			0,41	0,82
5	19,22	17,48	13,92	13,90	15,39	14,80	12,30	21,22	7,51	9,07	15,98	0,13	2,47	1,35	3,47
6	17,84	18,05	14,91	79,37	14,61	18,51	19,17	4,39	8,83	17,93	13,30	1,80	7,51	6,42	6,94
7	12,60	19,97	14,12	1,59	11,29	11,63	18,62	3,26	20,68	20,21	13,00	8,98	11,73	8,57	10,35
8	7,64	24,89	11,45	0,29	12,45	12,37	13,85	1,53	18,02	17,02	16,29	11,22	12,90	14,43	15,26
9	6,34	7,89	10,02	0,10	7,51	4,03	9,15	0,87	18,25	7,73	14,71	47,17	16,46	11,02	18,69
10	4,56	1,42	6,42		5,05	2,45	5,70	0,37	15,52	7,22	11,58	15,61	20,86	19,18	15,30
11	2,26	1,53	5,15		2,57	0,59	1,99	0,10	7,11	1,71	3,90	7,83	11,50	11,34	14,28
12	1,63	0,27	2,43	0,07	1,06	0,10	0,98	0,03	2,31	1,48	2,64	2,97	10,22	12,94	6,86
13	0,43	1,53	1,54		0,11	0,02	0,18		0,84	0,27	0,12	3,96	3,14	6,41	3,55
14	0,43		0,74		0,62		0,05		0,25	0,05	0,01	0,31	2,17	4,90	3,01
15	0,14	0,01	0,35		0,02		0,01		0,06	0,02	0,01	0,01	0,89	1,81	0,92
16	0,03		0,08						0,01				0,14	1,17	0,39
17	0,09		0,04						0,01				0,02	0,03	0,09
18	0,01													0,01	0,05
19			0,01											0,01	0,01

In Table 2 a histogram of ending lengths is presented. The length of an ending is taken from the tree as the length of a tree path leading to the first node in which a decision can be unambiguously made. The data are collected from the tree built for all word forms collected from both corpora, i.e. they are generated during learning on full data (any form could be perfectly recognised) and they describe the regularity observed in mapping: morpho-syntactic description—word form. It is hard to find any correlation between results of recognition and the length histogram. For some more specific classes, clear cut points can be observed, e.g. post-prepositional adjectives (`adjp`), anterior adverbial participle (`pant`) and imperative form (`impt`). In general, the automatically generated endings are much longer than reported in linguistic literature, but here the automatic endings are by definition unambiguous in the set of 504 320 word forms.

Some preliminary experiments on the application of the guesser in the improvement of handwriting OCR showed a significant positive impact. It is worth to notice, that the result for a domain corpus is much higher, especially the precision is improved. It means that a guesser constructed for the given domain generates descriptions better matching the vocabulary of given domain.

In the next experiment we have tested the influence of the learning set size on the results of guesser. In Fig. 2, one can notice that the result is increasing with the increase in the size — linearly in logarithmic scale. We expected this, as there were only 504 320 different word forms used in the experiment in relation to about 1 700 thousands of word forms known to *Morfeusz*.

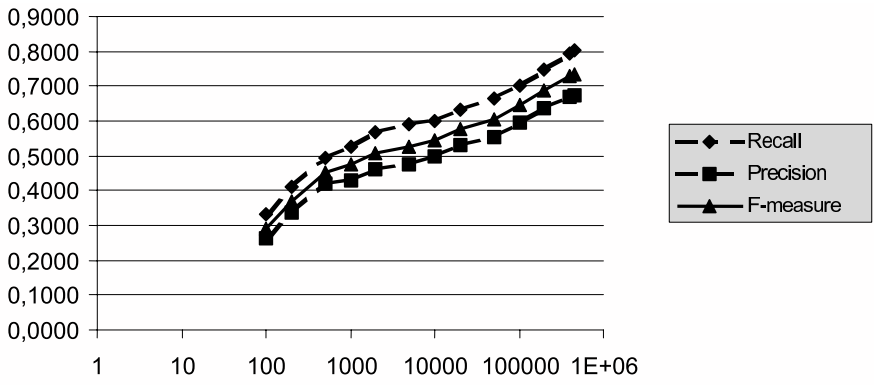


Fig. 2. Incremental test of the guesser: results in relation to the size of learning set (the horizontal axis).

5 Conclusions

The achieved results are much lower than reported for French [9] or English [5], but there are 1642 tags used in the IPIC, i.e. more than ten times more than in an English corpus. It is worth to emphasize that the coverage of the guesser is very good especially in comparison to [14], it leaves unrecognised only about 0.6% of words. The precision is lower than the recall, yet this means a little over-generation of tags. The good result for the very large general corpus of Polish—IPIC, was achieved on the basis of less than 30% of theoretically possible word forms according to *Morfeusz*. The result for a small domain corpus—KorMedIIS is even higher, probably due to capturing some patterns of inflection specific for the given corpus. Statistical information kept in guesser and the tree of pseudo-suffixes facilitate defining different strategies of generalisation by pruning. These positive results were achieved by application of a straightforward method, especially simple in the case of generating the base forms. The method can be applied to any other inflectional language with only minor technical changes concerning the input format of the learning data.

The necessity of the introduction of a heuristic for negative participles shows the limitation of the statistical *a tergo* index. In order to improve the result, one needs to add some mechanism sensitive for the leading segments of word forms. The created tree could be also transformed into an automaton by the application of a solution proposed in [9].

Acknowledgement. This work was financed by the Ministry of Education and Science project No 3 T11E 005 28.

References

1. Woliński M.: *Morfeusz — a practical tool for the morphological analysis of Polish,* [18].
2. Piasecki M., Godlewski G., Pejcz J.: *Corpus of medical texts and tools,* In: Proceedings of Medical Informatics and Technologies 2006, Silesian University of Technology (2006) 281–286.
3. Godlewski G., Piasecki M.: *Application of syntactic properties to three-level recognition of Polish hand-written medical texts,* In Brailsford D. F., ed.: Proceedings of 2006 ACM Symposium on Document Engineering, ACM (2006) 115–121.
4. Brill E.: *Some advances in transformation-based part of speech tagging,* In: Proceedings of the Twelfth National Conference on Artificial Intelligence (AAAAI-94). (1994).
5. Mikheev A.: *Automatic rule induction for unknown-word guessing,* Computational Linguistics **23**(3) (1997) 405–423.
6. Schone P., Jurafsky D.: *Knowlege-free induction of inflectional morphologies,* In: Proceedings of the North American chapter of the Association for Computational Linguistics (NAACL-2001). (2001).
7. Novák A., Nagy V., Oravecz C.: *Combining symbolic and statistical methods in morphological analysis and unknown word guessing,* In: Proceedings of LREC'04. (2004).
8. Kazakov D.: *Achievements and prospects of learning word morphology with inductive logic programming,* In: Learning Language in Logic. LNCS, Springer (2004) 89–111.
9. Daciuk J.: *Treatment of unknown words,* In: Proceedings of Workshop on Implementing Automata WIA'99, Potsdam, Germany, 1999. Volume 2214 of LNCS., Springer Verlag (2001) 71–80.
10. Szafran K.: *SAM-96 — the morfological analyser for Polish,* In Narin'yani, A., ed.: Proceedings of International Workshop DIALOGUE'97 Computational Linguistics and its Applications, Yasnaya Polyana, Russia, June, 10–15, 1997, Moskwa (1997) 304–308.
11. Tokarski J.: *Schematyczny indeks* a tergo *polskich form wyrazowych,* Warszawa (2002).
12. Rudolf M.: *Metody automatycznej analizy korpusu tekstów polskich,* Uniwersytet Warszawski, Wydz. Polonistyki (2004).
13. Džeroski S., Erjavec T.: *Learning to lemmatise slovene words,* In Cussens J., Džeroski S., eds.: Proceedings of LLL'99. Volume 1925 of LNAI., Springer (2000) 69–88.
14. Hlaváčová J.: *Morphological guesser of Czech words,* In: Proceedings of Text, Speech, and Dialogue 2001. Volume 2166 of LNAI., Springer (2001) 70–75.
15. Przepiórkowski A.: *The IPI PAN Corpus Preliminary Version,* Institute of Computer Science PAS (2004).
16. Godlewski G., Piasecki M., Pejcz J.: *Corpus of medical texts and tools,* Silesian University of Technology (2006) 273–280.
17. Sojka P., Kopecek I., Pala K., eds.: *Proceedings of the Text, Speech and Dialog 2006 Conference,* Lecture Notes in Artificial Intelligence, Springer (2006).
18. Kłopotek M. A., Wierzchoń S. T., Trojanowski K., eds.: *Intelligent Information Processing and Web Mining—Proceedings of the International IIS: IIPWM'06 Conference held in Zakopane,* Poland, June, 2006. Advances in Soft Computing. Springer, Berlin (2006).

Proceedings of the International Multiconference on
Computer Science and Information Technology pp. 257–266
ISSN 1896-7094
© 2007 PIPS

Computational Efficiency of Suboptimal Nonlinear Predictive Control with Neural Models

Maciej Ławryńczuk

Institute of Control and Computation Engineering
Warsaw University of Technology
ul. Nowowiejska 15/19, 00-665 Warszawa, Poland
tel. +48 22 234-73-97
M.Lawrynczuk@ia.pw.edu.pl

Abstract. This paper studies computational efficiency of suboptimal Model Predictive Control (MPC) with neural models. The algorithm requires solving on-line only a quadratic optimisation problem. Considering a nonlinear polymerisation process, for which the linear MPC algorithm is inadequate, it is shown that the suboptimal algorithm results in closed-loop control performance similar to that obtained in the fully-fledged nonlinear MPC technique, which hinges on non-convex optimisation.

1 Introduction

Model Predictive Control (MPC) is recognised as the only advanced control technique which has been very successful in practical applications [2, 8, 13–15]. It is largely because MPC algorithms can take into account constraints imposed on both process inputs (manipulated variables) and outputs (controlled variables), which usually decide on quality, economic efficiency and safety. Moreover, MPC techniques are very efficient in multivariable process control.

Because properties of many technological processes are nonlinear, different nonlinear MPC techniques become more and more popular [2, 13, 14]. In view of practical importance, both accuracy and computational efficiency determine reliability of any nonlinear MPC algorithm. These factors are influenced by the structure of the model and the way it is used in the control algorithm. Fundamental models, although potentially very precise, are usually not suitable for on-line control because they are very complicated and may lead to numerical problems. Since neural network models [1] are able to approximate precisely nonlinear behaviour of technological processes [3, 10], have relatively small number of parameters and simple structure, they can be effectively used in MPC algorithms as process models [4–7, 10–12, 14, 15].

This paper investigates computational efficiency of a suboptimal MPC algorithm with Nonlinear Prediction and Linearisation (MPC-NPL) [5–7, 14, 15] with neural models of processes. Unlike the MPC technique with Nonlinear Optimisation (MPC-NO), which hinges on on-line non-convex optimisation, the algorithm

needs solving on-line only a quadratic optimisation problem. At the same time, in practice the suboptimal algorithm gives control performance similar to that obtained in the MPC-NO one.

2 Model Predictive Control Algorithms

In the MPC algorithms [8, 14] at each consecutive sampling instant k a set of future controls

$$\boldsymbol{u}(k) = [u(k|k)\ u(k+1|k) \ldots u(k+N_u-1|k)]^T \tag{1}$$

or corresponding increments

$$\Delta\boldsymbol{u}(k) = [\Delta u(k|k)\ \Delta u(k+1|k) \ldots \Delta u(k+N_u-1|k)]^T \tag{2}$$

is calculated. It is assumed that $\Delta u(k+p|k) = 0$ ($u(k+p|k) = u(k+N_u-1|k)$) for $p \geq N_u$, where N_u is the control horizon. The objective is to minimise the differences between the reference trajectory $y^{ref}(k+p|k)$ and the predicted values of the output $\hat{y}(k+p|k)$ over the prediction horizon N. The following quadratic cost function is usually used

$$J(k) = \sum_{p=1}^{N}(y^{ref}(k+p|k) - \hat{y}(k+p|k))^2 + \sum_{p=0}^{N_u-1}\lambda_p(\Delta u(k+p|k))^2 \tag{3}$$

where $\lambda_p > 0$ are weighting factors. Typically, $N_u < N$, which decreases the dimensionality of the optimisation problem and leads to smaller computational load. Only the first element of the determined sequence (1) or (2) is applied to the process, the control law is then

$$u(k) = u(k|k) \qquad \text{or} \qquad u(k) = \Delta u(k|k) + u(k-1) \tag{4}$$

At the next sampling instant, $k+1$, the prediction is shifted one step forward and the whole procedure is repeated.

Since the constraints have to be usually taken into account, future control increments are determined as the solution to the following optimisation problem (assuming hard output constraints [8], [14] for simplicity of presentation)

$$\begin{aligned}
&\min_{u(k|k)\ldots u(k+N_u-1|k)} \{J(k)\} \\
&\text{subject to} \\
&\quad u^{\min} \leq u(k+p|k) \leq u^{\max}, \quad p = 0, \ldots, N_u - 1 \\
&\quad -\Delta u^{\max} \leq \Delta u(k+p|k) \leq \Delta u^{\max}, \quad p = 0, \ldots, N_u - 1 \\
&\quad y^{\min} \leq \hat{y}(k+p|k) \leq y^{\max}, \quad p = 1, \ldots, N
\end{aligned} \tag{5}$$

Predicted values of the output, $\hat{y}(k+p|k)$, over the prediction horizon are calculated using a dynamic model of the process. All MPC algorithms discussed in this paper use feedforward neural network models with one hidden layer and a linear output [1]. As an alternative, RBF type neural networks can be used in MPC algorithms [6].

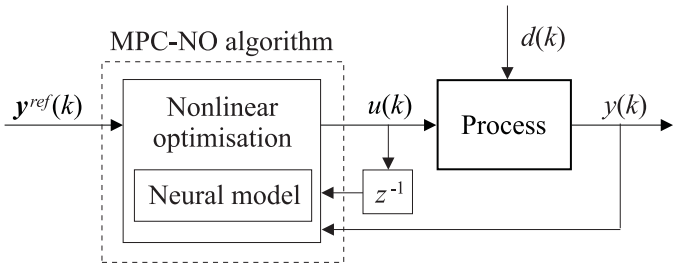


Fig. 1. Structure of the MPC algorithm with Nonlinear Optimisation (MPC-NO)

3 MPC Algorithm with Nonlinear Optimisation (MPC-NO) and Neural Models

In the MPC-NO algorithm for prediction purposes the nonlinear neural model is used without any simplifications [5, 14, 15]. At each sampling instant future values of control signals, $\boldsymbol{u}(k)$, are determined as the solution to a nonlinear optimisation problem. Structure of the MPC-NO algorithm is depicted in Fig. 1. From (3) and (5) the MPC-NO optimisation problem is

$$\begin{aligned} &\min_{\boldsymbol{u}(k)} \left\{ J(k) = \left\| \boldsymbol{y}^{ref}(k) - \hat{\boldsymbol{y}}(k) \right\|^2 + \left\| \Delta \boldsymbol{u}(k) \right\|^2_{\boldsymbol{\Lambda}} \right\} \\ &\text{subject to} \\ &\quad \boldsymbol{u}^{\min} \leq \boldsymbol{u}(k) \leq \boldsymbol{u}^{\max} \\ &\quad -\Delta \boldsymbol{u}^{\max} \leq \Delta \boldsymbol{u}(k) \leq \Delta \boldsymbol{u}^{\max} \\ &\quad \boldsymbol{y}^{\min} \leq \hat{\boldsymbol{y}}(k) \leq \boldsymbol{y}^{\max} \end{aligned} \tag{6}$$

where

$$\boldsymbol{y}^{ref}(k) = \left[y^{ref}(k+1|k) \dots y^{ref}(k+N|k) \right]^T \tag{7}$$

$$\hat{\boldsymbol{y}}(k) = \left[\hat{y}(k+1|k) \dots \hat{y}(k+N|k) \right]^T \tag{8}$$

$$\boldsymbol{y}^{\min}(k) = \left[y^{\min} \dots y^{\min} \right]^T \tag{9}$$

$$\boldsymbol{y}^{\max}(k) = \left[y^{\max} \dots y^{\max} \right]^T \tag{10}$$

are vectors of length N,

$$\boldsymbol{u}^{\min}(k) = \left[u^{\min} \dots u^{\min} \right]^T \tag{11}$$

$$\boldsymbol{u}^{\max}(k) = \left[u^{\max} \dots u^{\max} \right]^T \tag{12}$$

$$\Delta \boldsymbol{u}^{\max}(k) = \left[\Delta u^{\max} \dots \Delta u^{\max} \right]^T \tag{13}$$

are vectors of length N_u and $\boldsymbol{\Lambda} = diag(\lambda_0, \dots, \lambda_{N_u - 1})$. In the simplest and the most common case the reference trajectory is not known in advance, hence $\boldsymbol{y}^{ref}(k) = \left[y^{ref}(k) \dots y^{ref}(k) \right]^T$ where $y^{ref}(k)$ is the current set-point.

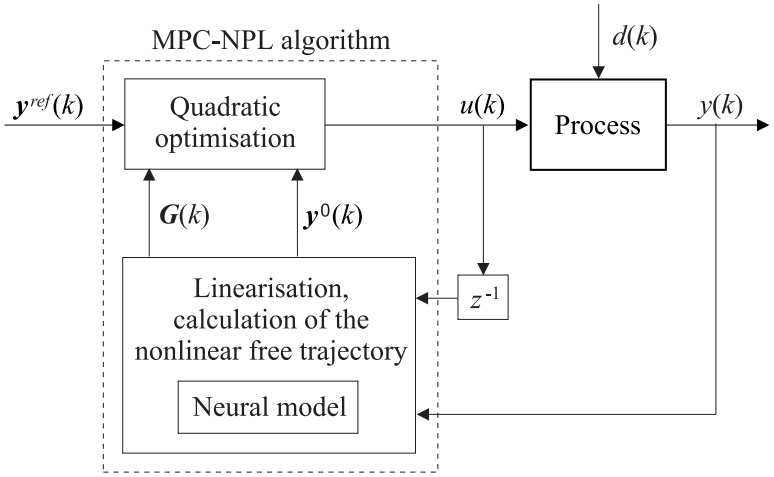


Fig. 2. Structure of the MPC algorithm with Nonlinear Prediction and Linearisation (MPC-NPL)

Sequential Quadratic Programming (SQP) is used for solving the nonlinear MPC-NO optimisation problem (6). Structure of the neural model is exploited, gradients of the cost function $J(k)$ with respect to future controls, $\boldsymbol{u}(k)$, are calculated analytically. It is necessary to emphasise the fact that the difficulty of the MPC-NO optimisation problem (6) is twofold. First of all, it is nonlinear, computationally demanding, the computational burden is big. Secondly, it may be non-convex and even multi-modal. Unfortunately, for such problems there are no sufficiently fast and reliable optimisation algorithms, i.e. those which would be able to determine the global optimal solution at each sampling instant and within predefined time limit as it is required in on-line control.

4 MPC Algorithm with Nonlinear Prediction and Linearisation (MPC-NPL) and Neural Models

In the MPC-NPL algorithm at each sampling instant k the neural model is used on-line twice: to determine a local linearisation and a nonlinear free trajectory [5], [6], [7], [14], [15]. It is assumed that the output prediction can be expressed as the sum of the forced trajectory, which depends only on the future (on future input moves $\Delta\boldsymbol{u}(k)$) and the free trajectory $\boldsymbol{y}^0(k)$, which depends only on the past

$$\hat{\boldsymbol{y}}(k) = \boldsymbol{y}^0(k) + \boldsymbol{G}(k)\Delta\boldsymbol{u}(k) \tag{14}$$

where $\boldsymbol{G}(k)$ is a dynamic matrix of the linearised model and

$$\boldsymbol{y}^0(k) = \left[y^0(k+1|k) \dots y^0(k+N|k)\right]^T \tag{15}$$

Both the free trajectory and the dynamic matrix are calculated on-line from the nonlinear model taking into account the current state of the plant.

On the one hand, the suboptimal prediction calculated from (14) is different from the optimal one determined from the nonlinear neural model as it is done in the MPC-NO algorithm. On the other hand, thanks to using the superposition principle, the optimisation problem (6) becomes the following quadratic programming task

$$\begin{aligned} &\min_{\Delta \boldsymbol{u}(k)} \left\{ J(k) = \left\| \boldsymbol{y}^{ref}(k) - \boldsymbol{y}^0(k) - \boldsymbol{G}(k)\Delta \boldsymbol{u}(k) \right\|^2 + \left\| \Delta \boldsymbol{u}(k) \right\|^2_{\boldsymbol{\Lambda}} \right\} \\ &\text{subject to} \\ &\quad \boldsymbol{u}^{\min} \leq \boldsymbol{u}(k) \leq \boldsymbol{u}^{\max} \\ &\quad -\Delta \boldsymbol{u}^{\max} \leq \Delta \boldsymbol{u}(k) \leq \Delta \boldsymbol{u}^{\max} \\ &\quad \boldsymbol{y}^{\min} \leq \hat{\boldsymbol{y}}(k) \leq \boldsymbol{y}^{\max} \end{aligned} \tag{16}$$

In the MPC-NPL algorithm it is convenient to calculated future control increments $\Delta \boldsymbol{u}(k)$ rather than absolute values $\Delta \boldsymbol{u}(k)$ of future control signal as it is done in the MPC-NO algorithm.

Structure of the MPC-NPL algorithm is depicted in Fig. 2. At each sampling instant k the following steps are repeated:

1. Linearisation of the neural model: obtain the matrix $\boldsymbol{G}(k)$.
2. Calculate the nonlinear free trajectory $\boldsymbol{y}^0(k)$ using the neural model.
3. Solve the quadratic programming problem (16) to determine $\Delta \boldsymbol{u}(k)$.
4. Apply $u(k) = \Delta u(k|k) + u(k-1)$.
5. Set $k := k+1$, go to step 1.

5 Simulation Results

5.1 Process, Models and Compared MPC Strategies

The process under consideration is a polymerisation reaction taking place in a jacketed continuous stirred tank reactor [9]. The reaction is the free-radical polymerisation of methyl methacrylate with azo-bis-isobutyronitrile as initiator and toluene as solvent. The output $NAMW$ (Number Average Molecular Weight) is controlled by manipulating the inlet initiator flow rate F_I.

Three models of the process are used. The fundamental model [9] is used as the real process during simulations. An identification procedure is carried out, a linear model and a neural one are obtained. Empirical models used in MPC algorithms are:

a) a linear model

$$y(k) = b_2 u(k-2) - a_1 y(k-1) - a_2 y(k-2) \tag{17}$$

b) a neural model containing 6 neurons in the hidden layer

$$y(k) = g(u(k-2), y(k-1), y(k-2)) \tag{18}$$

where $u = F_I$, $y = NAMW$. Compared MPC strategies are:

a) the linear MPC algorithm with the linear model (17),
b) the nonlinear MPC-NO algorithm with the neural model (18),
c) the suboptimal nonlinear MPC-NPL algorithm with the neural model (18),

Because one may expect that the initial point for the MPC-NO optimisation problem, $\boldsymbol{u}^0(k)$, is likely to affect the computational burden, four subversions of the MPC-NO algorithm are used:

a) MPC-NOa: $\boldsymbol{u}^0(k)$ uses $N_u - 1$ control values calculated at the previous sampling instant and not applied to the process

$$\boldsymbol{u}^0(k) = \begin{bmatrix} u^0(k|k) \\ \vdots \\ u^0(k+N_u-3|k) \\ u^0(k+N_u-2|k) \\ u^0(k+N_u-1|k) \end{bmatrix} = \begin{bmatrix} u(k|k-1) \\ \vdots \\ u(k+N_u-3|k-1) \\ u(k+N_u-2|k-1) \\ u(k+N_u-2|k-1) \end{bmatrix} \tag{19}$$

b) MPC-NOb: $\boldsymbol{u}^0(k)$ uses the value of the manipulated variable calculated and applied to the plant at the previous sampling instant, i.e $\boldsymbol{u}^0(k) = [u(k-1) \dots u(k-1)]^T$,
c) MPC-NOc: $\boldsymbol{u}^0(k)$ is found from the steady-state characteristic of the process for the current set-point $NAMW^{ref}(k)$,
d) MPC-NOd: $\boldsymbol{u}^0(k) = \text{const}$.

If $N_u = 1$, in the MPC-NOa algorithm it is assumed that $\boldsymbol{u}^0(k) = u(k-1)$, as it is done in the MPC-NOb one. The manipulated variable is constrained, $F_I^{\min} = 0.003$, $F_I^{\max} = 0.06$, $\lambda_p = 0.2$, the sampling time is 1.8 min.

5.2 Control Accuracy

Simulation results of the MPC algorithm with the linear model are depicted in Fig. 3. As the reference trajectory, four set-point changes are considered, the horizons are: $N = 10$, $Nu = 3$. The linear algorithm works well only for the smallest set-point change, whereas for bigger ones the system becomes unstable. Simulation results of the MPC-NPL and the MPC-NO (MPC-NOa) algorithms with the same neural network model are depicted in Fig. 4. Both nonlinear algorithms are stable. Moreover, the closed-loop performance obtained in the suboptimal MPC-NPL algorithm with quadratic programming is very close to that obtained in the computationally demanding MPC-NO approach, in which a nonlinear optimisation problem has to be solved on-line at each sampling instant.

5.3 Computational Complexity

Table 1 shows the influence of control and prediction horizons on floating point operation (MFLOPS). Six control horizon ($N_u = 1, 2, 3, 4, 5, 10$) and two prediction horizon ($N = 5, 10$) lengths are considered. Fig. 5 depicts this influence for

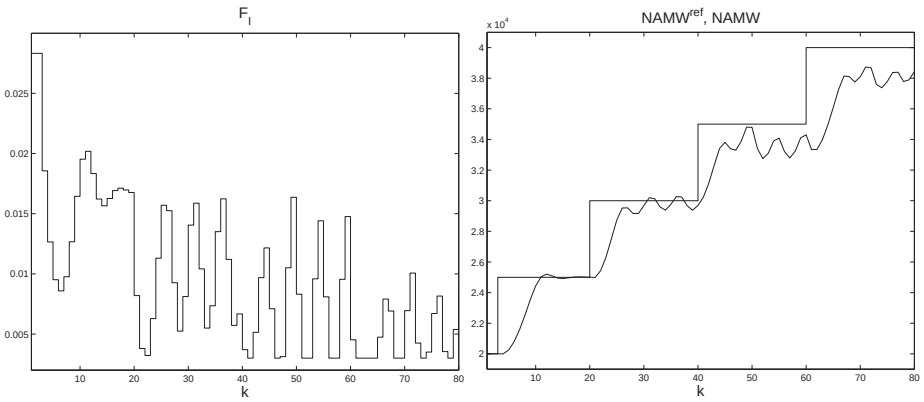


Fig. 3. Simulation results of the MPC algorithm with the linear model

all possible combinations of horizons ($N_u = 1, \ldots, 10$, $N = 2, \ldots, 10$). In general, the control horizon has significantly bigger impact on computational burden than the prediction horizon since N_u is the number of the decision variables of the optimisation problem (6) or (16).

It is best when the MPC-NO algorithm uses as the initial point for the optimisation problem $N_u - 1$ control values calculated at the previous sampling instant and not applied to the process (the MPC-NOa algorithm), which is clearly visible when the control horizon is long. Quite good results (for short control horizons comparable to those obtained in the MPC-NOa algorithm) are obtained when as the initial point the value of the manipulated variable calculated and applied to the plant at the previous sampling instant is used (the MPC-NOb algorithm). When the initial point is found from the steady-state characteristic of the process for the current set-point (the MPC-NOc algorithm) or a constant value is used (the MPC-NOd algorithm), the computational burden is very big.

Table 2 shows floating point operation reduction obtained in the MPC-NPL algorithm compared to the MPC-NO algorithm with different initial points for all possible combinations of horizons. In the worst case the MPC-NPL algorithm is more than 3 times less computationally demanding than the MPC-NO one, in the best case more than 25 times.

6 Conclusions

Accuracy and computational efficiency are the advantages of the suboptimal MPC-NPL algorithm with neural models. In practice [5–7, 14, 15], the algorithm gives closed-loop control performance comparable to that obtained in the MPC-NO schemes with nonlinear optimisation. Computational efficiency is twofold. First of all, the MPC-NPL algorithm uses on-line only the numerically reliable quadratic programming procedure, unlike the nonlinear optimisation, which may

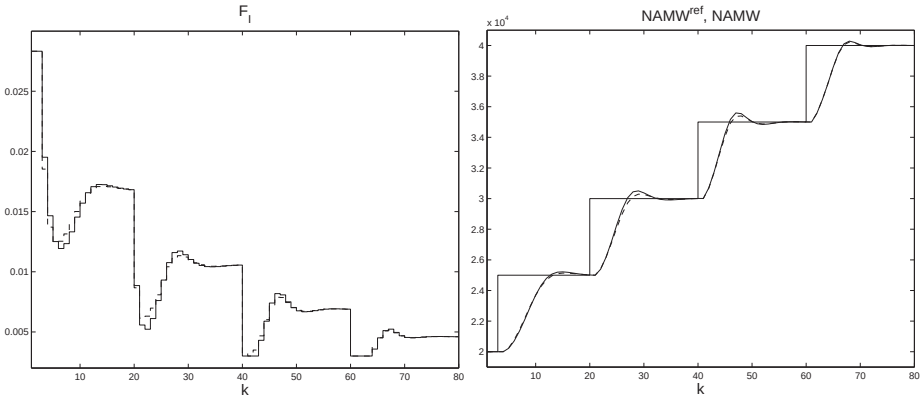


Fig. 4. Simulation results of the MPC-NPL (dotted) and MPC-NO (solid) algorithms with the same neural network model

Table 1. Influence of control and prediction horizons on floating point operation (MFLOPS) in the MPC-NPL algorithm and the MPC-NO algorithm with different initial points

Algorithm	N	$N_u = 1$	$N_u = 2$	$N_u = 3$	$N_u = 4$	$N_u = 5$	$N_u = 10$
MPC-NPL	5	0.0998	0.1221	0.1739	0.2469	0.3515	–
MPC-NOa	5	0.6522	1.0488	1.4110	2.1493	2.9892	–
MPC-NOb	5	0.6522	1.0206	1.6195	2.5040	3.6211	–
MPC-NOc	5	1.0558	1.7913	2.5847	3.9007	5.1672	–
MPC-NOd	5	1.0558	1.7913	2.5843	3.9045	5.1707	–
MPC-NPL	10	0.1607	0.1861	0.2427	0.3209	0.4315	1.4956
MPC-NOa	10	1.4705	2.4544	3.3510	4.4030	5.3262	12.5403
MPC-NOb	10	1.4705	2.6686	3.8834	5.2310	6.3347	17.4621
MPC-NOc	10	2.7457	4.0431	5.6171	7.7429	11.1757	28.2162
MPC-NOd	10	2.7504	4.0548	5.6054	7.7607	11.2266	28.2670

Table 2. Floating point operation reduction obtained in the MPC-NPL algorithm compared to the MPC-NO algorithm with different initial points for all possible combinations of horizons ($N_u = 1, \ldots, 10$, $N = 2, \ldots, 10$)

Algorithm	minimal	maximal
MPC-NPL vs. MPC-NOa	3.4317	13.6146
MPC-NPL vs. MPC-NOb	3.4340	16.1267
MPC-NPL vs. MPC-NOc	4.5388	25.6879
MPC-NPL vs. MPC-NOd	4.5276	25.8049

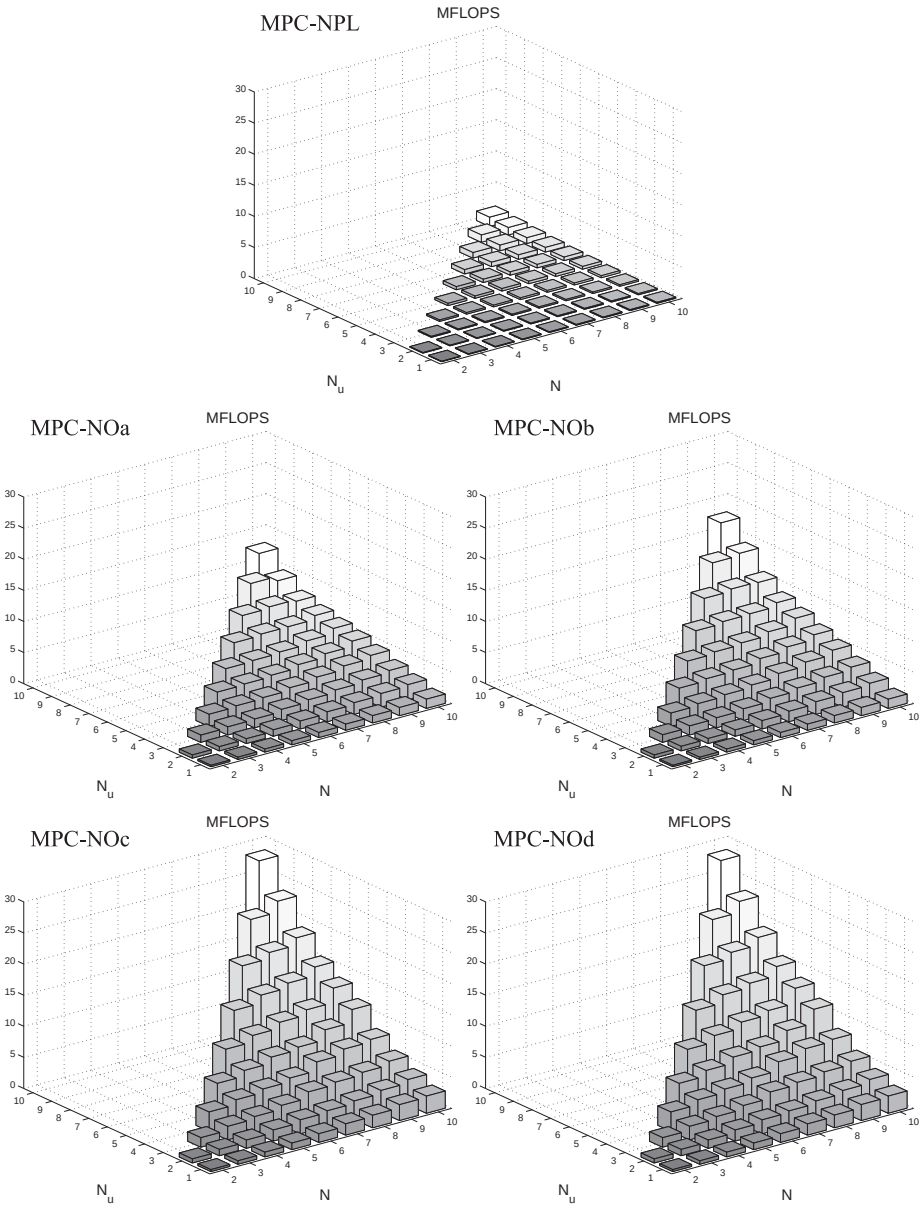


Fig. 5. Influence of prediction and control horizons on floating point operation (MFLOPS) in the MPC-NPL algorithm and the MPC-NO algorithm with different initial points

terminate in a local minimum, the global solution is always found. Moreover, for the studied polymerisation process, floating point operation reduction obtained in the suboptimal algorithm compared to the MPC-NO algorithm in the best case is bigger than 25 (Table 2).

Acknowledgement

This work was supported by Polish national budget funds 2005-2007 for science as a research project.

References

1. Haykin S.: *Neural networks—a comprehensive foundation. Prentice Hall,* Englewood Cliffs. (1999).
2. Henson M. A.: *Nonlinear model predictive control: current status and future directions,* Computers and Chemical Engineering. **23** (1998) 187–202.
3. Hussain M. A.: *Review of the applications of neural networks in chemical process control—simulation and online implmementation,* Artificial Intelligence in Engineering. **13** (1999) 55–68.
4. Liu G. P., Kadirkamanathan V., Billings S. A.: *Predictive control for non-linear systems using neural networks,* International Journal of Control. **71** (1998) 1119–1132.
5. Ławryńczuk M.: *A family of model predictive control algorithms with artificial neural networks,* International Journal of Applied Mathematics and Computer Science. **17** (2007) 217–232.
6. Ławryńczuk, M. and Tatjewski, P.: *A computationally efficient nonlinear predictive control algorithm with RBF neural models and its application,* Lecture Notes in Artificial Intelligence, Springer. **4585** (2007) 603–612.
7. Ławryńczuk, M., Tatjewski, P.: *An efficient nonlinear predictive control algorithm with neural models and its application to a high-purity distillation process,* Lecture Notes in Artificial Intelligence, Springer. **4029** (2006) 76–85.
8. Maciejowski J. M.: *Predictive control with constraints,* Prentice Hall. Harlow. (2002).
9. Maner B. R., Doyle F. J., Ogunnaike B. A., Pearson R. K.: *Nonlinear model predictive control of a simulated multivariable polymerization reactor using second-order Volterra models,* Automatica. **32** (1996) 1285–1301.
10. Nørgaard M., Ravn O., Poulsen N. K., Hansen L. K.: *Neural networks for modelling and control of dynamic systems,* Springer. London. (2000).
11. Parisini T., Sanguineti M., Zoppoli R.: *Nonlinear stabilization by receding-horizon neural regulators,* International Journal of Control. **70** (1998) 341–362.
12. Piche S., Sayyar-Rodsari B., Johnson D., Gerules M.: *Nonlinear model predictive control using neural networks,* IEEE Control Systems Magazine. **20** (2000) 56–62.
13. Qin S. J., Badgwell T. A.: *A survey of industrial model predictive control technology,* Control Engineering Practice. **11** (2003) 733–764.
14. Tatjewski P.: *Advanced control of industrial processes, Structures and algorithms,* Springer. London. (2007).
15. Tatjewski P., Ławryńczuk M.: *Soft computing in model-based predictive control,* International Journal of Applied Mathematics and Computer Science. **16** (2006) 101–120.

Proceedings of the International Multiconference on Computer Science and Information Technology, pp. 267 – 274
ISSN 1896-7094
© 2007 PIPS

Improved TBL algorithm for learning context-free grammar

Marcin Jaworski [1] and Olgierd Unold [2]
[1] The Institute of Computer Engineering, Control and Robotics, Wroclaw University of Technology, jaworski.marcin.e@gmail.com
[2] The Institute of Computer Engineering, Control and Robotics, Wroclaw University of Technology, olgierd.unold@pwr.wroc.pl

Abstract. In this paper we introduce some improvements to the *tabular representation algorithm* (TBL) dedicated to inference of grammars in Chomsky normal form. TBL algorithm uses a Genetic Algorithm (GA) to solve partitioning problem, and improvements described here focus on this element of TBL. Improvements involve: i nitial population block size manipulation, block delete specialized operator and modified fitness function. The improved TBL algorithm was experimentally proved to be not so much vulnerable to block size and population size, and is able to find the solutions faster.

1 Introduction

Learning structural models from data is known as grammatical inference (GI) [1]. The data can represent sequences of natural language corpora, biosequences (DNA, RNA, primary structure of proteins), speech etc., but can also include trees, metabolic networks, social networks or automata. Typical models include formal grammars, and statistical models in related formalisms such as probabilistic automata, Hidden Markov Models, probabilistic transducers or conditional random fields.

GI is the gradual construction of a model based on a finite set of sample expressions. In general, the training set may contain both positive and negative examples from the language under study. If only positive examples are available, no language class other than the finite cardinality languages is learnable [1]. It has been proved that deterministic finite automata are the largest class that can be efficiently learned by provable converging algorithms. There is no context-free grammatical inference theory which provable converges, if language defined by a grammar is infinite [8]. Building algorithms that learn context–free grammars (CFG) is one of the open and crucial problems in the grammatical inference [2].

The approaches taken have been to provide learning algorithms with more helpful information, such as negative examples or structural information; to formulate alternative representation of CFGs; to restrict attention to subclasses of context-free languages that do not contain all finite languages; and to use Bayesian methods [4]. Many researchers have attacked the problem of grammar induction by using evolutionary methods to evolve (stochastic) CFG or equivalent pushdown automata (for references see [9]), but mostly for artificial languages like brackets, and palindromes. For surveys of the non-evolutionary approaches for CFG induction see [4].

In this paper we introduce some improvements to the TBL dedicated to inference of CFG in Chomsky normal form. The TBL (tabular representation algorithm) was proposed by Sakakibara and Kondo [6], in [7] Sakakibara analyzed some theoretical

foundations for the algorithm. The improved TBL algorithm is not so much vulnerable to block size and population size, and is able to find the solution faster.

2 Context-free grammar parsing

A context free grammar (CFG) is a quadruple $G=(\Sigma, V, R, S)$, where Σ is finite set of terminal symbols called alphabet and V is finite set of nonterminal symbols such that $\Sigma \cap V = \emptyset$. R is set of production rules in the form $W \rightarrow \beta$ where $W \in V$ and $\beta \in (V \cup \Sigma)^+$. $S \in V$ is a special nonterminal symbol called start symbol. All derivations start using the start symbol S. A derivation is sequence of rewriting the string containing symbols $(V \cup \Sigma)^+$ using production rules of CFG G. In each step of a derivation a nonterminal symbol from string is selected and replaced by string on the right side of the production rule. For example if we have CFG $G=(\Sigma, V, R, S)$ where Σ = { a, b }, V = { A, B }, R = { A→ a, A → BA,A → AB, A→ AA, B→ b }, S = A, then the derivation can have the form:

A→ AB → ABB →AABB→ aABB → aABb → aAbb → aabb

The derivation ends when there are no more nonterminal symbols left in the string. The language generated by CFG is denoted $L(G)=\{x \in \Sigma^* | S \Rightarrow_G^* x\}$, where x represents words of language L. The word x is a string of any number of terminal symbols derived using production rules fromCFG G. A CFG $G=(\Sigma, V, R, S)$ is in Chomsky normal form if every production rule in R has form:

A → BC or A → a, where $A, B, C \in V$ and $a \in \Sigma$.

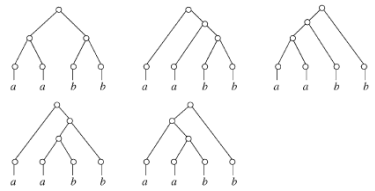
A CFG describes specific language L(G) and can be used to recognize words (strings) that are members of language L(G). But to answer the question which words are members of language L(G) and which are not we need a parsing algorithm. This problem, called membership problem can be solved using CYK algorithm [3]. This algorithm is a bottom up dynamic programming technique that solves the membership problem in a polynomial time.

3 CFG induction using TBL algorithm

Grammar induction target is to find a CFG structure (set or rules and set of nonterminals) using positive and negative examples in a form of words. It is a very complex problem, especially hard is finding grammar topology (set of rules). Its complexness results from great amount of possible solutions that grows exponentially with length of the examples.

TBL algorithm uses tabular representations of positive examples. Tabular representation is similar to parse table of CYK algorithm and is able to remember exponential number of possible grammatical structures of example in a table of polynomial size. Figure 1 shows all possible grammatical structures of word „aabb" and it's tabular representation.

a)



b)

j=4	<12><13><14>			
j=3	<8><9>	<10><11>		
j=2	<5>	<6>	<7>	
j=1	<1>	<2>	<3>	<4>
	i=1	i=2	i=3	i=4
	a	a	b	b

Fig. 1: All possible grammatical structures of word "aabb" (a) and its tabular representation (b)

More precisely said, having an example $w = a_{1,} a_{2,} \ldots, a_n$ its tabular representation is a tree dimensional table T(w) where each element $t_{i,j}$ ($1 \le i \le n$ and $2 \le j \le n - i + 1$) contains set $\{X_{i,j,k_1}, \ldots, X_{i,j,k_{j-1}}\}$ of j-1 nonterminals. For i=1 $t_{i,1}$ is a single element $\{X_{i,1,1}\}$.

Tabular representations are used to create primitive grammars $G(T(w)) = (N, \Sigma, R, S)$ which are defines as:

$$N = \{X_{i,j,k} \mid 1 \le i \le n, 2 \le j \le n-i+1, 1 \le k \le j\} \cup \{X_{i,1,1} \mid 1 \le i \le n\} \quad (1)$$

$$R = \{X_{i,j,k} \to X_{i,k,l1} X_{i+k,j-k,l2} \mid 1 \le i \le n, 2 \le j \le n-i+1, 1 \le k \le j, 1 \le l1 < k, 1 \le l2 < j-k\} \cup \{X_{i,1,1} \to a_i \mid 1 \le i \le n\} \cup \{S \to X_{1,n,k} \mid 1 \le k \le n-1\} \quad (2)$$

This pseudo code shows all steps of TBL algorithm.

1. Create tabular representation $T(w)$ of each positive example w in U_+.
2. Derive primitive CFG $G(T(w))$ i $G^w(T(w))$ for each w in U_+
3. Create union of all primitive CFG's, $G(T(U_+)) = \bigcup_{w \in U_+} G^w(T(w))$
4. Find smallest partition π_* such that $G(T(U_+))/\pi_*$ is coherent with U, which is compatible with all positive and negative examples U_+ and U_-
5. Return result CFG $G(T(U_+))/\pi_*$

Primitive grammars $G^w(T(w))$ are grammars that have nonterminals with unique names,and are created only because all nonterminals of all primitive grammars have to have distinct names (step 3). Thank to use of tabular representations in TBL algorithm, the grammar induction problem is reduced to partitioning problem (step 4). This problem is simpler but still it is NP-hard and because of this TBL uses a genetic algorithm (GA), to solve the partitioning problem. Use of GA in minimal partition problem is well studied in [5]

In experiments described in section 5, as partitioning algorithm was used a typical, one population (not parallel) GA with some modifications described in [7]. This algorithm uses more intelligent genetic operators: *structural crossover, structural mutation, and special deletion operator*.

Population individual representation. Individuals (partitions) are encoded using a single string of integers, for example:

p = (1 3 4 3 6 3 3 4 5 2 6 4).

In such string position of each integer encodes number of nonterminal and value encodes to which block it belongs. So string p represents partition shown below (for more information about partitions see [7]):

π_{p1} : {1}, {10}, {2 4 6 7}, {3 8 12}, {9}, {5 11}.

Structural crossover. Let's assume we have two individuals p_1 and p_2.

p_1=(1 1 2 3 4 1 2 3 2 2 5 3), encoded partition

π_{p1} : {1 2 6}, {3 7 9 10}, {4 8 12}, {5}, {11}.

p_2=(1 1 2 4 2 3 1 2 2 3 3 5). encoded partition

π_{p1} : {1 2 7}, {3 5 8 9}, {6 10 11}, {4}, {12}.

During structural crossover a random block is selected from one partition, say block 2 from p_1, and is merged with block from p_2. The result is:

p_2'=(1 1 2 4 2 3 2 2 2 2 3 5), encoded partition

$\pi_{p'_1}$: {1 2}, {3 5 7 8 9 10}, {6 11}, {4}, {12}.

Structural mutation. Structural mutation simply replaces one integer in string with another integer. Let's assume we have individual p_2' that should be mutated:

p_2'=(1 1 2 4 2 3 2 2 2 2 3 5),

After replacing single random integer, for example integer number 9 the result is:

p_2''=(1 1 2 4 2 3 2 2 5 2 3 5), encoded partition

$\pi_{p'_2}$: {1 2}, {3 5 7 8 10}, {6 11}, {4}, {9 12}.

Special deletion operator. This operator replaces single integer in string with a special value (-1). This value means that this nonterminal is no longer used. This operators is important because it reduces the number of nonterminals in result grammar.

Fitness function. Fitness function is a number from interval [0, 1]. The fitness function was designed to both maximize number of accepted positive examples f_1 and minimize number of nonterminals f_2. If grammar based on individual accepts any negative example it gets fitness 0 to be immediately discarded. C1 and C2 are some constants that can be selected experimentally.

$$f_1(p)=\frac{|\{w\in U_+ \mid G(T(U_+))/\pi_p\}|}{|U_+|}, \tag{3}$$

$$f_2(p)=\frac{1}{|\pi_p|}, \tag{4}$$

$$f(p)=\begin{cases} 0 & \text{if any negative example from } U_- \\ & \text{is generated by } G(T(U_+))/\pi_p \\ \dfrac{C_1\cdot f_1(p)+C_2\cdot f_2(p)}{C_1+C_2} & \text{otherwise,} \end{cases} \tag{5}$$

Although this algorithm is effective we have encountered many difficulties using it during experiments. The main problem was the fitness function, which was not always good enough to describe differences between individuals. When partitions with small block were used (average size 1-5) algorithm usually could find solution, but when partition with larger blocks were used, TBL algorithm often could not find solutions because all individuals in population had fitness 0 and started to work completely randomly. To solve this problem we propose some modifications described in section 4.

4 Improved TBL algorithm

The proposed improvements affect only the genetic algorithm solving the partitioning problem. First we want to discuss closely concept of partition block size in initial population. Partitions used by GA group integers that represent nonterminals in blocks. For example the partition π_{pI} contains of 6 blocks of size 1 to 4:

π_{pI} : {1}, {10}, {2 4 6 7}, {3 8 12}, {9}, {5 11}.

Range of partition block size in initial population have great influence at evolution process. If blocks are small (size 1-4) then it is easier to find solutions (partitions) π generating *CFG* $G(T(U_+))/\pi$ that do not parse any negative examples. On the other hand creating partitions that contain bigger blocks should assure that the reduction of nonterminal number in $G(T(U_+))/\pi$ (number of blocks in π) to the optimal level proceeds quicker (this is slower process than finding solution that do not accept any negative example), because partitions in initial generation contain less blocks.

The main problem with using bigger blocks is low percentage of successful runs if the standard fitness function is used. Usually in such case all individuals (partitions) create *CFG's* that accept at least one negative example and then whole population has the same fitness equal 0. To overcome those difficulties, we have used some modifications in *GA* listed below.

Initial population block size manipulation. During the experiments was tested the influence of block size on the evolution process. Block size is a range of possible sizes (for example 1-5), not one number.

Block delete specialized operator. This operator is similar to special deletion operator described earlier but instead of removing one nonterminal from partition it deletes all nonterminals that are in some randomly selected block.

Modified fitness function. In order to solve problems with using standard fitness function when block sizes are bigger (usually all individuals have fitness 0) we propose modified fitness function that does not give all individuals that accept a negative example the same fitness value. To be able to distinguish better and worse solutions that still accept any negative example we give that solutions negative fitness value (values in range [-1, 0)). This fitness function range [-1, 1] is used only to be able to compare modified fitness function with standard fitness function more easily but it can be of course normalized to range [0,1].

$$f_1(p)=\frac{|\{w\in U_+ \mid G(T(U_+))/\pi_p\}|}{|U_+|}, \tag{6}$$

$$f_2(p)=\frac{1}{|\pi_p|}, \tag{7}$$

$$f_3(p)=\frac{|\{w\in U_- \mid G(T(U_+))/\pi_p\}|}{|U_-|}, \tag{8}$$

$$f(p)=\begin{cases} -(C_3\cdot f_3+C_4\cdot(1-f_1)) & \text{if any example from } U_- \text{ is} \\ & \text{generated by } G(T(U_+))/\pi_p \\ \dfrac{C_1\cdot f_1(p)+C_2\cdot f_2(p)}{C_1+C_2} & \text{otherwise,} \end{cases} \tag{9}$$

5 Experimental results

In experiments was used similar data to those used by Y. Sakakibara in [7], to have good comparison to results shown there. In both experiments (and for each block size range) induction was repeated until 10 successful runs. As a successful run (in this paper) we understand run, when algorithm finds grammar $G(T(U_+))/\pi_*$, that accepts all positive examples and none negative example within 1100 generations. *Successful runs* values in Table 1 and 2 are defined as $SR=\lfloor SRN/ORN \rfloor \cdot 100\lfloor\% \rfloor$, where SRN – successful run number, ORN – overall run number. Charts in Figure 2 and 3 were created using data only from successful runs.

Experiment 1. As positive examples were used words from language $L=\{a^n b^n c^m \mid n,m \geq 1\}$ up to length 6, and set of 15 negative examples length up to 6, the population size was 100 and $C_1=C_2=C_3=C_4=0{,}5$.

Table 1: Experiment 1 results

	Standard fitness function		Modified fitness function	
Block size	Successful runs	Aver. number of generations needed	Successful runs	Aver. number of generations needed
1-5	100%	950	100%	950
3-7	91%	800	100%	600

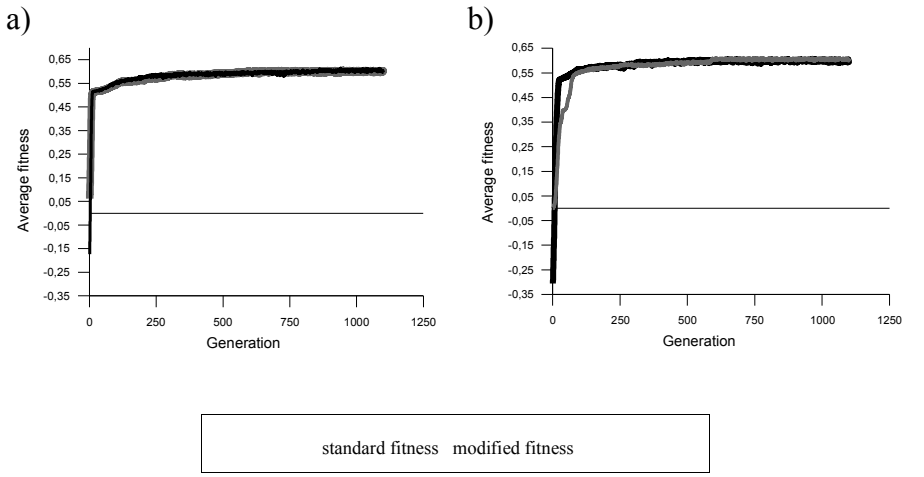


Figure 2: Learning over all examples from language $L=\{a^n b^n c^m \mid n,m \geq 1\}$ up to length 6. Initial population block size: 1-5 (a) and 3-7 (b).

Experiment 2. As positive examples were used all words from language $L=\{ac^n \cup bc^n \mid n \geq 1\}$ up to length 5, and set of 25 negative examples up to length 5, the population size was 20.

Table 2: Experiment 2 results

	Standard fitness function		Modified fitness function	
Block size	Successful runs	Aver. number of generations needed	Successful runs	Aver. number of generations needed
1-5	77%	1000	100%	1000
2-6	9,5%	900	100%	900

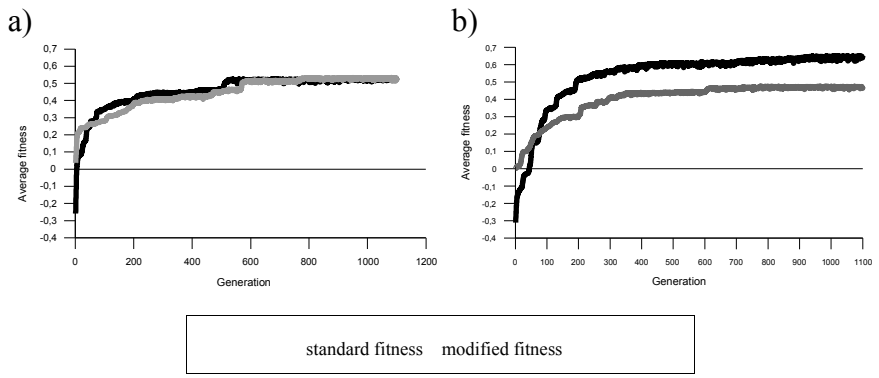


Figure 3: Learning over all examples from language $L=\{ac^n \cup bc^n \mid n \geq 1\}$ up to length 5. Initial population block size was:1-5 (a) and 2-6 (b).

6 Conclusions

In this paper we have introduced some modifications to TBL algorithm which are: initial population block size manipulation, block delete specialized operator and modified fitness function. To test influence of this modifications on TBL performance and results quality we made several experiments. Results of those experiments are shown in section 5.

If we compare experiment results in table 1 and 2 we can see that size of block has big influence on TBL's ability to find solution within 1100 generations when standard fitness function is used. As shown in results of experiment 1, if block size in initial population is 1-5, then TBL algorithm finds solution within 1100 generations in 100% cases, but if block size is 3-7 it finds solution only in 91% cases. In experiment 2 the difference is even greater. If block size is 1-5 algorithm finds solution in 77% cases and if block size is 3-7, then it finds solution only in 9,5% cases. Block size has no influence on number of successful runs when modified fitness function is used and TBL always finds solution. It means that standard fitness is really effective only if we use very small blocks (size 5 or smaller) to build initial population. If we use modified fitness function instead, we can build initial population using much greater blocks.

Experiment results in section 5 show also that population size has big influence on number of successful runs when the standard fitness function is used. In both experiments size of the problem was similar but results (number of successful runs if

the standard fitness function was used) were much better in experiment 1, where population size was bigger (100) than in experiment 2 (20). Experiment results show also that population size has no bigger influence on effectiveness of algorithm if the modified fitness function is used.

The use of larger blocks has improves also quality of solutions and algorithm speed. Figure 3 shows more detailed results of experiment 2. If we use smaller blocks the results are similar (Figure 3a), but when blocks are bigger, the average fitness rises quicker when the modified fitness function is used (Figure 3b) and the solutions achieved after 1100 generations are better (greater fitness). The solutions are not only better in comparison with standard fitness using larger blocks (2-6), but also better than solutions achieved when smaller blocks were used (1-5).

Using larger block size reduces also average number of generations needed to find best solution. This is possible because each partition π_i in initial population contain smaller number of blocks and *CFG* $G(T(U_+))/\pi_i$ based on π_i contain less nonterminals. Algorithm finds solution faster because nonterminal number reduction process is quicker (smaller initial number of nonterminals).

References

1. Gold E.: *Language Identification in the Limit,* Information Control, 10: (1967) 447-474.
2. Higuera C.: *Current Trends in Grammatical Inference,* In: Ferri F. J. et al. (eds) Advances in Pattern Recognition. Joint IAPR International Workshops SSPR+SPR'2000, LNCS 1876, Springer, Berlin, (2000) 28-31.
3. Kasami T.: *An efficient recognition and syntax-analysis algorithm for context-free languages,* Sci. Rep. AFCRL-65-558, Air Force Cambridge Research Laboratory, Bedford, Mass. (1965).
4. Lee L.: *Learning of Context-Free Languages: A Survey of the Literature,* Report TR-12-96, Harvard University, Cambridge, Massachusetts (1996).
5. Michalewicz Z.: *Genetic Algorithms + Data Structures = Evolution Programs,* Springer, Berlin (1996).
6. Sakakibara Y., Kondo M.: *GA-based learning of context-free grammars using tabular representations,* In: Proceedings of 16th International Conference in Machine Learning (ICML-99), Morgan-Kaufmann, Los Altos, CA, (1999) 354-360.
7. Sakakibara Y.: *Learning context-free grammars using tabular representations,* Pattern Recognition 38: (2005) 1372-1383.
8. Tanaka E.: *Theoretical Aspects of Syntactic Pattern Recognition,* Pattern Recognition, 28(7): (1995) 1053-1061.
9 . Unold O.: *Context-free grammar induction with grammar-based classifier system,* Archives of Control Science, 15 (LI) 4: (2005) 681-690.

Proceedings of the International Multiconference on Computer Science and Information Technology pp. 275–284
ISSN 1896-7094
© 2007 PIPS

Pattern Extraction for Event Recognition in the Reports of Polish Stockholders

Michał Marcińczuk and Maciej Piasecki

Institute of Applied Informatics, Wrocław University of Technology,
Wybrzeże Wyspiańskiego 27, Wrocław, Poland, maciej.piasecki@pwr.wroc.pl

Abstract. In the paper the application of the general Memory Base Learning to Event Recognition in the domain of reports of stock issuers is investigated. A multi-classifier scheme is applied in which the boundaries of annotations are identified first and then a heuristic algorithm of merging into pair is applied. A modified method based only on positive examples is proposed. Several types of simple features requesting only simple processing of text are tested. The proposed method can be trained on a small annotated corpus.

1 Introduction

Event Recognition (ER) is a specific case of Named Entity Recognition and relies on detecting given type of events and their attributes. ER has been used to retrieve informations about terrorist activities from news (MUC-3, MUC-4), air plane crashes and rocket/missile launches (MUC-7) and medical data from mammograhical reports [6].

The approaches to ER can be divided into three groups: manual approaches, automatic approaches and combinations of these two. The manual approach makes use of grammar [6], regular expressions [13] and predicate-argument models [14]. In the automatic approach the machine learning methods (decision trees [2]) and statistical models (Hidden Markov Model [15]) have been applied.

The manual approach allows to achieve very good quality (in terms of precision and recall) in comparison to the automatic approach. However, it has some disadvantages, like high cost in terms of time and human work to create the extraction patterns for a given type of annotation.

2 Event Recognition

ER is one of the Information Extraction tasks. It relies on detecting descriptions of events in text and extracting events attributes from text. An event is a fact that occurred in a time [1], for instance, the annual meeting of the stockholders announcement. Each type of event is characterized by set of attributes. For instance, the annual meeting of the stockholders announcement contains following information: the date, the hour and the place of the meeting, the meeting agenda, the date, the hour and the place of the trust deposit.

ER is a specific case of Named Entity Recognition [10, 5]. The core of Named Entity Recognition is finding descriptions of entities of the defined types in text. The task of ER is finding instances but also ER techniques take also the role of the instance into account. For example, the annual meeting of stockholders can be associated with two instances of the date, time and place but the instances have different roles. One refers to the meeting and the others refer to the trust deposit.

The goal of ER is to detect an event description in a text document and extract the values of the event attributes. Next, the found beginnings and endings of description are marked (annotated) with some *tags*.

The goal of this work is to construct a method of recognition of selected events in the domain of activities of Polish stock issuers. The method should be possible to be applied on the basis of a limited corpus annotated manually.

2.1 Event Recognition as Classification Task

In the naive approach ER can be treated as a single-classification task that it is performed by testing every subsequence of tokens, where a *token* is a basic segment of text e.g. a word form, number, date or some symbol in a document. However, in this approach some problems occur. The main problem is the number of instances to be tested. This is caused by variable lengths of annotations that are depending on the complexity of event descriptions. All different subsequences should be checked, i.e. the huge number of subsequences can be generated even for a sentence consisting of 20 tokens. The next problem is how to represent the problem by the means of a constant number of learning / testing features as most of the classification algorithms assume.

Fortunately, ER can be also considered as a multi-classification task. The idea is to decompose ER into several classification tasks and then merge the partial results into whole annotations. Bennett et al. [2] used two classifiers to recognize the beginnings and the endings of annotations. Pradhan et al. [11] used *IOB representation* (abbreviation stands for *Inside Outside Begin*) and created three classifiers that divide tokens into three classes: tokens beginning some annotation, being part of some annotation or being outside of any annotation.

Both approaches has the same problem that the non-boundaries tokens are strongly prevalent. This means that for every single positive instance there are hundreds of even thousands negative instances.

2.2 Instance Features

Descriptions of events are natural language expressions. The beginning and ending of expression can be characterised by several linguistic features that are used during training and classification. The groups of features presented below are put in order of increasing demands on the complexity of processing:

- *Rough characteristics of token characters*—type of characters occurring in token structure. There are 7 types of tokens distinguished according to types of characters constituting them. The types are presented in Table 1.

Table 1. Groups of tokens according to the characteristics of characters

Group	Description
LowerCase	a sequence of lower case letters
UpperCase	a sequence of upper case letters
InitCap	a capitalized word
MixedCase	a sequence of lower and upper case letters
Number	a sequence of digits
Symbol	a sequence of characters other than letters and digits
Mixed	a sequence of letters, digits and/or symbols

- *Morpho-syntactic description* of a word form (called also a morpho-syntactic tag)—we assume the IPI PAN Corpus (IPIC) format [12], according to which a description consists of: *a grammatical class* (more fine grained division of Parts of Speech into 32 classes), and value of grammatical categories, e.g. number, case, gender, person, degree and other. There are 12 grammatical categories assign in different subsets to different grammatical classes, see [12]. The morphological base form which is a part of an IPIC tag, is separated here because of its importance in the algorithms proposed here.
- *Morphological base form*—represents a set of word forms differing in values of grammatical categories but described as one lexeme in a lexicon. As assignment of morpho-syntactic tags to word forms is often ambiguous (many tags possible for a word form) for these two features the usage of tagger disambiguating description is necessary.
- *Shallow/deep syntactic structure*—is necessary in the construction of the *predicate-argument model* of a sentence that is the basis for the assignment of semantic roles to the sentence elements in [14].
- *Semantic groups*—defined on the basis of automatically extracted semantic similarity (semantic relatedness), or defined manually in the form of sets of synonyms, e.g. [7], could be used to divide word forms into semantic classes, that would simplify the classifier training and work. However, any such semantic resource has not been constructed for the domain of our interest, yet.

3 Memory Based Learning

Most of the Machine Learning algorithms needs a large set of training examples in order to achieve good results. In our domain of interest, the preparation of such set is very laborious as it means that hundreds of annotations must be manually introduced to text documents. Thus we assumed the general scheme of the *Memory Based Learning* as the basis for our solution. Decreased demands on the preparation of training data makes the proposed solution closer to practical applications.

In our approach, two separate classifiers are built: one for annotation beginnings, and the second one for endings. The identified beginnings and endings are next combined by a heuristic algorithm, see Sec. 3.2.

Both classifier use three features types (token type, base form and grammatical class) of three tokens before a position being classified (an *outer context*) and two tokens after (an *inner context*).

Base forms and grammatical classes are identified by the previous application of the TaKIPI tagger [9]. As two additional features, we introduced the first preceding noun and preposition. A sample training instance of date annotation is shown in Table 2.

Table 2. A sample training instance of date annotation starting boundary

feature no.		w	dniu	19	czerwca	2006
1-5	type	LowerCase	LowerCase	Number	LowerCase	Number
6-10	base form	w	dzień	19	czerwiec	1006
11-15	grammatical class	prep	subst	ign	subst	ign
16	preceding noun base	odbyć				
17	preceding preposition	w				

3.1 Original Approach

Memory Based Learning [3] (also known as *Instance Based Learning* or *Lazy Learning* [8]) relies on storing the training instances as they are. No generalization or reduction is performed at the training step so no information is lost. The problem of generalization is postponed until new instances are classified.

As all the features used in our experiments are symbolic, we have chosen the overlap metric (Equation 1 and 2).

$$\Delta(X, Y) = \sum_{i=1}^{n} \delta(x_i, y_i) \tag{1}$$

$$\delta(x_i, y_i) = \begin{cases} 1 & \text{if } x_i = y_i \\ 0 & otherwise \end{cases} \tag{2}$$

Memory Base Learning applies the k-NN (k-Nearest Neighbours) method to classify new objects during working phase. The method assign this class to a new object that is most common among a group of the k most similar stored, training examples. A visualization of this method is presented on the Figure 1 on the left side., where triangle represent our tested object (i.e. a vector of feature values describing a given position in text), the circles negative training examples and the discs positive examples. The gray disc encapsulate the k=5 neighbors, which are nearest to the tested object. In this case the new object will be classified as a positive one.

3.2 Modified MBL

Because of the fact that negative instances (non-boundary tokens) are strongly prevalent we introduced a modification of the original Memory Based Learning methods that can cope with this problem. In the modified MBL only positive examples, which represent starting or ending boundaries, are stored. The negative examples (non-boundary tokens) are not taken into consideration at all. However, in that way for almost any position in text we can expect to have some positive neighbours in a distance greater than zero.

Because only examples of one class are stored, the k-NN methods cannot be used. Instead, we introduced a *threshold* for the number of neigbours written as the parameter k. The classification process starts with measuring the distance between a new object and all stored examples. Next, the number of examples whose similarity is greater or equal to the *value threshold* t is counted—only these are treated as neighbours in the next step. A new object is classified as positive only if the number of found neighbours is greater or equal to the parameter k. A visualization of this method is presented in Figure 1 (right side).

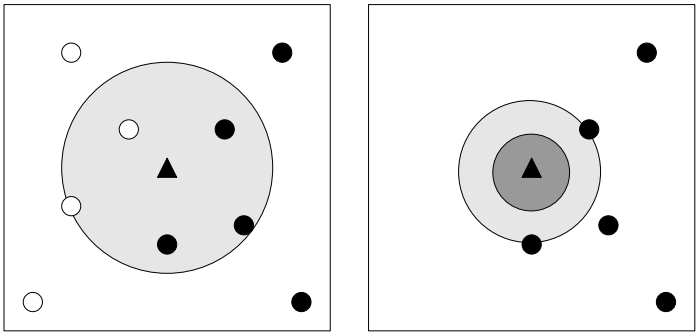


Fig. 1. Visualisation of the k-NN method (on the left) and the *k-threshold classification* (Modified MBL)(on the right)

After identification of possible beginnings and endings we have to identify pairs of them defining borders of subsequent annotations. The important domain factor is that the annotations do not cross, i.e. the beginning of the next one cannot be located between the beginning and the end of the previous one. This fact helped in constructing a heuristic procedure of finding pairs.

First, the highest values of similarity which were found during classification are stored for each classified object. Next sequences (possibly of the length one) of token classified as possible beginnings are identified in text. Then, inside each sequence the hill-climbing algorithm is used to find the token with the highest similarity. For the selected token, in the next step we are looking to the right for a token that can be an ending boundary. When we search for the ending boundary we consider only tokens that appear on the positions from the *starting boundary*

position + a *minimum length of the annotation to the ending boundary position* + a *maximum length of the annotation*. The values of the *minimum length of the annotation* and the *maximum length of the annotation* are calculated during the learning process.

4 Experiment

4.1 Training Set

For the needs of experiments, a set of training examples was extracted from documents concerning stock exchange domain. Each document contains a report posted by a stock issuer. In Poland every unit that issues stocks is obligated by law to present current and periodical information about the issuer according to act [4]. The act defines 26 types of information that the issuer is obligated to publish. During the several months of the year 2006 we collected a set of documents including more then 10000 documents. At the first step the documents were filtered in order to separate those describing the annual meeting of stockholders. 390 documents were found by simple keyword searching. Then each document was manually annotated in terms of the *date*, the *time* and the *place* of a meeting.

Finally, the training set consisted of **424** instances of the meeting date, **383** instances of the meeting time and **385** instances of the meeting place.

4.2 Base Line

Because it was hard to find a exactly similar work (especially for Polish), as a base line we chose the performance of the manually created regular expressions. We wrote a set of rules to extract each type of annotations (time, date and place) from plain text. A sample regular expression for meeting place:

```
1 (w|we)\s*
2 (?<annotation>
3   (siedzibie\s*[sS]polki\s*((,\s*)?w\s*)?)?\w*(,|\s*przy)\s*
4  (?<street>ul\.\s*\w*\s*(nr\s*)?(?<number>\d*(/\d*)?))
5 )
```

Table 3. Performance of the regular expressions

	Precision	Recall	F-measure
Meeting time	93.75	97.91	95.78
Meeting date	59.57	96.76	73.75
Meeting place	17.91	16.10	19.96

The construction of the regular expression took several working days. The results of the their application are presented in Table 3.

In the case of relatively simple patterns of time expression, the result is quite good. The outstanding low results for the *meeting place* annotation were caused by the two facts. First, for each type of annotation a 2-hours time limit was set on the work spent on the manual construction of the expression. The *meeting place* annotation is a combination of smaller elements like a city name, a postal code, a street name and number, a building name and others. The variety of the annotation elements (the order and the number) caused that it was difficult to create a good regular expression during the 2-hours limit time.

In the base line we consider only the performance for whole annotation recognition, we did not constructed and tested expressions for the identification of beginnings and endings, separately.

Table 4. Number of positive and negative instances for the meeting date annotation

	Positives	Negatives	Total
Balanced	424	424	848
1%	424	3015	3439
10%	424	30157	30581
100%	424	301579	302003

4.3 MBL Approach

Applying the original MBL, we have conducted 3 experiments with different number of negatives instances used in the training set. Each case was testes using 10-fold Cross Validation and the minimum, average and maximum vales of precision, recall and F-measure are presented. In the first run we used balanced number of positives and negatives instances. For each positive instance one randomly taken negative instance was used. In the following runs we used 1%, 10% and 100% of all negative instances.

In the original MBL approach we consider only the performance for starting and ending boundaries recognition, see Table 5 and Figure 2 (MBL precision, recall and F-measure), as the first results of the application of the heuristic search methods were not encouraging.

4.4 Modified MBL

In the modified MBL approach we calculated the performance for both starting and ending boundaries, as well, as whole annotation recognition (Tables 6, 7, 8).

Table 5. Original MBL with the k=3, all positive and 1% of all negative examples

	Begin			End			Annotation		
	Precision	Recall	F-measure	Precision	Recall	F-measure	Precision	Recall	F-measure
Min.	10,42	96,08	18,87	9,79	96,43	17,83	18,75	39,22	25,68
Avg.	14,78	98,84	25,61	13,37	99,31	23,52	22,24	46,30	29,92
Max.	20,33	100,00	33,56	16,09	100,00	27,72	26,67	54,90	35,90

Table 6. Results for the *meeting time* using modified MBL with the *threshold*=0.75

	Begin			End			Annotation		
	Precision	Recall	F-measure	Precision	Recall	F-measure	Precision	Recall	F-measure
Min.	84,444	90,244	90,476	81,395	87,500	84,337	97,143	85,000	90,667
Avg.	92,789	97,041	94,802	85,499	91,575	88,402	99,714	89,378	94,240
Max.	97,619	100,000	97,619	92,857	97,674	94,382	100,000	94,872	97,368

Table 7. Results for the *meeting date* using modified MBL with the *threshold*=0.75

	Begin			End			Annotation		
	Precision	Recall	F-measure	Precision	Recall	F-measure	Precision	Recall	F-measure
Min.	86,667	83,673	85,417	81,356	91,667	86,275	92,593	80,392	88,636
Avg.	92,167	88,269	90,116	88,085	95,106	91,408	98,332	85,184	91,230
Max.	97,826	93,333	92,784	96,667	98,039	96,667	100,000	90,000	94,737

Table 8. Results for the *meeting place* using modified MBL with the *threshold*=0.75

	Begin			End			Annotation		
	Precision	Recall	F-measure	Precision	Recall	F-measure	Precision	Recall	F-measure
Min.	40,000	85,000	55,474	51,220	45,238	50,602	44,828	29,268	36,923
Avg.	50,549	90,636	64,725	74,487	60,505	65,919	62,971	40,036	48,580
Max.	57,813	97,561	70,476	96,296	78,571	78,873	81,481	53,659	64,706

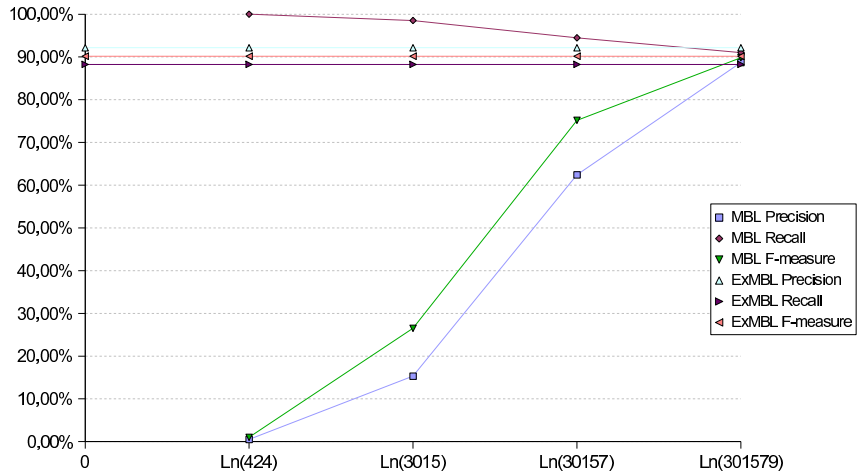


Fig. 2. The impact of the negative instances

5 Conclusions and Future Work

Event Recognition is a specific case of Named Entity Recognition and relies on recognizing events and their attributes. Event Recognition can be treated as a single classification problem as well as multi-classification problem. In both cases the negative instances (non-boundary tokens) are strongly prevalent. The experiment shown that the best results using Memory Based Learning were obtained when all instances (positives and negatives) were used in the training process. Reduction of the negative instances caused the decrease of the performance.

We proposed a simple modification of the original Memory Based Learning methods that uses only of the positive instances. The obtained results for the modified method are comparable to the original MBL results but only when in MBL all examples are used (Figure 2). However, when we decrease the number of negative examples, the MBL methods stops working in the case of this particular tasks and the domain. Contrary to this, the Modified MBL method produces good results for the number of training data was reduced about 1000 times. It means that time of processing is reduced enormously: the modified MBL works on fly, while the application of MBL for the whole training corpus takes days.

The experiments showed that for this particular task, a simple memory based learning can be a solution competitive to manually constructed rules, as a limited corpus is required which can be annotated in short time, the annotation of documents is much easier than the construction of rules, and the method works efficiently.

As we have only the TaKIPI tagger in our disposal, we wanted to stay with as simple features as possible. However, the natural direction of the further research is enrichment of the description of text structure, and adding semantic properties

of word forms. We want also to investigate different similarity functions, e.g. applying some domain heuristics.

References

1. The "Message Understanding Conference (MUC)" web page `http://www-nlpir.nist.gov/related_projects/muc`
2. Scott W. Bennett, Chinatsu Aone, and Craig Lovell. Learning to tag multilingual texts through observation. *Proceeding of the Second Conference on Empirical Methods in NLP*, page 8, 1997.
3. Walter Daelemans and Antal van den Bosch. *Memory-Based Language Processing*. Cambridge University Press, 2005.
4. Dz.U.05.209.1744. *Rozporządzenie ministra finansów z dnia 19 października 2005 r w sprawie informacji bieżących i okresowych przekazywanych przez emitentów papierów wartościowych, dziennik ustaw z 2005 r. nr 209 poz. 1744,* `http://www.abc.com.pl/serwis/du/2005/1744.htm`
5. Xiaoshan Fang and Huanye Sheng. *Advances in Information Systems: Second International Conference, ADVIS 2002, Izmir, Turkey, October 23-25, 2002. Proceedings*, chapter Pattern Acquisition for Chinese Named Entity Recognition: A Supervised Learning Approach, pages 166–175. Springer Berlin / Heidelberg, 2002.
6. A. Kupść., A. Marciniak, A. Mykowiecka, J. Piskorski, and T. Podsiady-Marczykowska. Information extraction from mammographic reports. In *KONVENS 2004, Osterischeen Gesellschaft fur Artificial Intelligence*, pages 113–116, 2004.
7. Derwojedowa Magdalena, Piasecki Maciej, Szpakowicz Stanisław, and Zawisławska Magdalena. Polish wordnet on a shoestring. In W G. Rehm, A. Witt, and L. Lemnitzer, editors, *Proceedings of Biannual Conference of the Society for Computational Linguistics and Language Technology*, pages 169–178. Universität Tübingen, 2007.
8. Tom M. Mitchell. *Machine Learning*. McGraw-Hill Science/Engineering/Math, 1997.
9. Maciej Piasecki and Grzegorz Godlewski. Effective architecture of the polish tagger.
10. Jakub Piskorski. *Intelligent Media Technology for Communicative Intelligence*, chapter Named-Entity Recognition for Polish with SProUT, pages 122–133. Springer Berlin / Heidelberg, 2005.
11. Sameer Pradhan, Kadri Hacioglu, Wayne Ward, James H. Martin, and Daniel Jurafsky. Semantic role chunking combining complementary syntactic views. In *Proceedings of the 9th Conference on Computational Natural Language Learning (CoNLL)*, pages 217–220, 2005.
12. Adam Przepiórkowski. *The IPI PAN Corpus: Preliminary version*. Institute of Computer Science, Polish Academy of Sciences, Warsaw, 2004.
13. Ellen Riloff. Automatically generating extraction patterns from untagged text. *Thirteenth National Conference on Artificial Intelligence (AAAI-96)*, page 1044–1049, 1996. Portland, OR.
14. Kiyoshi Sudo, Satoshi Sekine, and Ralph Grishman. *An improved extraction pattern representation model for automatic ie pattern acquisition, CiteSeer*, page 8, 2003.
15. Gökhan Tür. *A Statistical Information Extraction System For Turkish*. PhD thesis, Bilkent University, 2000.

Proceedings of the International Multiconference on
Computer Science and Information Technology, pp. 285 – 292
ISSN 1896-7094
© 2007 PIPS

Collecting Polish-German Parallel Corpora in the Internet

Monika Rosińska

Adam Mickiewicz University in Poznań
monika.rosinska@gmail.com

Abstract. Parallel corpora have recently become indispensable resources in multilingual natural language processing. Manual preparation of a bilingual corpus is a laborious task. Therefore methods for the automated creation of parallel corpora are currently a topic of concern for many researches. A number of sophisticated and effective algorithms for collecting parallel texts from the Internet have already been created. The aim of the research has been to verify the efficiency of existing algorithms for the collection of Polish-German parallel corpora, intended as a reference source for a Machine Translation system, and possibly, to propose a new algorithm – best suitable for the task.

1 Introduction

Parallel corpora – sets of corresponding texts in two or more languages – seem to be an indispensable resource for widely comprehended natural language processing. Melamed (2000) and Brown (1990) have proved the need for parallel corpora as an essential resource for the preparation of language statistical models in statistical machine translation. Gale and Church (1991) showed the importance of parallel corpora in automatic lexical acquisition.

Unfortunately, preparation of large high-quality corpora is not an easy task. "Handmade" corpora are usually limited in size, very expensive in preparation and not as representative as they should be. That is why most researches attempt to develop efficient and fully automated mechanisms that could yield demanded parallel corpora in relatively short time.

One of the problems that researches have come across is the appropriate choice of source of texts. Developers of latest mechanisms have chosen Internet as such a source because of its vast size, dynamic nature (the content of websites is continually updated), availability and multilinguality.

The most popular algorithms that use the Web as a repository of bilingual texts are: STRAND (Resnik 1999, Resnik and Smith 2003), BITS (Ma and Liberman 1999) and

PTMiner (Chen and Nie 2000). The authors of the algorithms have published results for various pairs of languages – however none of the tests included the Polish-German pair.

In our experiment we wanted to learn how existing algorithms would deal with the Polish-German pair and what innovations might be done to make the algorithms more efficient for this specific pair.

2 Definitions

2.1 Definition of the Corpus

The most general definition of a text corpus – "corpus is a set of texts" – seems not detailed enough for language researches. They are of the opinion that not every set of texts might be regarded as a corpus. That is why a more descriptive definition has to be specified by listing distinctive features of a text corpus.

Some of the criteria proposed by Tony McEnery and Andrew Wilson (1996) are the following:

- representativeness
- finite size
- machine-readable form

On the other hand, distinctive features of corpora considered by John Sinclair (1991) as the most important are:

- high quality
- infinite size
- detailed documentation.

2.2 Types of Corpus

We can distinguish various types of text corpus, depending on their structure, origin, source of texts, types of texts (specialized or general), purpose of creation. The most useful types for machine translation are:

Parallel corpus.

A parallel corpus is a set of texts in two or more languages that meets a criterion of text linkage (this feature distinguishes them from the collection of monolingual corpora in a few languages). Texts must be linked to one another on a defined level: document, paragraph, sentence phrase or word. That means that each part of a text (a document, a paragraph, a sentence a phrase or a word, depending on the level of linkage), in language A must correspond to an appropriate part of text in language B.

Indexed corpus.

An indexed corpus is a set of texts that contains additional data (called metadata) that stores information about the content of the corpus. The information may concern linguistic categories of individual words, the origin of the text or text formatting. The most popular standard for the annotation of parallel corpus is TMX (Translation Memory eXchange format `http://www.lisa.org/tmx/`).

2.3 Web Crawler

Crawler, called also a robot or an Internet spider, is a program that crawls through the Internet in search of reference materials on websites. Its main task is to gather information about visited websites and to download specified content of websites.

3 Parallel Corpus Finding Algorithms

The creation of a bilingual corpus consists of a few stages:

1. Determine required attributes of the corpus (languages, source of texts, typeof formalism, etc.). Those attributes significantly influence consecutive stages of the algorithm.
2. Collect pairs of texts that seem to be mutual translations in earlier determined languages.
3. Clean up collected texts by removing unnecessary information.
4. Align texts, i.e. link parts of text on the defined level.
5. Add crucial metadata and wrap up the texts into the right formalism.

The most difficult steps are 2 and 4. Here, we will discuss step 2 – collecting parallel texts. The first problem that has to be solved is to choose an appropriate source of texts. There are many possibilities (magazines that are translated into various languages, official European Union documents, etc.) but it has been verified by P. Resnik (1999), M. Liberman, X. Ma (1999) and many others, that the most promising source for obtaining appropriate texts is the Internet. In addition to other virtues (such as mentioned above: vast size, dynamic nature and multilinguality) texts in the Internet are easily available and have a computer-readable format. Thanks to computer-readable format the whole process of corpus creation can be made fully automatically.

The most commonly used algorithms for research purposes are STRAND (Resnik and Smith, 2003), BITS(Ma and Liberman, 1999) and PTMiner (Chen and Nie, 2000).

3.1 STRAND

STRAND is the algorithm created by Philipp Resnik. It is based on the assumption that corresponding texts have alike URL addresses and similar web page layouts (HTML tags architecture). A draft of STRAND (version from 1999) is presented below.

1. Use search engine (like AltaVista or Google) to find websites that might be parallel and in desired languages.
2. Verify language of each website
3. Look for candidate pairs in the set of collected websites: candidate pairs should have similar URLs (differing from each other only with the language marker, for example `http://google.com` and `http://google.de`), should be of similar sizes and should be written in different languages.
4. Verify the candidate pair – check if the HTML structure of both websites is identical.

STRAND (proposed in 1999) achieved very promising results while searching for English-French pairs (98% recall (The proportion of relevant documents that are retrieved, out of all relevant documents available.) and 97.4% precision (The proportion of retrieved and relevant documents to all the documents retrieved.)).
In 2003 Resnik came to the conclusion that his assumptions are too general and enriched STRAND with content-based matching functions (the source code is unpublished yet). Moreover, STRAND has now its own mechanism for candidate websites searching (crawler). Those changes allow better results for other pairs of languages achieving.

3.2 BITS

The BITS algorithm has been developed by Xiaoy Ma and Mark Liberman in 1999. The mechanism used for collecting candidate websites was relatively simple. The whole .de domain has been downloaded and for each web-site the algorithm tried to find a parallel page. A draft of BITS is presented below.

1. Download the whole **.de** domain.
2. Generate list of websites that are possibly bilingual.
3. For each element on the list verify if it has a likely correspondent in another language, if so, download the whole content of the correspondent.
4. Clean up collected texts by removing unnecessary information.
5. Create two lists: L1, containing only texts in the source language (German) and L2, containing texts in the target language,
6. For each element in L1 find the most similar element in L2 – using Levenshtein distance (Levenshtein, 1965) or LikeIt distance (Yianilos and Kanzelberger, 1997) .

The main difference between BITS and STRAND is that BITS uses more complex methods to define the parallelism between texts in two languages. The most efficient one uses the LikeIt distance – it is a refined form of the edit distance applied to determine the similarity between two texts (rather than words). Ma and Liberman have used BITS to prepare an English-German corpus. The size of created corpus was 63 MB. The results achieved by BITS were very good – 97.1% recall and 99.1% precision.

4 Research Description

4.1 The Aim of Research

The aim of the research has been to verify the efficiency of existing algorithms for the collection of Polish-German parallel corpora, intended as a reference source for a Machine Translation system, and possibly, to propose a new algorithm – best suitable for the task.

At the beginning we implemented an algorithm based on STRAND. The achieved results were not satisfying due to various factors:

- German translations of Polish web pages were usually of poor quality,
- the addresses of bilingual services seldom followed any naming convention,
- web pages were highly noised.

Therefore, in the second step we enriched the algorithm with some linguistic methods inspired by BITS. The results were better but the algorithm still needs improvements.

4.2 The Algorithms

During the research two specialized algorithms have been implemented. Both implementations have the same module architecture (the modules are presented in 4.2.1, 4.2.2, 4.2.3). However, they use different methods for finding pages and verifying the similarity of candidates.

The main steps in both implementations are following:

1. Create a set of starting URLs.
2. Crawl recursively through the Internet and find as many links as possible.
3. Add found links to the database.
4. For each link define its web page language and size.
5. Find candidate pairs in the database.
6. Verify similarity of candidate pages.
7. Clean up verified pages and add them to the corpora.

4.2.1 Parallel Pages Finder

In the first algorithm we followed the STRAND assumption: the URL addresses of parallel websites should differ with each other only on language suffix (that is why the Levenshtein distance (Levenshtein, 1965) between parallel URLs should not exceed 5). Unfortunately, in case of Polish-German web pages we observe a lack of a naming convention for parallel web pages. Only a part of parallel web-pages meet this criterion. Therefore in the second implementation the crawler looks for Polish and German pages that meet the only criterion of similar size. Their parallelness is verified in next stages.

4.2.2 Language Recognizer

LanguageRecognizer is the module that determines the language of a page. The method we have implemented consists of two main steps:

In the first step we use heuristic methods based on occurrence of national characters. For example, if one of the characters: ą, ć, ę, ł, ś appears in the text, it is assumed that the text is in Polish.

The second step of recognition consults a statistical language model, prepared beforehand. The model consists of trigrams of letters, and their probabilities of occurrence in texts of the modeled language.

4.2.3 Similarity Verifier

Firstly we have implemented a method based on the Resnik assumption that parallel web pages have similar HTML tags structure. Unfortunately, the achieved results were not satisfying. Inspired by methods used in BITS we tried to implement some content based methods to verify if candidate web pages are really equivalent.

One of the ideas that we implemented was the analysis of numbers. We verified whether the same numbers appear in the same order in the candidate pages.

In future we want to improve this method by dividing web page content into smaller pieces of text and then stemming and using dictionary to find out if those content are parallel.

5 Results

Firstly, we developed a crawler and used STRAND to filter candidates for parallel texts. The crawler has visited 9225 websites and has found there 521664 links. 46422 were marked as parallel ones. The results were not satisfactory – only about 10% found candidates were indeed parallel pages.

Therefore we decided to combine STRAND with methods inspired by the LikeIt distance applied in BITS (a source code of LikeIt has not been published). The

crawler visited 3583 websites and found 290264 links 104496 of them were recognized as parallel. This improved results significantly – the accuracy increased to about 40%. Unfortunately the speed of the improved crawler was much lower due to more complex methods that verify the similarity of candidate web pages.
Still, the results are clearly worse than those achieved by Resnik using pure STRAND for English and Basque or English and French.

6 Conclusion and Future Plans

The experiment draws us to the following conclusions:

- The accuracy of existing algorithms used for language pairs like Polish and German, where neither of languages is leading in the Internet is much lower than those obtained if one of the languages is English.
- Collection of Polish-German parallel corpora is hindered by the information noise appearing in German translations of Polish texts. The information noise has strong impact on all of linguistic methods used in the algorithm. The methods must be well adjusted to overcome the problem.
- The starting web links strongly influence the efficiency of a crawler.

According to these conclusions we intend to improve the similarity verifier and develop new methods for choosing the appropriate set of starting URLs for the crawler. The similarity methods will be enhanced by some latest metrics used to strings comparison and dictionary methods.

References

1. Brown P., J. Cocke, S. Della Pietra, V. Della Pietra, F. Jelinek, R. Mercer, and P. Roossin. 1990. *A statistical approach to machine translation,* Computational Linguistics, 16(2):79-85.
2. Melamed I. Dan. 2000. *Models of translational equivalence among words,* Computational Linguistics, 26(2):221-249, June.
3. Gale William A. and Kenneth W. Church. 1991. *Identifying word correspondences in parallel texts,* In Fourth DARPA Workshop on Speech and Natural Language, Asilomar, California, February.
4. Resnik Philip. 1998. *Parallel strands: A preliminary investigation into mining the Web for bilingual text,* In Proceedings of the Third Conference of the Association for Machine Translation in the Americas, AMTA-98, in Lecture Notes in Artificial Intelligence, 1529, Langhorne, PA, October 28-31.
5. Ma, Xiaoyi and Mark Liberman. 1999. *Bits: A method for bilingual text search over the web,* In Machine Translation Summit VII, September. `http://www.ldc.upenn.edu/Papers/MTSVII1999/BITS.ps`

6. Chen Jiang and Jian-Yun Nie. 2000. *Web parallel text mining for Chinese English cross-language information retrieval,* In International Conference on Chinese Language Computing, Chicago, Illinois.
7. Levenshtein V. I.: *Binary codes capable of correcting deletions, insertions and reversals,* Doklady Akademii Nauk SSSR, 163 (No. 4, 1965).
8. Peter N. Yianilos, Kirk G. Kanzelberger *The LikeIt intelligent string comparison facility,* Technical Report, NEC Research Institute, May 1997.

Proceedings of the International Multiconference on
Computer Science and Information Technology pp. 293–303
ISSN 1896-7094
© 2007 PIPS

A Rule Based Approach to Temporal Expression Tagging

Paweł Mazur[1,2] and Robert Dale[2]

[1] Institute of Applied Informatics, Wrocław University of Technology
Wyb. Wyspiańskiego 27, 50-370 Wrocław, Poland
Pawel.Mazur@pwr.wroc.pl
[2] Centre for Language Technology, Macquarie University,
NSW 2109, Sydney, Australia
{mpawel, rdale}@ics.mq.edu.au

Abstract. In this paper we present the DANTE system, a tagger for temporal expressions in English documents. DANTE performs both recognition and normalization of the expressions in accordance with the TIMEX2 annotation standard. The system is built on modular principles, with a clear separation between the recognition and normalisation components. The interface between these components is based on our novel approach to representing the local semantics of temporal expressions. DANTE has been developed in two phases: first on the basis of the TIMEX2 guidelines alone, and then on the ACE 2005 development data. The system has been evaluated on the ACE 2005 and ACE 2007 data. Although this is still work in progress, we already achieve highly satisfactory results, both for the recognition of temporal expressions and their interpretation (normalisation).

1 Introduction

The task of temporal expression recognition and normalisation involves identifying, within texts, expressions that refer to points or periods of time, and re-expressing these temporal references in a standard format which (a) precisely describes the semantics of the expressions, (b) disambiguates dates and times from different time zones, and (c) makes it easier to determine the sequencing of events described in these texts.

The time expression normalisation task is an interesting and challenging one because, while some temporal references appear in well-defined formats, others are expressed using a wide range of natural language constructions, and are often ambiguous, requiring analysis of the surrounding text in order to arrive at an interpretation.

Of course, there are cases where information external to a document—perhaps contained in another document, or best considered part of world knowledge—is required in order to interpret a temporal expression; such cases are not considered here.

There have always been sections of the linguistics, philosophy and natural language processing communities that have been interested in temporal referring expressions. However, interest in the recognition and interpretation of these expressions has grown significantly as a result of the DARPA-sponsored competitions in named entity recognition from the mid-1990s onwards. In contrast to earlier work in the area, these competitions and related exercises introduced a rigorous evaluation paradigm, whereby success

or failure was measured in terms of the ability of software systems to replicate human 'gold standard' annotations of the scope and interpretation of temporal referring expressions.

Undoubtably, the key events and exercises that have played a role in this growth have been the Message Understanding Conferences (MUCs) in 1996 and 1998, and three workshops associated with the Automatic Content Extraction (ACE) program[3] in 2004, 2005 and 2007. While both MUC evaluations covered only recognition of two types of temporal expressions (dates and times), there has been a significant increase in the level of task difficulty in the ACE competitions. The fundamental move forward here was the addition of a normalisation task to the recognition task: annotations were provided for the interpretation of dates and times by using TIMEX2, a slightly modified version of ISO 8601, as the standard for the representation of normalized dates and times. The introduction of TIMEX2 also influenced the recognition task, as the range of the temporal expressions to be recognised was broadened significantly as compared to the MUC-6 and MUC-7 task definitions.

Subsequently, the TIMEX2 standard has evolved through a number of versions, partially due to the wide interest it has received in the community, and the existence of the ACE program and similar competitions. This has also resulted in quite a large number of temporal expression taggers being constructed by the participants in these competitions. Details of the current, and most likely final, version of the annotation standard are provided in [4].

In this paper we present the DANTE (Detection And Normalisation of Temporal Expressions) system, which, as its name suggests, performs both recognition and normalisation of temporal expressions. Currently, the system works only for English texts; however, its extension to other languages is facilitated by its modular architecture, where some components are language independent. In January 2007, DANTE participated in the ACE Time Expression Recognition and Normalization (TERN) task.

The rest of this paper is organized as follows. First, in Section 2 we briefly introduce the TIMEX2 annotation scheme. In Section 3, we describe related work, briefly presenting other existing temporal expression taggers. Then, in Section 4, DANTE's system architecture and development process is discussed. Section 5 provides information on DANTE's performance both in terms of recognition and normalisation results, and in terms of resource consumption and execution time. In Section 6 we discuss the most problematic cases for DANTE that give rise to errors in the current version. Conclusions and future work are described in Section 7.

2 The TIMEX2 Annotation Scheme

The TIMEX2 scheme provides an inline SGML tag, `TIMEX2`, for annotating temporal expressions. Annotations can be nested, as shown in Example (1).

(1) I'm leaving on vacation `<TIMEX2 VAL="1999-08-03">`two weeks from `<TIMEX2 VAL="1999-07-20">`next Tuesday`</TIMEX2></TIMEX2>`.

[3] See `http://www.nist.gov/speech/tests/ace`.

Table 1. Attributes in TIMEX2.

Attribute	Description
VAL	Contains a normalized value in ISO-like format of the date or time of the annotated expression.
MOD	Captures temporal modifiers, using values such as BEFORE, AFTER, LESS_THAN, MORE_THAN, EQUAL_OR_LESS, START, MID, END or APPROX.
ANCHOR_VAL	Contains a normalized value in ISO-like format of an anchoring date or time.
ANCHOR_DIR	Captures the relative direction or orientation between VAL and ANCHOR_VAL attributes, as in WITHIN, STARTING, ENDING, AS_OF, BEFORE or AFTER. It is used to express information about *when* a duration is placed.
SET	Identifies expressions denoting sets of times; either takes the value YES or is empty.
COMMENT	Contains any comment that the annotator wants to add to the annotation; ignored from the point of view of automatic processing of the text.

There are six attributes defined in the scheme for the tag; values of attributes express the semantics of an expression. A description of the attributes is given in Table 1. In this context, the recognition task is about finding boundaries of expressions in texts, and the normalisation task requires determining values of TIMEX2 attributes.

3 Related Work

The earliest approaches, typical of work undertaken for MUC-6, were based on the construction of hand-crafted rules using a grammatical formalism that would match both fixed-format dates and times, and a range of expressions in natural language within the scope defined in the guidelines. For MUC-7, there were both solutions based on transducers, such as those described by [9] and [6], and also other techniques, such as Hidden Markov Models as used in IdentiFinder [10].[4] In both MUC competitions, the results achieved for TIMEX recognition by the best systems were high:

- at MUC-6, Recall of 93% and Precision of 96%; and
- at MUC-7, Recall of 89% and Precision of 99% for dates, and Recall of 81% and Precision of 97% for times [6].

TempEx (see [7]) was the first TIMEX2 tagger developed. It is a relatively simple Perl tool that implements a few heuristics based on part-of-speech tags using finite state automata. It also performs limited normalisation of the expressions. The most recent version, from December 2001, implements the 2001 version of the TIMEX2 standard. There are certain classes of phrases that are not recognized by this tool: for example *the last Monday of January*, *the end of 1999*, and *late yesterday morning*. This tool was provided to all participants of ACE 2004 for use as an external source of text features; as such, it provides a reasonable baseline for performance on new data.

GUTime [14] was developed as an extension of TempEx for the purpose of constructing an automatic temporal annotation tool for TimeML (see [12]). TimeML is

[4] See also [1] for an extended description.

Table 2. The F-measure results for GUTime, ATEL and LingPipe on ACE 2004 data.

	Detection	Extent Recognition	VAL Attribute
GUTime	85	78	82
ATEL	90.4	81.5	–
LingPipe	89.1	75.8	–

a sophisticated schema for the annotation of events; its complexity means that automatic tagging of events is best achieved via a cascade of modules that successively add more and more TimeML annotations to the document being processed. In this context, GUTime is the module responsible for the detection of temporal expressions and the introduction of the TIMEX3 tag into the annotations. GUTime's coverage of temporal expressions is greater than that of TempEx. It also handles TIMEX3's functional approach to expressing values: i.e., for relative expressions it first identifies the function realised by an expression (for example, for *tomorrow* this would be 'plus one day'), with the actual value of that function (for example, *25th January 1996*) being calculated at a later stage.

Chronos [11] is a more complex system designed to perform both recognition and normalisation of temporal expressions. Text processing in Chronos involves tokenization, statistical part-of-speech tagging and multiwords recognition based on a list of 5000 entries retrieved from WordNet. Then, the text is processed by a set of approximately 1000 basic rules that recognize temporal constructions and gather information about them that is expected to be useful in the process of normalization. This is followed by the application of composition rules, which resolve ambiguities when multiple tag placements are possible. The results in terms of F-measure on ACE 2004 data are 92.6%, 83.9%, 87.2% for detection, recognition and determining the VAL attribute value, respectively.

The increasing availability of corpora annotated with temporal expressions makes it possible to apply supervised machine learning techniques to the time expression recognition problem. Examples of such systems are ATEL [5] and Alias-i's LingPipe.[5] The former is based on Support Vector Machine (SVM) classifiers, and the latter is constructed using a Hidden Markov Model (HMM). Results are summarised in Table 2.

4 System Architecture and Development

We take the view that an important step towards a truly broad coverage yet semantically well-founded approach is to recognize that there is a principled distinction to be made between the interpretation of the semantics of a temporal expression devoid of its context of use, and the fuller interpretation of that expression when the context is taken into account. The first of these, which we refer to here as the **local semantics** of a temporal expression, should be derivable in a compositional manner from the components of the expression; determining the value of the second, which we refer to as the **global semantics** of the expression, may require arbitrary inference and reasoning.

[5] See http://www.alias-i.com/lingpipe.

Such a distinction is implicit in other accounts: Schilder's [13] use of lambda expressions allows representation of partially specified temporal entities, and the temporary variables that Negri and Marseglia [11] construct during the interpretation of a given temporal expression capture something of the same notion.

The above assumptions are reflected in our design, which comprises separate and independent modules for the recognition and normalisation subtasks. These components communicate via an intermediate format for expressing the local semantics of temporal expressions, as described in [8] and [3].

The stages of text processing are organized as a pipeline of processing resources using the architectural constructs provided in GATE [2]. The elements in our pipeline are a tokenizer, gazetteers, a sentence splitter, a POS tagger, named entity recognition, temporal expression recognition, and temporal expression interpretation.

4.1 Temporal Expression Recognition

The temporal expression recognizer is implemented using GATE's JAPE grammar formalism. The grammar consist of five phases which are run over a document in sequence. Each phase contains rules which match annotations introduced by earlier processing components (for example, the tokenizer or POS tagger) and JAPE grammar phases. There is also one initial additional phase which consists only of macros used in the grammar rules. Altogether there are 80 macros and 250 rules. Macro expansions are textually copied into the bodies of rules, and then the rules are compiled into Java code.

JAPE rules are traditional pattern–action rules, where the left-hand side contains the pattern to be matched, and the right-hand side specifies the action to be taken when the pattern is matched. The pattern on the left-hand side is written using JAPE syntax, but the right-hand side can be implemented either in JAPE or directly in Java code. Our recognition rules use 31 gazetteers with a total of 1418 entries: these are strings used in the expression of dates and times, such as numbers written in words; the names of days, months and time zones; and the most common fractions.

The development of our temporal expression recognition module took two and a half person months. The module was developed on the basis of the TIMEX2 guidelines and the examples contained therein; then we tested DANTE on the ACE 2005 development data and identified frequently-occurring cases which were problematic for the system. Addressing these problems constituted a second stage of system development.

4.2 Local Semantics

The recognition phase outputs the extents of temporal expressions and their local semantics. We use this term to refer to a level of representation that corresponds to the semantic content that is derivable directly from the text representation of temporal expressions; in the case of temporal expressions that are arguments to prepositions, this includes the interpretation of the preposition as well.[6]

[6] In the TIMEX2 annotation scheme, these prepositions are not included within the markable extent of temporal expressions.

The abstract model we developed for representing the local semantics uses attribute–value matrices, while its implementation uses a string-based format, deliberately similar to the format of TIMEX2 annotations; see [8] and [3] for more details and examples of this representation.

4.3 Temporal Expression Interpretation

The interpreter module is a process that steps through a document sentence by sentence. The local semantic representation of each recognized temporal expression is transformed into a document-internal semantic representation, which we call the global semantics, and which in DANTE is expressed by means of TIMEX2 annotations. The interpreter is implemented entirely in Java and includes a library of functions for various calculations on dates and times. This module took approximately one and a half person months to develop.

In our current model, we assume that a document has a simple linear structure, and that any hierarchical structure in the document has no bearing on the interpretation of temporal expressions; for present purposes we also make the simplifying assumption that the **temporal focus** used to compute document-level values for temporal expressions does not advance during the processing of the document. Both assumptions may not always hold true, but appear to work for the majority of cases we are dealing with.

Depending on the type of the temporal expression being interpreted (fully specified points in time, underspecified points in time, relative expressions, durations, frequencies and so on), different actions are taken. The two basic operations used in the interpretation are unification with some reference date and the addition or subtraction of a specified number of units to or from a reference date.

5 Evaluation

The most significant evaluations of DANTE to date are our participation in the ACE 2007 TERN task, and our subsequent re-evaluation of the system on the same data after further development on the ACE 2005 development data set.

The execution time for our text processing modules is presented in Table 3 as measured on a laptop with a 2GHz Intel Core Duo processor and 2GB of available RAM; only one core of the processor was used for processing documents. In characterising the processing cost, we do not take into account initialization of the system, the exporting of results into XML files, and the postprocessing required to meet the ACE formatting requirements, including the conversion of results from our inline XML annotation into the APF XML format.

Memory consumption during system execution is to some extent dependent on the size of the processed document, but on the ACE 2007 evaluation the variation was not great (from 116MB to 126MB). The system also required approximately 15MB of disk space to store the input corpus. The ACE 2007 evaluation data consisted of 254 documents from six different domains (see Table 3). As one might expect, documents were not equally distributed across the domains, neither in terms of the number of documents

Table 3. Execution times on the ACE 2007 eval data set.

Domain	No of docs	Time [s]	Av. time per one doc [s]	Approx. size [B]	Av. time per 10kB [s]
Broadcast Convers.	9	10.902	1.211	48,722	2.29
Broadcast News	74	15.983	0.216	75,731	2.16
Newswire	106	43.632	0.412	209,973	2.13
Telephone Convers.	6	12.221	2.037	54,522	2.30
Usenet Newsgroups	13	11.398	0.877	48,377	2.41
Weblogs	46	29.355	0.638	137,549	2.19
Total	254	123.491	0.486	574,874	2.20

Table 4. Attribute value recognition evaluation for DANTE on ACE 2007 evaluation data.

TIMEX2 Attribute	Precision	Recall	F-Measure
VAL	99.8%	98.0%	98.9%
MOD	76.0%	75.0%	75.5%
SET	100.0%	100.0%	100.0%
ANCHOR_VAL	88.4%	83.5%	85.9%
ANCHOR_DI R	88.1%	87.4%	87.8%

Table 5. The results of evaluation of the DANTE system on the ACE 2007 evaluation data set.

Domain	Entities in gold standard	Spurious	Missing	Error	Precision	Recall	F-measure	ACE Value
Broadcast Convers.	142	33	29	43	47.9	49.3	48.6	46.5
Broadcast News	322	103	38	69	55.6	66.8	60.6	55.2
Newswire	894	128	110	273	56.0	57.2	56.6	58.8
Telephone Convers.	70	23	11	25	41.5	48.6	44.7	51.4
Usenet Newsgroups	167	20	22	43	61.8	61.1	61.4	65.3
Weblogs	433	68	58	139	53.3	54.5	53.9	57.3
Total	2028	375	268	592	54.7	57.6	56.1	57.2

Table 6. The official results of the ACE 2007 evaluations.

Site	Overall	Broadcast Conversations	Broadcast News	Newswire	Telephone Conversations	Usenet	Weblogs
A	61.6	44.2	68.4	67.4	52.6	63.1	51.4
B	59.3	48.2	68.6	60.9	60.2	58.2	52.9
C	58.2	46.6	67.8	57.3	64.2	59.0	54.8
D	48.3	30.0	44.4	54.2	38.7	55.9	44.8

nor in terms of the total size of documents in a domain. We ran the system for each document source type separately in order to identify variations in performance across the different domains.

In the ACE evaluations a correctly recognized time expression is one which has a strictly accurate extent and correct values for all the TIMEX2 attributes. An annota-

tion generated by the system is classified as matched with an annotation from the gold standard if there is at least a 30% text span overlap between them.

The ACE 2007 evaluation data included 2028 time expressions to be recognized and interpreted. Across all domains we currently achieve 54.7, 57.6 and 56.1 for precision, recall and F-measure, respectively, for correct recognition of temporal expressions. After applying weights[7] to particular elements which are subject to evaluation, the scores are 69.7, 69.2 and 69.4 respectively; the overall ACE TERN score for DANTE is 57.2. These results indicate that DANTE's performance is already very close to state-of-the-art systems (see Table 6 with the official ACE 2007 results[8]). For 13 documents in the corpus we scored 100%, which means that all time expressions in these documents were recognised and interpreted correctly and the system did not make any spurious identifications.

The precision, recall and F-measure metrics for specific attributes are presented in Table 4, calculated for those expressions which matched with the gold standard. Table 5 presents detailed performance of DANTE across all domains.

6 Error Analysis

In order to determine which aspects of DANTE most need attention, we analysed the errors made by the system on the ACE 2005 development data set; this is larger than the evaluation data set, containing 5428 temporal expressions (as annotated in the gold standard).

6.1 Errors in Recognition

Using the evaluation tool provided by NIST for use in the ACE program we have found that the errors in recognition of temporal expressions can be broken down as follows:

- 1056 spurious matches (51.09% of our errors),
- 586 missing temporal expressions (28.35%), and
- 425 extent errors (20.56%).

Based on an analysis of what falls into the set of spurious matches, we observe that about 50% of these are in fact due to legitimate temporal expressions that are missing from the gold standard. For the remaining 50%, DANTE's errors are generally due to ambiguity in the meaning of some expressions. Most of these are expressions based on the following trigger words: *now*, *fall*, *a second*, *night*, *May*, *March* and expressions which contain numbers wrongly recognised as years, dates or hours. In most of these cases the fix should be quite straightforward.

[7] In the ACE 2007 TERN evaluations the weights were as follows: 1.0 for type VAL, 0.5 for ANCHOR_VAL, 0.25 for ANCHOR_DIR, 0.1 for MOD, 0.1 for SET, 0.1 for extent (where there is at least a 30% overlap between matched elements; otherwise elements are not mapped at all). The cost for spurious TIMEX2 mentions was -0.75.

[8] See `http://www.nist.gov/speech/tests/ace/ace07/doc/ace07_eval_official_results_20070402.htm`

Among those strings which are not recognised as temporal expressions, most errors are due to either the ambiguous trigger word *time* or expressions whose extent can only be determined by syntactic means, as in *four days after Americans first penetrated the Baghdad outskirts*. In the latter case, DANTE only recognises the string *four days* as a temporal expression, and since this corresponds to less than 30% of the total length of the gold-standard expression, this is not treated by the scoring tool as a matched expression. There is also large group of expressions which appear with very low frequency (the 'long tail'), and which were not therefore considered a priority when developing DANTE.

An analysis of those cases where DANTE identified a temporal expression with an incorrect extent shows that the problems are due to failures to recognise the following: some variations of time zones; modified expressions (for example in *just recently* we recognised only *recently*); and expressions built from smaller constituent expressions (for example, in *around 11:30 Saturday night* we incorrectly recognise the time and date as separate expressions).

6.2 Errors in Interpretation

For the interpretation task, i.e., the determination of values for the TIMEX2 attributes, the error statistics are as follows (for each attribute, these show the numbers of expressions with an incorrect value):

- 1460 for the VAL attribute (69.00%),
- 1067 for the ANCHOR_VAL attribute (50.43%),
- 897 for the ANCHOR_DIR attribute (42.39%),
- 192 for the MOD attribute (9.07%), and
- 53 for the SET attribute (2.50%).

The total number of expressions with at least one incorrect attribute–value was 2116.

In determining the correct value of the VAL attribute, the biggest problem is interpreting names of weekdays such as *Tuesday*, where we often get a date one week earlier or later than the correct date. However, in the development data we observed that in about 15% of cases where there is a difference in the value generated by DANTE and that provided in the gold standard, the gold standard appears to be incorrect. Other errors in the gold standard further weaken the reliability of these numbers. Our worst results are obtained for the MOD attribute: many expressions are not given a value at all or they are given the wrong value. There are also cases when DANTE interprets expressions as modified, but they are not according to the gold standard annotators. Less problematic is the SET attribute, as it is quite obvious which expressions refer to more than one point in time, and the attribute's value is binary.

7 Conclusions and Future Work

We have presented the DANTE system for recognition and interpretation of temporal referring expressions in English natural language texts. The system has been evaluated

on the ACE 2007 evaluation corpus, which is a data set widely accepted by the community as a gold standard for the temporal expression recognition and interpretation task. The results achieved are good enough to enable DANTE's use in many applications that require the interpretation of temporal expressions in text processing, such as information extraction and question answering.

The evaluation has brought to light several areas where DANTE can be improved. Our error analysis indicates that the following steps will be the most important in producing a more robust solution:

- First, we need to further develop the recognition grammar. This will require both the addition of vocabulary to our existing rules, and also the development of new rules covering previously unseen structures. As the system is rule-based, this also requires careful testing to ensure that the addition or modification of rules does not introduce any incompatibilities or inconsistencies in the grammar.
- Second, we need to improve our mechanism for focus tracking in documents in order to more accurately resolve ambiguities. Although using the document creation date as the temporal focus often works fairly well, it is not reliable enough alone for a state of the art temporal expressions tagger.
- Although execution time performance is not critical in an evaluation such as this, we are keen to develop a robust and scalable solution. Our third task will therefore include identifying scalability improvements to DANTE.

8 Acknowledgements

We gratefully acknowledge the support of the Defence Science and Technology Organisation in the work described here.

References

1. D. M. Bikel, R. Schwartz, and R. M. Weischedel. An Algorithm that Learns What's in a Name. *Machine Learning*, 34:211–231, 1999.
2. H. Cunningham, D. Maynard, K. Bontcheva, and V. Tablan. GATE: A framework and graphical development environment for robust NLP tools and applications. In *Proceedings of the 40th Anniversary Meeting of the ACL*, 2002.
3. Robert Dale and Paweł Mazur. Local Semantics in the Interpretation of Temporal Expressions. In *Proceedings of the Workshop on Annotating and Reasoning about Time and Events (ARTE)*, pages 9–16, Sydney, Australia, July 2006. Association for Computational Linguistics.
4. L. Ferro, L. Gerber, I. Mani, B. Sundheim, and G. Wilson. TIDES 2005 Standard for the Annotation of Temporal Expressions. Technical report, MITRE, September 2005.
5. K. Hacioglu, Y. Chen, and B. Douglas. Automatic Time Expression Labeling for English and Chinese Text. In Alexander F. Gelbukh, editor, *Computational Linguistics and Intelligent Text Processing, 6th International Conference, CICLing'05*, LNCS, pages 548–559. Springer, 2005.
6. G. Krupka and K. Hausman. IsoQuest Inc.: Description of the NetOwl(TM) Extractor System as Used for MUC-7. In *Proceedings of the 7th Message Understanding Conference (MUC-7)*, Fairfax, Virginia, 1998.

7. I. Mani and George Wilson. Robust Temporal Processing of News. In *ACL '00: Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*, pages 69–76, Morristown, NJ, USA, 2000. Association for Computational Linguistics.
8. Paweł Mazur and Robert Dale. An Intermediate Representation for the Interpretation of Temporal Expressions. In *Proceedings of the COLING/ACL 2006 Interactive Presentation Sessions*, pages 33–36, Sydney, Australia, July 2006. Association for Computational Linguistics.
9. A. Mikheev, C. Grover, and M. Moens. Description of the LTG System Used for MUC-7. In *Proc. of MUC-7 Conf.*, 1998.
10. S. Miller, M. Crystal, H. Fox, L. Ramshaw, R. Schwartz, R. Stone, R. Weischedel, and the Annotation Group. BBN: Description of the SIFT System as Used for MUC-7. In *Proceedings of the 7th Message Understanding Conference (MUC-7)*, Fairfax, Virginia, 1998.
11. M. Negri and L. Marseglia. Recognition and Normalization of Time Expressions: ITC-irst at TERN 2004. Technical Report WP3.7, Information Society Technologies, February 2005.
12. J. Pustejovsky, B. Ingria, R. Sauri, J. Castano, J. Littman, R. Gaizauskas, A. Setzer, G. Katz and I. Mani. The Specification Language TimeML. In I. Mani, J. Pustejovsky, and R. Gaizauskas, editors, *The Language of Time: A Reader*. Oxford University Press, 2004.
13. F. Schilder. Extracting Meaning from Temporal Nouns and Temporal Prepositions. *ACM Transactions on Asian Lang. Information Processing*, 3(1):33–50, March 2004.
14. M. Verhagen, Inderjeet Mani, Roser Sauri, Jessica Littman, Robert Knippen, Seok B. Jang, Anna Rumshisky, John Phillips and James Pustejovsky. Automating Temporal Annotation with TARSQI. In *Proceedings of the ACL Interactive Poster and Demonstration Sessions*, pages 81–84, Ann Arbor, Michigan, June 2005. Association for Computational Linguistics.

Proceedings of the International Multiconference on
Computer Science and Information Technology pp. 305–315
ISSN 1896-7094
© 2007 PIPS

Truncated Importance Sampling for Reinforcement Learning with Experience Replay

Paweł Wawrzyński, Andrzej Pacut

Warsaw University of Technology, Institute of Control and Computation Engineering
Nowowiejska 15/19, 00-665 Warsaw, Poland
p.wawrzynski@elka.pw.edu.pl, a.pacut@ia.pw.edu.pl
WWW home page: http://staff.elka.pw.edu.pl/~pwawrzyn/

Abstract. Reinforcement Learning (RL) is considered here as an adaptation technique of neural controllers of machines. The goal is to make Actor-Critic algorithms require less agent-environment interaction to obtain policies of the same quality, at the cost of additional background computations. We propose to achieve this goal in the spirit of *experience replay*. An estimation method of improvement direction of a changing policy, based on preceding experience, is essential here. We propose one that uses truncated importance sampling. We derive bounds of bias of that type of estimators and prove that this bias asymptotically vanishes. In the experimental study we apply our approach to the classic Actor-Critic and obtain 20-fold increase in speed of learning.

1 Introduction

We consider the problem of a controller (or a decision maker) that optimizes its reactive policy in a poorly structured and initially unknown environment. The general solution to this problem is Reinforcement Learning (RL). Algorithms in that area can be viewed as computational processes that transform observations of states, actions, and rewards, into policy parameters. Popular RL algorithms, like Q-Learning [17], and Actor-Critic methods [2, 6, 7], process the data sequentially. A single observation is utilized for an adjustment of the algorithms' parameters and becomes unavailable for further use. Let us call such methods *single-adjustment algorithms*. This sequential processing approach follows a common understanding that applications of RL to real life learning control problems require large amounts of data which cannot be kept in a limited memory assigned to the algorithm.

However, there are methods, not of single-adjustment type, that require much less environment steps to obtain a policy of the same quality. They achieve that at the cost of collecting data and their extensive processing. Let us call them *multi-adjustment algorithms* as they are not limited to a small number of operations after each environment step. Note that we do not identify single-adjustment- and on-line methods neither we identify multi-adjustment- and off-line algorithms. We are here interested *only* in on-line algorithms in the sense that they should optimize the policy as the agent-environment interaction is going

on. The difference lies rather in that how busy is the computer that optimizes the policy during that interaction.

Multi-adjustment algorithms can be roughly divided into three classes. The most obvious way to design the computations for a multi-adjustment algorithm is to estimate the model of the environment dynamics and use it in some variant of dynamic programming [14, 5, 1]. The second approach to multi-adjustment algorithms is as follows. Assume there exists a quality index $J : \Re^{n_\theta} \mapsto \Re$ that assigns to each policy parameter $\theta \in \Re^{n_\theta}$ a value $J(\theta)$ that evaluates the policy based on θ. Let us denote by $\widehat{J}_t(\theta)$ an estimator of $J(\theta)$ based on the course of the MDP before time t. The objective of data processing can be to optimize the policy by maximizing $\widehat{J}_t(\theta)$ with respect to θ [13, 10]. Within the third approach, a multi-adjustment algorithm is constructed in reference to a single-adjustment method: It takes historical events and applies to them the operations of the original single-adjustment method as if the events have just taken place. This idea of repeating many similar operations to the same event, called *experience replay* [3], was popular a few years ago but seems to receive less attention recently. In our paper we show how to apply this idea to a wide class of RL methods that includes Actor-Critics.

The approach proposed in the present paper is as follows. A single-adjustment RL algorithm defines a drift of the policy parameter θ. Within the process of computing previous experience, the original path of θ is *simulated* such that the parameter is adjusted along estimators of the direction of the drift. The estimators are based on the observations of preceding agent-environment interaction collected in a database. Our main concern here is how to estimate the direction of the drift on the basis of historical data.

2 Problem formulation

We will consider the standard RL setup [15]. A Markov Decision Process is a tuple $\langle \mathcal{S}, \mathcal{A}, P_s, r, P_0, \mathcal{S}^* \rangle$ where $\mathcal{S}$ and $\mathcal{A}$ are the state and action spaces, respectively; $\{P_s(\cdot|s,a) : s \in \mathcal{S}, a \in \mathcal{A}\}$ is a set of state transition distributions; we write $s_{t+1} \sim P_s(\cdot|s_t, a_t)$ and assume that each P_s is a density. The immediate reward, r_t, depends on the action and the next state, $r_t = r(a_t, s_{t+1})$. P_0 is the distribution of initial states of each episode and $\mathcal{S}^*$ is the set of terminal states. Whenever $s_t \in \mathcal{S}^*$, the next state s_{t+1} is drawn from P_0.

Actions are generated according to a policy, π, which is a family of distributions parameterized by the state and a *policy vector* $\theta \in \Re^{n_\theta}$, namely

$$a_t \sim \pi(\cdot\,; s_t, \theta).$$

The objective of reinforcement learning is, in general, to make the policy maximize future rewards by optimizing θ. The strict meaning of this goal may be specified in various ways. We may require the policy to maximize the average reward or to maximize the expected sum of rewards within an episode. Alternatively, we may require the policy to maximize the sum of future discounted rewards expected in each state.

We are interested in applications of the MDP framework to adaptive control tasks. We thus require the learning algorithm to be fast, i.e. to obtain a satisfying policy after as few control steps as possible. The control cost should be minimized and the controlled machine should be kept from being damaged by inappropriate actions. We assume that the time span of learning is short enough to keep the entire agent-environment interaction history in a database. This history is available between each pair of consecutive steps for a limited number of operations aiming at policy optimization.

Below we analyze a large class of the existing incremental RL methods suitable for policy determination in simulations and propose a way of their acceleration based on a more extensive data processing. Our intention is to design methods that obtain a satisfying control policy after a much smaller amount of control time but not necessarily after a smaller amount of computation.

2.1 Single-adjustment Methods with Actor

Here we analyze a large class of RL algorithms defined by the following features:

1. Actions are generated by a stationary policy (Actor), i.e. a distribution π parametrized by state s_t and the policy parameter θ, namely

$$a_t \sim \pi(\cdot\,; s_t, \theta).$$

2. A visit in state s_t causes a modification of the policy parameter θ by a vector

$$\beta_t^\theta \widehat{\phi}_t.$$

   $\widehat{\phi}_t$ on the average indicates the direction in which θ assures larger future rewards expected in state s_t. $\{\beta_t^\theta, t = 1, 2, \dots\}$ is a vanishing sequence of step-sizes.
3. The algorithm may compute $\widehat{\phi}_t$ with the use of an auxiliary parameter $\upsilon \in \mathfrak{R}^{n_\upsilon}$ (like the weights of Critic, i.e. a value-function approximator). A visit in state s_t results in a modification of υ by a vector

$$\beta_t^\upsilon \widehat{\psi}_t.$$

   $\widehat{\psi}_t$ on the average points into the direction where υ assures better quality of $\widehat{\phi}_t$. $\{\beta_t^\upsilon, t = 1, 2, \dots\}$ is a vanishing sequence of step-sizes.
4. The vectors $\widehat{\phi}_t$ and $\widehat{\psi}_t$, while different from one another, are of the same form

$$G_t(\theta, \upsilon) \sum_{k \geq 0} (\alpha\rho)^k z_{t,k}(\theta, \upsilon) \tag{1}$$

   where G_t is a certain vector, $\alpha \in [0, 1)$, $\rho \in [0, 1)$, and $z_{t,k} \in \mathfrak{R}$ is unknown until instant $t + k + 1$.

Typically, the step-sizes β_t^θ and β_t^υ satisfy the standard stochastic approximation conditions [8], namely

$$\sum_{t\geq 1}\beta_t = +\infty, \qquad \sum_{t\geq 1}\beta_t^2 < +\infty. \tag{2}$$

Obviously a number of RL algorithms fit into the above schema. In the classic Actor-Critic algorithm with $TD(\lambda)$-errors [6] we have

$$\widehat{\phi}_t = \frac{\partial \ln \pi(a_t; s_t, \theta)}{\partial \theta^T} \sum_{k\geq 0}(\gamma\lambda)^k \frac{\beta_{t+k}^\theta}{\beta_t^\theta}\widehat{e}_{t+k}, \qquad \widehat{\psi}_t = \frac{\partial \bar{V}(s_t;\upsilon)}{\partial \upsilon^T} \sum_{k\geq 0}(\gamma\lambda)^k \frac{\beta_{t+k}^\upsilon}{\beta_t^\upsilon}\widehat{e}_{t+k},$$

where $\bar{V}$ is an approximation of the value function parametrized by υ, $\gamma \in [0,1)$ is a discount factor, $\lambda \in (0,1)$, and

$$\widehat{e}_t = r_t + \gamma\bar{V}(s_{t+1};\upsilon) - \bar{V}(s_t;\upsilon) \tag{3}$$

is the *temporal difference*. More recent Actor-Critics [6, 7] generally fit into the discussed schema.

Analysis of RL methods usually assumes that for each policy parameter θ, the sequence of states $\{s_t, t = 1, 2, \dots\}$ forms an aperiodic and irreducible Markov chain with a stationary distribution. In our presentation of the mechanics of the incremental RL algorithms, we will also assume that the states visited when the policy parameter θ is applied are drawn independently from a stationary distribution. A reader interested in a deeper analysis and proofs of convergence is referred to [9, 7].

Let us analyze the average direction of $\widehat{\phi}_t$ and $\widehat{\psi}_t$. Namely, let ϕ be a function defined as

$$\phi(s,\theta,\upsilon) = \mathcal{E}_{\theta,\upsilon,\beta}\left(\widehat{\phi}_t \middle| s_t = s\right). \tag{4}$$

The definition of ϕ is based on the assumption that θ, υ, and the step-sizes remain constant when $\widehat{\phi}_t$ is calculated. In fact they slightly vary and each $\widehat{\phi}_t$ is in fact a biased estimator of $\phi(s_t,\theta,\upsilon)$ for θ and υ used at time t. However, this bias is small and since the dynamics of the parameters decreases in time, the bias asymptotically vanishes. ϕ averaged over the steady state distribution, approximates the drift of θ and the smaller the step-sizes, the better the approximation.

The drift of υ may be analyzed in a similar way. Namely, let ψ be a function defined as

$$\psi(s,\theta,\upsilon) = \mathcal{E}_{\theta,\upsilon,\beta}\left(\widehat{\psi}_t \middle| s_t = s\right). \tag{5}$$

As above, the definition of ϕ requires that θ, υ, and the step-sizes remain constant during the time when $\widehat{\psi}_t$ is computed. The approximation of the drift of υ is defined by ψ averaged over the steady state distribution.

The incremental RL algorithms make their parameters θ and υ follow the directions defined by estimators of $\phi(s,\theta,\upsilon)$ and $\psi(s,\theta,\upsilon)$, respectively, averaged over the steady-state distribution.

2.2 Experience Replay Based Acceleration

The main idea analyzed in this paper is to apply to the agent's experience the same treatment as a single-adjustment algorithm with Actor would do, but more intensively. Namely, let us consider a certain algorithm of this type. It defines vectors $\widehat{\phi}_t$ and $\widehat{\psi}_t$ and corresponding functions ϕ and ψ, respectively. A generic accelerated version of this algorithm has the form of the following loop:

1. Draw and execute an action, $a_t \sim \pi(\cdot\,; s_t, \theta)$.
2. Register s_t, a_t, θ, the reward received and the next state in a database.
3. Perform a certain number of times:
   3.1. Draw a state $s_i, 1 \leq i < t$ from the database.
   3.2. Adjust θ along an estimator of $\phi(s_i, \theta, \upsilon)$.
   3.3. Adjust υ along an estimator of $\psi(s_i, \theta, \upsilon)$.
4. Assign $t := t + 1$ and repeat from Point 1.

In words, the initial single-adjustment algorithm uses ϕ to define a drift of the policy parameter θ. The direction of this drift is an approximation of a policy gradient or a natural policy gradient. The drift itself can be *simulated* in a parallel computation process (Point 3. above) while the Markov Decision Process is going on. This way the original algorithms can be substantially accelerated.

In the following section, we discuss importance sampling as the main tool to construct estimators of ϕ and ψ. In Section 4, those estimators are constructed.

3 Importance Sampling Estimation

Let $g(\cdot\,;\theta)$ be a density of random elements a of values in $\mathcal{A}$ parametrized by a vector $\theta \in \Theta \subset \mathfrak{R}^{n_\theta}$. Suppose for a given parameter θ_0 an action a_0 is drawn from $g(\cdot\,;\theta_0)$. The action a_0 yields a certain vector $d(a_0)$ where $d : \mathcal{A} \mapsto \mathfrak{R}^n$. We are interested in estimation of the expected $d(a)$ for actions drawn with the use of an arbitrary parameter θ, namely

$$\mathcal{E}_\theta d(a) = \int\limits_{\mathcal{A}} d(\alpha) g(\alpha;\theta)\,\mathrm{d}\alpha = D(\theta). \tag{6}$$

In order to construct an estimator of $D(\theta)$, we may apply the estimation technique called *importance sampling* [12] frequently applied in RL [11, 13, 10, 16]. Namely, given θ_0 and θ, the statistic

$$d(a_0)\, g(a_0;\theta)/g(a_0;\theta_0) \tag{7}$$

is an unbiased estimator of $D(\theta)$ since

$$\mathcal{E}_{\theta_0}\left(d(a_0)\,\frac{g(a_0;\theta)}{g(a_0;\theta_0)}\right) = \int d(\alpha)\frac{g(\alpha;\theta)}{g(\alpha;\theta_0)}\, g(\alpha;\theta_0)\,\mathrm{d}\alpha = D(\theta). \tag{8}$$

A well-known drawback of estimator (7) is its excessive variance when distributions $g(\cdot\,;\theta_0)$ and $g(\cdot\,;\theta)$ are far from one another [19]. A simple yet satisfying in

practice way of fighting the variance problem is to truncate the density ratio in (7). Namely, the modified estimator is of the form

$$d(a_0)\min\{g(a_0;\theta)/g(a_0;\theta_0),\, b\} \tag{9}$$

for constant $b > 1$. The above *truncated IS estimator* has also been applied in [16, 18]. Let us analyze its properties. Obviously if $\|d\|$ is bounded, then also the variance of (9) is bounded. Importantly, its bias happens to be bounded as well if only g satisfies certain regularity conditions and θ is close enough to θ_0. Let us first specify those conditions. Namely, for the purpose of this discussion, a density g will be called (M_1, M_2)-*regular* if it has the following features:

(i) for every $a \in \mathcal{A}$ the mapping $\theta \mapsto g(a;\theta)$ is continuous and differentiable,
(ii) the trace of the Fisher information, namely $\operatorname{tr} I(\theta) = \mathcal{E}_\theta \frac{\partial \ln g(a;\theta)}{\partial \theta} \frac{\partial \ln g(a;\theta)}{\partial \theta^T}$, is bounded by M_1,
(iii) absolute eigenvalues of the Hesian $\frac{\partial^2 \ln g(a;\theta)}{\partial \theta^T \partial \theta}$ are bounded by M_2.

The below proposition establishes bounds of the bias of estimator (9).

Proposition 1. *Let the density g parameterized by θ be (M_1, M_2)-regular for certain M_1, M_2. Then, for all θ_0, θ,*

$$\left\| \mathcal{E}_{\theta_0}\left(d(a_0)\min\left\{ \frac{g(a_0;\theta)}{g(a_0;\theta_0)}, b \right\}\right) - D(\theta) \right\| \le M P_\theta\left(\frac{g(a;\theta)}{g(a;\theta_0)} > b \right).$$

where $M = \sup_{a\in\mathcal{A}} \|d(a)\|$.

A slightly different form of the above fact is proved on p. 34 of reference [18]. ■
Note that P_θ is the probability defined by the condition that the random value a is drawn from the density parameterized by θ. Another proposition defines a bound of P_θ.

Proposition 2. *For each $M_1, M_2 > 0$, there exists $M > 0$ such that if the density g is (M_1, M_2)-regular than, for all θ_0, θ,*

$$P_\theta\big(g(a;\theta)/g(a;\theta_0) > b\big) \le M(\ln b)^{-2}\|\theta - \theta_0\|^2.$$

This proposition is proved on p. 35 of reference [18]. ■
In the present work we shall also discuss a special case of the analyzed problem, where d is a function of a finite sequence of independent random elements. Obviously, both the above propositions apply to this case because a sequence of random vectors is also a random vector. However, a joint density of a random sequence need not be (M_1, M_2)-regular even if densities of all its elements are. This rises difficulties in application of Proposition 2. The proposition below treats this issue.

Proposition 3. *For each $M_1, M_2 > 0$, there exists $M > 0$ such that if all densities in the sequence $g^1, \ldots, g^k$ are (M_1, M_2)-regular than for each set of pairs $(\theta^i, \theta_0^i), i = 1, \ldots, k$*

$$P_{\theta^1 \ldots \theta^k}\left(\prod_{i=1}^k \frac{g^i(a^i;\theta^i)}{g^i(a^i;\theta_0^i)} > b \right) \le \frac{k^2 M}{(\ln b)^2} \sum_{i=1}^k \|\theta^i - \theta_0^i\|^2.$$

provided random elements $a^i, i = 1, \ldots, k$ are drawn independently.

Proof (sketch): Let us denote

$$A = \left\{ \prod_{i=1}^{k} \frac{g^i(a^i;\theta^i)}{g^i(a^i;\theta_0^i)} > b \right\}, \quad A_i = \left\{ \frac{g^i(a^i;\theta^i)}{g^i(a^i;\theta_0^i)} > \sqrt[k]{b} \right\}, i = 1, \ldots, k.$$

We have

$$\Omega - A \supseteq \bigcap_{i=1}^{k} (\Omega - A_i) \quad \text{and hence} \quad 1 - P_{\theta^1 \ldots \theta^k}(A) \geq \prod_{i=1}^{k} (1 - P_{\theta^i}(A_i)).$$

After simple transformations we obtain

$$P_{\theta^1 \ldots \theta^k}(A) \leq \sum_{i=1}^{k} P_{\theta^i}(A_i).$$

Application of Proposition 2 completes the proof. ■

4 Experience Replay

In this section, we introduce estimators of $\phi(s_i, \theta, \upsilon)$ and $\psi(s_i, \theta, \upsilon)$ and discuss their properties. These estimators will play the role of stochastic gradients. In general, stochastic gradients are required to be of bounded variance and (asymptotically) unbiased. While unbiased estimators of ϕ and ψ may be of excessive variance, their modifications will be presented that will be of bounded variance and asymptotically unbiased.

In Section 2.1 we defined ϕ and ψ of the form (4) and (5), respectively, for $\widehat{\phi}_t$ and $\widehat{\psi}_t$ having, for constant step parameters, the same generic form

$$G_t(\theta, \upsilon) \sum_{k \geq 0} (\alpha\rho)^k z_{t,k}(\theta, \upsilon) \tag{10}$$

where G_t is a vector known at time t, $\alpha \in [0, 1), \rho \in [0, 1)$ is a given constant, and in a sequence $\{z_{t,k}(\theta, \upsilon), k \geq 0\}$ the value $z_{t,k}$ is known at time $t + k + 1$. Because ϕ and ψ have the same form, further in this section we will focus on ϕ, understanding that exactly the same can be said about ψ.

Let us first denote by $Geom(\rho)$ the geometric distribution with parameter $\rho \in [0, 1)$. That is, random variable K of values in $\{0, 1, \ldots\}$ has distribution $Geom(\rho)$, i.e. $K \sim Geom(\rho)$, iff

$$P(K = m) = (1 - \rho)\rho^m \quad \text{for} \quad m = 0, 1, \ldots$$

A simple property of that distribution is that $P(K \geq m) = \rho^m$. Let an estimator of $\phi(s_i, \theta, \upsilon)$ have the form

$$\widehat{\phi}_i^b(\theta, \upsilon) = G_i(\theta, \upsilon) \sum_{k=0}^{K} \alpha^k z_{i,k}(\theta, \upsilon) \min \left\{ \prod_{j=0}^{k} \frac{\pi(a_{i+j}; s_{i+j}, \theta)}{\pi(a_{i+j}; s_{i+j}, \theta_{i+j})}, b \right\}. \tag{11}$$

Because of truncation, it is easy to verify that its variance is bounded. However, for the same reason, it is also biased. Fortunately, the bias vanishes when the parameters θ_{i+j} of distributions that generated actions get closer to the analyzed value θ. The following proposition clarifies this issue.

Proposition 4. *Suppose G_i and $z_{i,k}$ are uniformly bounded and distributions $\pi(\cdot\,; s_{i+j}, \cdot)$ are (M_1, M_2)-regular for certain M_1, M_2 and all $j \geq 0$. Then, there exists a positive constants c such that*

$$\|\mathcal{E}\widehat{\phi}_i^b(\theta, \upsilon) - \phi(s_i, \theta, \upsilon)\| \leq c\delta^2$$

for all i, θ, υ and

$$\delta = \sup_{j \geq 0} \|\theta - \theta_{i+j}\|.$$

Proof (sketch): Each element of $\widehat{\phi}_i^b$ is a truncated importance-sampling estimator of $\mathcal{E}_{\theta,\upsilon}\big(G_i(\theta, \upsilon) z_{i,k}(\theta, \upsilon) | s_i\big)$. Propositions 1 and 3 define a bound of the bias of these estimators, which is of the form

$$\frac{M(k+1)^2}{(\ln b)^2} \sum_{j=0}^{k} \|\theta - \theta_{i+j}\|^2$$

Because the sum in the above formula is no greater than $(k+1)\delta^2$, we obtain

$$\begin{aligned}\|\mathcal{E}\widehat{\phi}_i^b(\theta, \upsilon) - \phi(s_i, \theta, \upsilon)\| &\leq \mathcal{E}\left(\sum_{k=0}^{K} \alpha^k \frac{M(k+1)^3\delta^2}{(\ln b)^2}\right) \\ &= \sum_{k \geq 0} (\rho\alpha)^k \frac{M(k+1)^3\delta^2}{(\ln b)^2} = \delta^2 \frac{M}{(\ln b)^2} \sum_{k \geq 0} (\rho\alpha)^k (k+1)^3.\end{aligned}$$

Because $\rho\alpha \in (0, 1)$, the value of the last sum is finite which completes the proof. ■

Implementation of the generic algorithm presented in Sec. 2.2 has to meet several postulates. The most important requirement for any algorithm based on stochastic approximation is boundedness of the variance of gradient estimators. That requirement is satisfied when the estimators of ϕ and ψ have the form (11). Second, we want the algorithm to inherit the limit properties from the original single-adjustment method.

The drawback of estimators (11) is that they are biased. However, they can be made asymptotically unbiased. Namely, suppose only a certain constant number, N, of recent events are kept in the database. Then, at the moment t the indexes i are drawn randomly from the set $\{t-N+1, t-N+2, \ldots, t\}$. Let us analyze δ_t as the upper bound of $\|\theta - \theta_{i+j}\|$ for i drawn at moment t. Because the step-sizes β_t^θ vanish in time, the changes of the policy parameter, θ, also vanish. Therefore, the value δ_t must vanish as well, the conditions of Proposition 4 are met and the estimators $\widehat{\phi}_i^b$ and $\widehat{\psi}_i^b$ are asymptotically unbiased.

5 Experimental Study

We apply the idea of simulation of parameters' path in a parallel computation process to the classic Actor-Critic (AC) [6]. For the purpose of this paper we shall call the resulting algorithm the replaying Actor-Critic. We apply this algo-

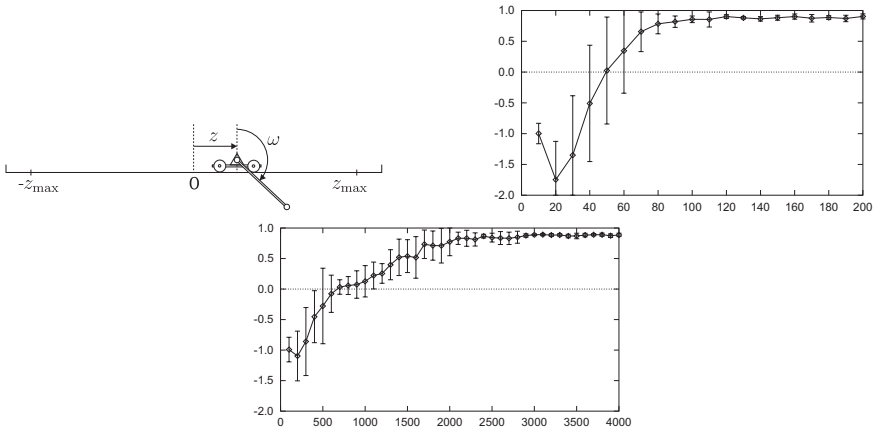


Fig. 1. Actor-Critics for Cart-Pole Swing-Up. *Left:* The plant. *Center:* The classic AC, the average reward vs. trial number. Each point averages 100 consecutive trials and the curve averages 10 runs. The one-sigma limits are calculated to assess run-to-run variability of trial averages. *Right:* The replaying AC, the average reward vs. trial number. Note that the number of trials in this figure is about 20 times smaller than that of the top figure for the classic method. Because of much shorter learning time, each point averages only 10 consecutive trials; the curve averages 10 runs. The one-sigma limits are wider than in the top figure because 10 times less trials are averaged.

rithm to a simple benchmark problem — Cart-Pole Swing-Up [4]. The plant here consists of a cart moving along a track, and a pole hanging freely from the cart (see Fig. 1). It is controlled by force applied to the cart. The control objective is to avoid hitting the track bounds, swing the pole, turn it up, and stabilize upwards. In our experiments the controlled plant is emulated, i.e. simulated in real time. A quantum of plant's time is equal to a quantum of the corresponding computer time. That means that the computer has a lot of spare time that can be devoted to parallel computations. The setup of our experiments is designed to closely resemble a situation in which control of a physical machine is optimized in real time by means of a certain form of adaptation.

The setting of our experiments is the same as of those presented in [18]. The result of application the replaying Actor-Critic is presented in the form of learning curve on the left-hand side of Fig. 1. For comparison we also apply the classic Actor-Critic to the same plant. The corresponding learning curve is presented in the center of Fig. 1.

The replaying AC (see Fig. 1) exhibits a satisfying behavior after just 90 trials (about 20 minutes of the plant's time) while the classic algorithm obtains the same behavior after about 2300 trials. Our experiments with the classic Actor-Critic may also be compared to those presented in [4]. The algorithm introduced there implements an indirect adaptive control method—it estimates a model of an unknown plant and optimizes its control on the basis of the model while the plant is controlled. This algorithm, applied to the Cart-Pole Swing-Up, obtained satisfactory results after about 750 trials (about 2 hours of the plant's real time).

6 Conclusions

In this paper we developed an idea of experience replay based speedup of incremental Actor-Critic-like reinforcement learning methods. A drift of parameters driven by a given incremental RL method may be simulated in a parallel computation process while the agent-environment interaction is going on. This simulation is based on estimators of the direction of parameter drift induced by the original algorithm. We introduced the appropriate estimators based on truncated importance sampling and analyzed their properties, namely bounded variance and asymptotically vanishing bias. In the experimental study we applied our approach to the classic Actor-Critic and obtained a satisfying controller of the Cart-Pole Swing-Up in 20 minutes of time of this plant which is about 20 shorter time than the original Actor-Critic requires.

References

1. P. Abbeel, M. Quigley, & A. Y. Ng: 2006, Using Inaccurate Models in Reinforcement Learning,. *Int. Conf. on Machine Learning.*
2. A. G. Barto, R. S. Sutton, & C. W. Anderson: 1983, Neuronlike Adaptive Elements That Can Learn Difficult Learning Control Problems. *IEEE Trans. on Systems, Man, and Cybernetics,* vol. SMC-13:834-846.
3. P. Cichosz: 1999, An analysis of experience replay in temporal difference learning. *Cybernetics and Systems,* 30:341-363.
4. K. Doya: 2000, Reinforcement learning in continuous time and space. *Neural Computation,* 12, pp. 243-269.
5. M. Kearns & S. Singh: 2002, Near-Optimal Reinforcement Learning in Polynomial Time. *Machine Learning,* Volume 49, Issue 2-3, pp. 209-232.
6. H. Kimura & S. Kobayashi: 1998, An Analysis of Actor/Critic Algorithm Using Eligibility Traces: Reinforcement Learning with Imperfect Value Functions. *Int. Conf. on Machine Learning.*
7. V. R. Konda & J. N. Tsitsiklis: 2003, Actor-Critic Algorithms. *SIAM Journal on Control and Optimization,* Vol. 42, No. 4, pp. 1143-1166.
8. H. J. Kushner & G. G. Yin: 1997, *Stochastic Approximation Algorithms and Applications,* Springer: New York.
9. P. Marbach & J. N. Tsitsiklis: 2001, Simulation-Based Optimization of Markov Reward Processes. *IEEE Trans. on Automatic Control,* Vol. 46, No. 2, pp. 191-209.
10. L. Peshkin & Ch. R. Shelton: 2002, Learning from Scarce Experience. *Int. Conf. on Machine Learning,* pp. 498-505.
11. D. Precup, R. S. Sutton, & S. Singh: 2000, Eligibility Traces for Off-Policy Policy Evaluation. *Int. Conf. on Machine Learning,* Morgan Kaufmann.
12. R. Rubinstein: 1981, *Simulation and The Monte Carlo Method.* New York, Wiley.
13. Ch. R. Shelton: 2001, Policy Improvement for POMDPs Using Normalized Importance Sampling. *Int. Conf. on Uncertainty in Artificial Intelligence,* pp. 496-503.
14. R. S. Sutton: 1990, Integrated Architectures For Learning, Planning, and Reacting Based on Approximating Dynamic Programming. *Int. Conf. on Machine Learning,* pp. 216-224, Morgan Kaufmann.

15. R. S. Sutton & A. G. Barto: 1998, *Reinforcement Learning: An Introduction*, MIT Press, Cambridge, MA.
16. E. Uchibe & K. Doya: 2004, Competitive-Cooperative-Concurrent Reinforcement Learning with Importance Sampling. *Int. Conf. On Artificial Intelligence.*
17. C. Watkins & P. Dayan: 1992, Q-Learning. *Machine Learning*, vol. 8, pp. 279-292.
18. P. Wawrzyński, Intensive Reinforcement Learning, *PhD Dissertation,* Institute of Control and Computation Engineering, Warsaw University of Technology, 2005.
19. P. Wawrzyński & A. Pacut: 2006, Balanced Importance Sampling Estimation. *Int. Conf. on Information Processing and Management of Uncertainty in Knowledge-based Systems (IPMU),* pp. 66-73.

Proceedings of the International Multiconference on
Computer Science and Information Technology pp. 317–326
ISSN 1896-7094
© 2007 PIPS

Adaptive Temporal Planning at Airports

Pim van Leeuwen[1], Lian Ien Oei[2], Pieter Buzing[2], and Cees Witteveen[2]

[1] National Aerospace Laboratory NLR,
Anthony Fokkerweg 2, 1059 CM Amsterdam, The Netherlands
leeuwenp@nlr.nl
[2] Delft University of Technology,
Faculty of Electrical Engineering, Mathematics and Computer Science,
Mekelweg 4, 2628 BZ Delft, The Netherlands
L.I.Oei@student.tudelft.nl, P.Buzing@ewi.tudelft.nl,
C.Witteveen@tudelft.nl

Abstract. Airports are getting more and more congested with ground handling activities during turnaround as one of the most constraining factors. To alleviate this bottleneck, robustness of the planning of these activities is of paramount importance. In this paper, we present a new idea to solve a strategic planning problem in a way that allows unforeseen, real-time disruptions to be handled in a straightforward and elegant manner. To that end we apply Hunsberger's decoupling algorithm to a Simple Temporal Network representation of the ground handling domain.

1 Introduction

Europe-wide flight traffic is expected to double by 2020 [1]. Given this increase in air traffic, airports will become a major bottleneck in the air transport system. As expansion of airports is often impossible, airport authorities are seeking methods to increase airport capacity by making more efficient use of existing resources. Currently, one of the most important factors preventing this is the occurrence of disruptions, of which many occur daily. Without re-planning, these disturbances may cause a chain reaction and lead to grave consequences. Therefore, re-planning has become a daily necessity.

Given the complex and distributed nature of the airport domain, however, re-planning is not an easy task. It typically involves a large group of parties, each having their own (commercial) interests, resources, and planning constraints. Between these parties, many dependencies exist: changing something in one planning domain may have repercussions for all.

In European Union projects, such as LEONARDO [2], it became visible that in an approach that involved all parties, large communication overhead lead to problems when trying to solve tactical and real-time disruption scenarios. Thus, a general tool assisting all planners in solving disturbances at airports in these phases turned out not to be feasible.

Airport planning is typically subdivided into a number of domains: arrival management, departure management, stand allocation management, taxi planning. In all of these areas, extensive research has been conducted to improve

planning and assist planners by means of decision support tools (e.g., [2, 3]). Ground handling, denoting all processes that take place when an aircraft is at the stand[3] between flights, is a notable exception. Up until now, not much research has been conducted in this area. This is surprising, since ground handling is recognised as a common and important source of delays in the air transport system, second after only air traffic control (ATC) related delays [4].

This paper presents a new planning approach to support airport authorities in the establishment of a robust pre-tactical[4] ground handling plan for servicing scheduled aircraft. These services (e.g., boarding, fuelling, cleaning and baggage loading) are performed during the so-called *turnaround*: the process of (un)loading and servicing an aircraft at the stand between two flights. Usually several agents (such as a boarding operator, fuelling operator, catering operator) are involved in this process, each performing a part of it.

In our approach the turnaround process is modelled as a Simple Temporal Problem (STP) [5]. We propose a decomposition framework to solve tactical disruptions to the pre-tactical plan, which offers additional flexibility in repairing such disruptions. First, this temporal plan can be decomposed into a task-based temporal plan and a resource constraint system. This allows us to distinguish task repair from resource repair actions. Furthermore, using a groups of actors, we use a temporal decoupling technique to decouple the temporal plan into independently solvable sub plans that allow parties to solve many tactical disruptions locally, making plan co-ordination and negotiation between parties superfluous. Given the large number of plan disruptions occurring daily at airports, and the increase expected in air traffic, such a plan repair tool seems a valuable asset.

The outline of this paper is as follows. In Section 2 we present the basics of the STP approach and its application to ground handling processes on airports. In Section 3 we present our plan repair framework. In Section 4 we conclude and present some suggestions for further research.

2 Background

2.1 The airport turnaround process

The airport turnaround process comprises all ground handling activities that need to be performed at an aircraft when parked at a stand between two flights. These activities need to be performed between in-block (when the aircraft arrives at the stand) and off-block (when the aircraft leaves the stand). Many different ground handling actors (also called *service providers*) are involved to perform these activities (e.g., the cleaning operator, and the fuelling operator). They all have their own (commercial) interests, resources, and planning constraints.

When planning their activities, these actors have to comply with a large number of temporal dependencies that exist between these activities (e.g., catering

[3] The term *stand* will be used for both gates and remote stands.

[4] The strategic and pre-tactical planning phases range from half a year until two hours before the day of operation. After that the tactical and real-time phases follow.

and cleaning should take place between de-boarding and boarding). Furthermore, the ground handlers have to take into account so-called *norm times* (specified by airlines), and *minimum service times* (specified by aircraft manufacturers), that prescribe for each aircraft type the start time and duration of each activity.

2.2 Simple Temporal Problems and temporal decoupling

Since the planning of the turnaround process involves a large number of temporal dependencies, we now discuss the *Simple Temporal Problem (STP)* [5] formalism for representation of and reasoning with temporal problems. In general, a *temporal problem* is a problem in which time constraints are involved. In particular, an STP consists of a finite set $X = \{x_0, \ldots, x_n\}$ of time point variables (representing events), and a set of constraints $C = \{c_{ij} \mid i, j \in 0, \ldots, n\}$ between these variables. A constraint c_{ij} is represented by an interval $I_{ij} = [a_{ij}, b_{ij}]$, abbreviating the inequalities $a_{ij} \leq x_j - x_i \leq b_{ij}$ or the expression $x_j - x_i \in [a_{ij}, b_{ij}]$, that indicates that x_j has to occur at least a_{ij} and at most b_{ij} time units after x_i. The time point x_0 is a special time point, called the *temporal reference point*, that denotes a fixed point in time. As it is usually assigned the value 0, it is also called the *zero time point variable*, and also referred to as z. A task t_i, which has a duration, can be represented by two events (its start and end), and a constraint that indicates how long the task will take. A tuple $T = (\tau_0, \ldots, \tau_n)$ is called a *solution* if the assignment $\{x_0 = \tau_0, \ldots, x_n = \tau_n\}$ satisfies all the constraints in C. An STP is *consistent* if at least one solution exists.

There are several types of graphs to represent an STP. In each of these, the vertices represent the time point variables and the edges represent the constraints. A *Simple Temporal Network (STN)* is a directed graph in which each edge $x_i \rightarrow x_j$ corresponds to the constraint c_{ij} and is labelled by the interval $[a_{ij}, b_{ij}]$. In a *distance graph* each edge $x_i \rightarrow x_j$ is labelled by a weight a_{ij} and represents the linear inequality $x_j - x_i \leq a_{ij}$. As each constraint c_{ij} corresponds to two such linear inequalities, each edge in the STN corresponds to two edges in the distance graph. If in the distance graph there exists a path from x_i to x_j, then implicitly there also exists a constraint between x_i and x_j, equal to the length of this path, which is the sum of its edge labels.

In a *z-partition* the set of time point variables X is partitioned into two or more subsets $X_1, \ldots, X_n$ that all have only the zero time point variable z in common, and whose union constitutes the original set X. A *temporal decoupling* of the STN $S = \langle X, C \rangle$ is a set of consistent STNs $S_1 = \langle X_1, C_1 \rangle, \ldots, S_n = \langle X_n, C_n \rangle$ such that $X_1, \ldots, X_n$ z-partition X, and any solutions for $S_1, \ldots, S_n$ may be merged to form a solution for the original STN S. In other words, temporal decoupling guarantees that even if each of the sub-STNs S_i is solved completely independent from the other sub networks S_j, the simple union of the individual solutions constitutes a solution of the original network S.

In 2002, Hunsberger [6] developed an algorithm that given an STN $S = \langle X, C \rangle$ and a partitioning of X produces a temporal decoupling of S. Here the global idea of this algorithm will be sketched, using an example with two subnetworks. The general case is analogous. Suppose that STN S is z-partitioned by

S_X and S_Y. An edge in the distance graph representation is called an xy-edge, if it connects a time point x in S_X with a time point y in S_Y. If there exists a path from x to y through z, that has a length equal to or shorter than the length of the xy-edge itself, the xy-edge is said to be dominated by a path through zero and may be removed. The idea of Hunsberger's algorithm now is to add constraints c_{xz} and c_{yz} for each xy-edge, until all xy-edges are dominated by a path through zero, and thus have become redundant and may be removed.

2.3 Modelling the turnaround process as an STN

The airport turnaround process can be modelled as an STN. In this model, for each aircraft, a set of time point variables X can be defined, that contains x_0, the in-block and off-block times, and the start and end times of all ground handling activities that have to be performed. The set of constraints C can be obtained by combining the temporal dependencies between activities with the norm times and minimal service times of all activities. The norm times specify the earliest start time and maximum duration of each activity. The minimal service times specify the minimum duration of the activities. For planning of all ground handling processes at the entire airport, a global STN can be constructed by combining all STNs of individual aircraft. The temporal reference point x_0 (common to all individual STNs) can be used to link the networks. The following example shows how this is done, and how the global STN can be decoupled into separate sub-STNs for each type of ground handling service.

Example 1. Aircraft X (KL310, type B737-300) is scheduled at Stand A17 to go in-block at 12:00 and off-block at 13:15. Aircraft Y (LH200, type MD11) is scheduled at Stand A23 to go in-block at 12:05 and off-block at 14:10. For both aircraft two ground handling services have to be planned: fuelling and boarding. Fuelling has to take place before boarding. Boarding has to end at most 15 minutes before off-block. Table 1 shows which norm and minimal service times apply for these types of aircraft.

Table 1. Norm and minimal service times for Aircraft X and Y.

Activity	Norm start time	Min. service time	Norm duration
B737-300 fuelling	8 min. after in-block	10 min.	37 min.
B737-300 boarding	32 min. after in-block	5 min.	18 min.
MD11 fuelling	11 min. after in-block	17 min.	59 min.
MD11 boarding	80 min. after in-block	16 min.	36 min.

The data for Aircraft X can be modelled as an STN as follows: there are seven time point variables $X = \{x_0, \ldots, x_6\}$, where x_0 = temporal reference point, x_1 = in-block, x_2 = start fuelling, x_3 = end fuelling, x_4 = start boarding, x_5 = end boarding, and x_6 = off-block. Let x_0 = 12:00. From the precedence constraints, and the in-block, off-block, norm and minimal service times, we can

derive the following constraints that exist between these variables: $c_{01} = [0, 0]$, $c_{06} = [75, 75]$, $c_{34} = [0, \infty]$, $c_{12} = [8, \infty]$, $c_{23} = [10, 37]$, $c_{14} = [32, \infty]$, $c_{45} = [5, 18]$, and $c_{56} = [0, 15]$. For Aircraft Y an STN can be obtained in a similar way, after which both STNs can be combined. This global STN is depicted in Figure 1. In this figure, the variables $x_1, \ldots, x_6$ correspond to the plan for Aircraft X, and the variables $y_1, \ldots, y_6$ correspond to the plan for Aircraft Y.

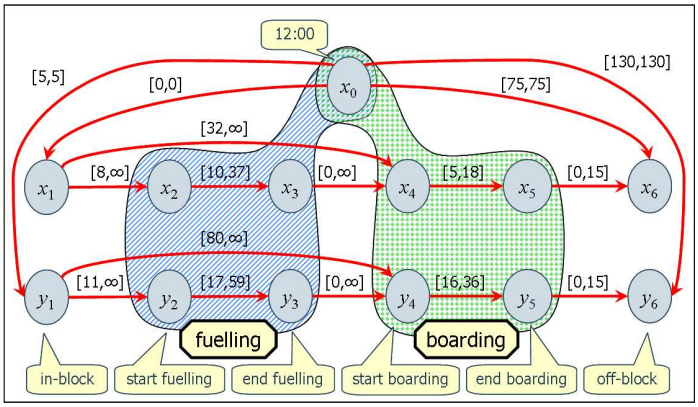


Fig. 1. An STN for Aircraft X and Y and a possible decoupling.

The figure also shows how the STN can be partitioned, such that sub-STNs for the fuelling operators (shaded) and for the boarding operators (spotted) can be decoupled from the network. This can be done using Hunsberger's decoupling algorithm. To decouple a sub-STN, extra constraints have to be added, that make all xy-edges redundant, so they can be removed from the graph. For the fuelling sub-STN these are $x_1 \rightarrow x_2$, $x_3 \rightarrow x_4$, $y_1 \rightarrow y_2$, and $y_3 \rightarrow y_4$. These extra constraints correspond to arrangements the different parties make beforehand, to make sure their plans will not interfere. For example, the fuelling and boarding operator can agree that for Aircraft X fuelling will be finished before 12:56 ($c_{03} = [-\infty, 56]$), whereas boarding will not start before 12:56 ($c_{04} = [56, \infty]$). After adding these two extra constraints, the constraint $x_3 \rightarrow x_4$ can be removed from the network. Once all xy-edges have been removed this way, the fuelling sub-STN can be decoupled from the rest of the network.

3 A Framework for STN-based Plan Repair

As remarked in the introduction, one of most important factors preventing more efficient use of resources in the turnaround process is the occurrence of disruptions. In terms of our STN-model of the turnaround process, a disruption causes a change in the constraints of a temporal event such that the current STN becomes

inconsistent. To restore consistency, we have to adapt the STN by changing one or more constraints. Determining the minimal number of adaptations (minimal repair) to restore inconsistencies is an intractable problem [Buzing, PhD thesis, forthcoming]. We therefore have to rely on *heuristics* to perform plan repair on an STN. An example of such a heuristic is *right-shifting* one or more tasks. This can be achieved by the basic constraint propagation methods in [5]. E.g., suppose x_1 is the in-block time of a flight and $c_{01} = [30, 40]$ is a temporal constraint in C. According to the schedule the aircraft should be at the gate at 12:30. However, we have 10 minutes "slack" time. If the aircraft arrives five minutes late, this can be accommodated by right-shifting the in-block time accordingly.

However, one should not concentrate on minimal repair planning alone. Even more important in applying plan repair in complex application areas, like airport planning, is that a *minimal number of parties* is involved in the plan/schedule repair process. Here, temporal decoupling (described in Section 2.2) can be used to decouple the plan of one party (e.g., the fuelling agent) from all other agents. If a disruption occurs, we can investigate if one the decoupled subnetworks has become inconsistent. If so, we can try to repair these subnetworks locally.

Often, however, it is not possible to repair an inconsistent STN by applying simple repair heuristics and it requires another view of the way the STN has been composed. Therefore in this section we will develop a hierarchical multi-actor STN-framework that enables us *(i)* to determine a minimal subset of actors that should be involved in repairing the current temporal plan and *(ii)* to use a set of plan repair heuristics that can be used by these agents to repair their individual temporal plan without affecting the other parties.

The idea of our model is to decompose the temporal planning problem underlying the resulting STN into a *task component*, a *resource component* and an *actor component*. Based upon this decomposition, a hierarchy of three STNs can be distinguished and the planning of repair actions can be based upon the inconsistency of a subset of them. We will discuss our model in the next subsection.

3.1 Decomposing an STN

In any temporal plan we can distinguish at least three components. First, we have a set of activities or *tasks* that have to be performed. Typically, these have a duration and temporal relations (before, after, during) between tasks. The execution of tasks requires *resources*, e.g., vehicles or processing units. Each of the resources can perform a limited amount of tasks at a time (its capacity). Such capacity constraints often will generate additional constraints besides the temporal constraints. Typically, *agents* have to use these resources to execute tasks they are responsible for. Often, agents are organised in groups where the members within a group are cooperative, whereas groups (companies) are assumed to behave self-interested and as independently from the other groups as possible. Such groups induce a partitioning of the set of agents such that each partition wants to have a plan that can be executed independently from the others.

We will discuss the consequences of this decomposition into task, resource and agent properties. Here we assume that a basic STN $S = \langle X, C \rangle$ has been

given, representing a set of tasks where resource constraints have been accounted for. We will start by decomposing S according to the set of actors/agents.

Actor decomposition Assume we have a basic STN $S = \langle X, C \rangle$ whose tasks have to be carried out by agents using the resources distinguished. Here, we have a set $A = \{A_1, \ldots, A_n\}$ of agents. Each agent A_i belongs to exactly one group of agents $\mathcal{A}_j$, $j = 1, \ldots, k$. These groups $\mathcal{A}_i$ want to be independent in executing their tasks and are responsible for a disjoint subset X_i of variables. Therefore, the set $\mathcal{A} = \{\mathcal{A}_1, \ldots, \mathcal{A}_k\}$ induces a partitioning $[X_1, X_2, \ldots X_k]$ of the temporal variables in S. The idea of *temporal decoupling* S is to construct for each group of actors $\mathcal{A}_i$ its own STN $S_i = \langle X_i, C_i \rangle$ such that any solutions chosen by the groups $\mathcal{A}_i$ for their own STN can always be merged into a total solution of S.

Regarding plan repair, temporal decoupling creates a repair hierarchy: if a disruption occurs, the original STN S might remain consistent, while one or more of the sub-STNs S_i are inconsistent. Then either one of the groups should try to repair its plan, e.g., by right shifting its activities, or some parties should try to reconsider the decoupling of their activities. If the original STN S is also inconsistent - and thus one or more sub-STNs are also inconsistent, the repair cannot be achieved by the reassignment of activities to the parties involved alone.

Task and resource decomposition If we abstract from the set of (independent) actors involved, a basic STN $S = \langle X, C \rangle$ can be viewed as the result of two constraint systems: a task constraint system and a resource constraint system. The task constraint system contains all the information about the temporal dependencies between the tasks to be performed, *irrespective* of the resources available. In fact, on this task level, we assume an infinite amount of resource tokens for each type distinguished. Such a task constraint system can be represented as an STN $S_t = \langle X, C_t \rangle$. For example, in our turnaround model, this STN exactly describes what has to be done and which constraints apply between the tasks to be achieved without paying attention to the actual resources available.

The resource constraint system describes the set of resources available together with the type of task that can be processed using the resources and the processing constraints that apply to these resources. This system can be easily represented as a tuple $S_r = \langle X, C_r \rangle$ where C_r describes the resource constraints that apply in performing the events X. For example, if two planes have to be fuelled, the given time constraints for fuelling might overlap. If, however, there is only one fuelling agent, a resource constraint preventing the simultaneous occurrence of two or more fuelling events would enforce a strict separation of those time constraints. Notice that often the resource constraints will encode *mutex relations* stating that one or more temporal variables cannot overlap. Such a constraint often needs to be represented as a disjunction of time intervals instead of simple intervals like in an ordinary STN[5]. The resulting STN $S = \langle X, C \rangle$ can be seen as the result of both the task and the resource constraint system and it is not difficult to see that inconsistency of S_t will imply inconsistency of S.

[5] For a detailed description of these resource constraints, see the forthcoming PhD thesis of P.C. Buzing, Faculty of EEMCS, Delft University of Technology, 2008.

With respect to plan repair, this part of the hierarchy can be used as follows: whenever S becomes inconsistent, we also check S_t. If S_t is consistent, it indicates that we might repair the plan by satisfying the resource constraints in an other way or by relaxing the resource constraints, for example by adding one or more resources. If, however, S_t is also inconsistent, this clearly indicates that we should adapt one or more tasks, for example by changing one or more temporal constraints or removing one or more tasks.

The hierarchy that results from these decompositions is depicted in Figure 2. Here the most general STN is the task-based STN S_t containing only the task constraints. This STN can be refined to the basic STN S by adding the resource constraints S_r. The basic STN S can be refined into a set of subnetworks S_i. If a disruption occurs, we first try to establish which STNs are affected. Suppose that inconsistencies are detected on the lowest (sub-STN) level. To determine the right level of repair, we first check whether or not the inconsistencies are still present at the basic STN or the task STN level. If not, we should be able to restore consistency either by repairing the individual sub-STNs or changing the agent partitioning. If, on the other hand, the basic STN is also inconsistent, but the task level STN is not, we have to change either the way resource constraints are satisfied or the resource constraints themselves. Finally, if the task level STN is inconsistent, we should change the tasks and/or their temporal relationships.

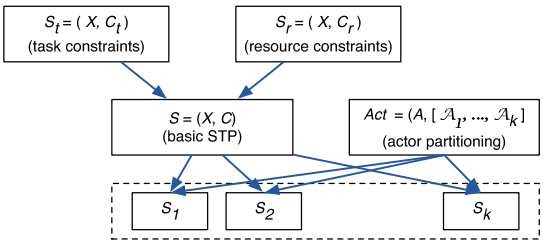


Fig. 2. Hierarchical decomposition and partitioning of an STN S into a task constraint system S_t and a resource constraint system S_r. We can use S and a set of actors A with their group decomposition to compose a set of sub-STNs S_i by temporal decoupling.

The advantage of this decomposition now should be clear: by using the actor decomposition we control the number of parties involved in the plan repair process, while the resource and task decomposition enables us to control the minimality of the repair measures themselves.

Example 2. Figure 3 shows a situation in which Aircraft X, Y and Z have to be fuelled by two fuelling operators, according to the task constraints depicted by the solid arrows. The actor partitioning is as follows: Fuelling operator 1 will fuel Aircraft X, and Fuelling operator 2 will fuel Aircraft Y and Z. The fuelling operators each own one fuelling vehicle, so only two aircraft can be fuelled simultaneously. This is expressed by the resource constraint $y_3 \rightarrow z_2$

(dashed arrow), which indicates that Aircraft Y has to be fuelled before Aircraft Z, and that it takes 5 minutes to travel from one aircraft to the other. The network is decoupled into two subnetworks, one for each fuelling operator, so the fuelling operators can each make their own schedule. Fuelling operator 1 decides to fuel Aircraft X between 13:10 and 13:20. Fuelling operator 2 will fuel Aircraft Y between 12:10 and 12:55, and Aircraft Z between 13:00 and 13:45.

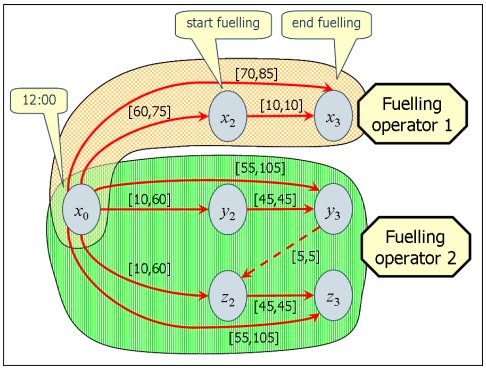


Fig. 3. Fuelling three aircraft by two fuelling operators.

Now these plans may be disrupted in several ways. We present some examples of disruptions and how they can be solved using the decomposition framework.

- *Aircraft X has a 15 min. delay.* Constraint $x_0 \rightarrow x_2$ is reduced to $[75, 75]$. Fuelling operator 1 can solve this disruption locally by right-shifting the task of fuelling Aircraft X, so the aircraft will be fuelled between 13:15 and 13:25.
- *Aircraft Y and Z both have a 20 min. delay.* Constraints $x_0 \rightarrow y_2$ and $x_0 \rightarrow z_2$ are both reduced to $[30, 60]$. The basic STN has become inconsistent (because $30 + 45 + 5 > 60$), but the task STN is still consistent - only the resource constraints have been violated. No resource constraints can be found that enable the two fuelling operators to fuel all aircraft, so additional resources have to be added, e.g., Fuelling operator 3 will fuel Aircraft Y.
- *Aircraft X has a 20 min. delay.* Fuelling Aircraft X cannot start before 13:20, so it will take at least until 13:30, whereas it has to be finished before 13:25. Thus, the task STN has become inconsistent and has to be revised.

4 Conclusions and future work

In this paper, a new approach has been presented to solve the strategic and pre-tactical planning and repair problem of ground handling services at airports. Central to this approach is a decomposition method that first of all allows us to view a temporal plan as the result of a task constraint system and a resource

constraint system. Secondly, it allows for partitioning a temporal plan into several independent subplans that can be solved locally and merged again into a conflict-free overall plan. The main advantage offered by this approach lies in its capability to repair disruptions. Here, the hierarchical decomposition allows us to select the right level of repair and the temporal decoupling allows parties to re-plan locally, making plan co-ordination and negotiation between parties largely superfluous. Given the large number of plan disruptions occurring daily at airports, this planning approach seems a valuable asset.

Currently, a prototype is being developed to implement this new approach. Its main goal is to show that decomposition offers important re-planning advantages. It aims to show that, given a disruption, less time is required and fewer actors are involved when re-planning a pre-tactical ground handling plan based on decoupling as opposed to current modes of operation. Future work will include extensive testing of the prototype against a small-scale, yet realistic scenario.

As a next step, research may focus on the level of decoupling. What happens if we partition the domain to the next level (e.g., each fuelling vehicle requiring its own decoupled plan)? Alternatively, one may group certain service providers together (e.g., cleaning and catering) for reasons of efficiency. Another topic concerns the implementation of a mechanism to upscale the level of decoupling when local re-planning is not feasible. In such a mechanism, a new decoupling is produced to group two (or more) service providers together if either one of them cannot find a re-planning solution individually. A final subject of further research is the refinement of the plan hierarchy, for example by allowing each of the parties to use its own resource constraint system and the inclusion of a task assignment system that given the resources and the tasks distinguished determines which task has to be executed using which resource.

After proving the new concept by means of a prototype, we intend to proceed towards a small-scale operational application. In the near future, such an application might be used by airlines and service providers to support their planning of ground handling activities at a real airport.

References

1. EUROCONTROL: *Launch of new collaborative decision making tool for airports,* website (2004).
2. van Leeuwen P.: *Requirements and design document for LEONARDO's collaborative decision making multi-agent system CDMMA,* Technical Report NLR-TR-2003-653, National Aerospace Laboratory NLR (2004).
3. Buzing P., ter Mors A. W., Witteveen C.: *Multi-agent plan repair with DTPs,* In Brown K. N., ed.: Proceedings of the 23rd Workshop of the UK Planning and Scheduling Special Interest Group, Cork, Ireland (2004) 3–14.
4. Wu C. L., Caves R. E.: *Modelling and optimization of aircraft turnaround time at airports,* Transportation Planning and Technology **27** (2004) 47–66.
5. Dechter R., Meiri I., Pearl J.: *Temporal constraint networks,* Artificial Intelligence **49** (1991) 61–95.
6. Hunsberger L.: *Group Decision Making and Temporal Reasoning,* PhD thesis, Harvard University, Cambridge, MA, USA (2002) Harvard Tech. Report TR-05-02.

Proceedings of the International Multiconference on
Computer Science and Information Technology, pp. 327 – 336
ISSN 1896-7094
© 2007 PIPS

Advantages of an Easy to Design Fuzzy Predictive Algorithm: Application to a Nonlinear Chemical Reactor

Piotr M. Marusak

Institute of Control and Computation Engineering, Warsaw University of Technology,
Nowowiejska 15/19, 00–665 Warsaw, Poland
P.Marusak@ia.pw.edu.pl

Abstract. Advantages of a fuzzy predictive control algorithm are discussed in the paper. The discussed fuzzy predictive algorithm is a combination of a DMC (Dynamic Matrix Control) algorithm and Takagi–Sugeno fuzzy modeling inheriting advantages of both techniques. The algorithm is numerically effective. Moreover, information about measured disturbance can be included in it in an easy way. A simple and easy to apply method of fuzzy predictive control algorithms synthesis is presented. The advantages of the fuzzy predictive control algorithm are demonstrated in the example control system of a control plant with difficult dynamics – a nonlinear chemical reactor with inverse response.

1 Introduction

Model predictive control (MPC) algorithms are widely used in practice due to the control performance they offer. They are usually used in an advanced control layer of the multilayer control system structure; see e.g. [1, 8, 10]. Basic formulations of the MPC algorithms are based on the linear control plant models. The advantage of these algorithms is that they are formulated as easy to solve, linear–quadratic optimization problems. However, application of the MPC algorithm based on a linear plant model to a nonlinear plant, if control in a wide range of set point values is needed, may bring unsatisfactory results or the results can be improved using the algorithm based on a nonlinear model.

Unfortunately, a direct usage of a nonlinear model to design the MPC algorithm leads to its formulation as a nonlinear, and in general, non–convex optimization problem that is hard to solve and computationally demanding. Thus, in practice, suboptimal algorithms that use approximate process models generated from the nonlinear model at each algorithm iteration are used; see e.g. [6, 10]. In these algorithms the nonlinear and the approximated models are used in such a way that the standard linear–quadratic optimization problem is formulated and solved at each algorithm iteration. Various kinds of nonlinear models can be used in such algorithms. However, the usage of Takagi–Sugeno (TS) fuzzy models [9] makes them especially efficient [6, 7, 10].

The DMC algorithm is practically a standard in the industrial applications; see e.g. [1, 2, 3, 5, 8, 10]. It can be relatively easily designed, because it is based on an easy to

1 This work was supported by the Polish national budget funds for science 2005-2007 as a research project.

obtain control plant model in the form of the step response. The fuzzy DMC (FDMC) algorithm used in the paper is an easy to design fuzzy predictive algorithm based on DMC technique and Takagi–Sugeno fuzzy models [6, 7, 10]. It uses a TS fuzzy model with step responses used as the local models. Such Takagi–Sugeno fuzzy models can be obtained relatively easy using the expert knowledge and/or simulation experiments.

As the FDMC algorithm is based on the nonlinear control plant model, it usually works well in a wide range of operating point changes, offering better control performance than the algorithms based on linear process models – the feature important when it operates in a hierarchical control system structure.

The discussed FDMC algorithm was applied to a nonlinear chemical plant – an isothermal CSTR in which a van de Vusse reaction takes place. The control plant is a difficult process because except it is nonlinear, it has also an inverse response. The experiments performed in two control systems: the first one – with DMC algorithm based on a linear process model and the second one – with FDMC algorithm, demonstrate the superiority of the fuzzy algorithm over the one based on the linear model.

The organization of the paper is as follows. In Sect. 2 the idea of predictive control algorithms is discussed, the DMC and easy to design FDMC algorithms used during the test are described. In Sect. 3 the control plant and algorithms used during the experiments are detailed. Sect. 4 contains description of experiments and discussion of obtained results. The paper is summarized in Sect. 5.

2. Predictive Controllers

In the MPC algorithms future behavior of the control plant is predicted many time steps ahead, using a dynamic control plant model and all available knowledge about conditions of control system operation. The future control values are then derived in such a way that the future behavior of the control system fulfills assumed criteria. Typically, the minimization of a performance function is demanded subject to the constraints put on manipulated and controlled variables; in the MPC algorithms based on input–output models usually the following optimization problem is solved at each iteration:

$$\min_{\Delta \boldsymbol{u}} \left\{ J_{MPC} = \sum_{i=1}^{p} \left(\bar{y}_k - y_{k+i|k} \right)^2 + \sum_{i=0}^{s-1} \lambda \cdot \left(\Delta \boldsymbol{u}_{k+i|k} \right)^2 \right\} \tag{1}$$

subject to: $\Delta \boldsymbol{u}_{\min} \le \Delta \boldsymbol{u} \le \Delta \boldsymbol{u}_{\max}$, $\boldsymbol{u}_{\min} \le \boldsymbol{u} \le \boldsymbol{u}_{\max}$, $\boldsymbol{y}_{\min} \le \boldsymbol{y} \le \boldsymbol{y}_{\max}$,

where $\bar{y}_k$ is a set–point value (in the classical multilayer control system structure it is obtained from the economic optimization layer), $y_{k+i|k}$ is an output value for $(k+i)^{\text{th}}$ time step predicted at k^{th} time step using control plant model, it depends on past and future control values, $\Delta u_{k+i|k}$ are future changes in the manipulated variables, $\lambda \ge 0$ is a weighting coefficient, p and s denote prediction and control

horizons, respectively; $y=\left[y_{k+1|k},\ldots,y_{k+p|k}\right]^T$, $\boldsymbol{\Delta u}=\left[\Delta u_{k|k},\ldots,\Delta u_{k+s-1|k}\right]^T$, $u=\left[u_{k|k},\ldots,u_{k+s-1|k}\right]^T$, $\boldsymbol{\Delta u}_{min}$, $\boldsymbol{\Delta u}_{max}$, $\boldsymbol{u}_{min}$, $\boldsymbol{u}_{max}$, $\boldsymbol{y}_{min}$, $\boldsymbol{y}_{max}$ are vectors of lower and upper bounds of changes and values of the control signals and of the output variable values, respectively. As a solution to the optimization problem (1) the optimal vector of changes of the manipulated variables is obtained. From this vector, the $\Delta u_{k|k}$ element is taken and applied in the control system. Then optimization is repeated at the next time instant.

The way the predicted output values $y_{k+i|k}$ are derived depends on the dynamic control plant model exploited by the algorithm. If the linear model is used then the optimization problem (1) is a standard linear–quadratic programming problem. The conventional (non–fuzzy) DMC algorithm is based on a process model in the form of the control plant step responses.

2.1. Standard DMC Algorithm

The classical (non–fuzzy) DMC algorithm uses the control plant model in the form of its step response; see e.g. [2, 3, 5, 8, 10]:

$$\tilde{y}_k = \sum_{i=1}^{p_d-1} a_i \cdot \Delta u_{k-i} + a_{p_d} \cdot u_{k-p_d}, \tag{2}$$

where $\tilde{y}_k$ is the output of the control plant model at the k^{th} time instant, Δu_k is a change of the manipulated variable at the k^{th} time instant, a_i (i=1,…,p_d) are step response coefficients of the control plant, p_d is equal to the number of time instants after which the coefficients of the step response can be assumed as settled, u_{k-p_d} is a value of the manipulated variable at the $(k–p_d)^{\text{th}}$ time instant.

The predicted output values are then calculated using the following formula:

$$y_{k+i|k}=\sum_{n=1}^{i} a_n\cdot\Delta u_{k-n+i}+\sum_{n=i+1}^{p_d-1} a_n\cdot\Delta u_{k-n+i}+a_{p_d}\cdot u_{k-p_d+i}+d_k, \tag{3}$$

where $d_k=y_k-\tilde{y}_{k-1}$ is assumed to be the same at each time instant in the prediction horizon (it is a DMC–type disturbance model). Thus (3) can be transformed into the following form:

$$\begin{aligned} y_{k+i|k}=y_k+&\sum_{n=i+1}^{p_d-1} a_n\cdot\Delta u_{k-n+i}+a_{p_d}\sum_{n=p_d}^{p_d+i-1}\Delta u_{k-n+i}\\ &-\sum_{n=1}^{p_d-1} a_n\Delta u_{k-n}+\sum_{n=1}^{i} a_n\Delta u_{k-n+i|k}. \end{aligned} \tag{4}$$

In (4) only the last component depends on future manipulated variable changes. Thus the vector of predicted output values $\boldsymbol{y}$ can be decomposed into the following components:

$$\boldsymbol{y} = \boldsymbol{y}^{fr} + \boldsymbol{A} \cdot \Delta \boldsymbol{u}, \tag{5}$$

where $\boldsymbol{y}^{fr} = \left[y^{fr}_{k+1|k}, \ldots, y^{fr}_{k+p|k} \right]^{\mathrm{T}}$, is a vector of length p and is called a free response of the plant, because it contains future output values calculated assuming that the control signal does not change in the prediction horizon (describes influence of the manipulated variable values applied to the control plant in previous iterations):

$$\boldsymbol{y}^{fr} = \boldsymbol{y}^{l} + \boldsymbol{A}_p \cdot \Delta \boldsymbol{u}_p, \tag{6}$$

where $\Delta u_p = \left[\Delta u_{k-1}, \ldots, \Delta u_{k-p_d+1} \right]^T$, $y^l = \left[y_k, \ldots, y_k \right]^T$ and

$$\boldsymbol{A}_p = \begin{bmatrix} a_2 - a_1 & a_3 - a_2 & \cdots & a_{p_d-1} - a_{p_d-2} & a_{p_d} - a_{p_d-1} \\ a_3 - a_1 & a_4 - a_2 & \cdots & a_{p_d} - a_{p_d-2} & a_{p_d} - a_{p_d-1} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ a_{p+1} - a_1 & a_{p+2} - a_2 & \cdots & a_{p_d} - a_{p_d-2} & a_{p_d} - a_{p_d-1} \end{bmatrix}. \tag{7}$$

$\boldsymbol{A}$ is a matrix of dimensionality $p \times s$, called the dynamic matrix and is composed of the control plant step response coefficients:

$$\boldsymbol{A} = \begin{bmatrix} a_1 & 0 & \cdots & 0 & 0 \\ a_2 & a_1 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ a_p & a_{p-1} & \cdots & a_{p-s+2} & a_{p-s+1} \end{bmatrix}. \tag{8}$$

Using the prediction (5) one can formulate the optimization problem (1), solution of which contains the future changes in the manipulated variables. The details concerning formulation of the DMC algorithm can be found, e.g. in [2, 3, 5, 8, 10].

Disturbance measurement can be taken into consideration in an easy way. Despite that, it can offer significant control system performance improvement. In order to take the disturbance measurement into consideration in the DMC algorithm the control plant model (2) should be supplemented by terms describing influence of the disturbance on the output variable:

$$\tilde{y}_k = \sum_{i=1}^{p_d-1} a_i \cdot \Delta u_{k-i} + a_{p_d} \cdot u_{k-p_d} + \sum_{i=1}^{p_z-1} b_i \cdot \Delta z_{k-i} + b_{p_z} \cdot z_{k-p_z}, \tag{9}$$

where Δz_k is a change of the manipulated variable at the k^{th} time instant, b_i (i=1,…, p_z) are disturbance step response coefficients, p_z is equal to the number of time instants after which the coefficients of the disturbance step response can be assumed as settled, z_{k-p_d} is a value of the manipulated variable at the $(k–p_z)^{\text{th}}$ time instant.

If there is no information available about future disturbance, the future changes of disturbance are assumed equal to 0. The influence of past changes of the disturbance should be included in the free response. Thus, this time, the free response is described by the following formula:

$$\boldsymbol{y}^{fr} = \boldsymbol{y}^{l}+\boldsymbol{A}_p\cdot\Delta\boldsymbol{u}_p+\boldsymbol{A}_z\cdot\Delta\boldsymbol{z}_p, \tag{10}$$

where $\Delta z_p=\left[\Delta z_{k-1},\ldots,\Delta z_{k-p_z+1}\right]^T$ and

$$\boldsymbol{A}_z=\begin{bmatrix} a_2-a_1 & a_3-a_2 & \cdots & a_{p_z-1}-a_{p_z-2} & a_{p_z}-a_{p_z-1} \\ a_3-a_1 & a_4-a_2 & \cdots & a_{p_z}-a_{p_z-2} & a_{p_z}-a_{p_z-1} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ a_{p+1}-a_1 & a_{p+2}-a_2 & \cdots & a_{p_z}-a_{p_z-2} & a_{p_z}-a_{p_z-1} \end{bmatrix}. \tag{11}$$

2.2. Easy to Design Fuzzy DMC Algorithm

The FDMC algorithm used in the paper exploits TS models with local models in the form of step responses. Thanks to such an approach each iteration of the algorithm is simple. Moreover, the local models are easy to obtain. The FDMC algorithm is thus based on TS fuzzy models described by the set of the following rules:

Rule j: if

$$\underbrace{y_k \text{ is } B_1^j \text{ and } \ldots \text{ and } y_{k-n_P+1} \text{ is } B_{n_P}^j \text{ and } u_k \text{ is } C_1^j \text{ and } \ldots \text{ and } u_{k-m_P+1} \text{ is } C_{m_P}^j}_{\text{antecedent}} \tag{12}$$

$$\text{then } \underbrace{\tilde{y}_{k+1}^j=\sum_{i=1}^{p_d-1} a_i^j\cdot\Delta u_{k-i}+a_{p_d}^j u_{k-p_d}}_{\text{consequent}},$$

where y_k is an output variable value of the control plant model at the k^{th} time instant, u_k is a manipulated variable value at the k^{th} time instant, $B_1^j,\ldots, B_{n_P}^j, C_1^j,\ldots, C_{m_P}^j$ are fuzzy sets, a_i^j are the coefficients of step responses in j^{th} local model, j=1,…,l, l – number of rules.

If measured disturbance is taken into consideration, the local model has the following form:

$$\tilde{y}^j_{k+1} = \sum_{i=1}^{p_d-1} a^j_i \cdot \Delta u_{k-i} + a^j_{p_d} \cdot u_{k-p_d} + \sum_{i=1}^{p_z-1} b^j_i \cdot \Delta z_{k-i} + b^j_{p_z} \cdot z_{k-p_z}, \qquad \textbf{(13)}$$

where b^j_i are the coefficients of step responses in j^{th} local model.

The design process of the TS model (12) can be simplified to a large extent. It is because instead of obtaining a complicated local models (as e.g. difference equations) it is sufficient to obtain a few step responses of the control plant (from the environs of a few operating points). The membership functions can be chosen using expert knowledge, simulation experiments, fuzzy neural networks or all these techniques combined.

In the basic FDMC algorithm, the TS model (12) is used, at each iteration, in the following way:

1. A linear model for current time instant is derived using current values of process variables, the TS fuzzy model (12) and fuzzy reasoning. The output value of the model is calculated using the following formula (in the case when a measured disturbance is taken into consideration):

$$\tilde{y}_k = \sum_{i=1}^{p_d-1} \tilde{a}_i \cdot \Delta u_{k-i} + \tilde{a}_{p_d} \cdot u_{k-p_d} + \sum_{i=1}^{p_z-1} \tilde{b}_i \cdot \Delta z_{k-i} + \tilde{b}_{p_z} \cdot z_{k-p_z}. \qquad \textbf{(14)}$$

where $\tilde{a}_i = \sum_{j=1}^{l} \tilde{w}_j \cdot a^j_i$, $\tilde{b}_i = \sum_{j=1}^{l} \tilde{w}_j \cdot b^j_i$ and $\tilde{w}_i$ are the normalized weights calculated using standard fuzzy reasoning, see e.g. [9].

In fact (14) is the step response control plant model valid for the current values of process variables. Thus, next steps of the FDMC algorithm are the same as in the conventional DMC algorithm (Sect. 2.1), i.e.

2. The obtained step response coefficients are used to generate the dynamic matrix.
3. The step response model (14) is used to generate the plant free response.
4. The free response and the dynamic matrix are used to formulate the quadratic optimization problem.
5. The optimization problem is solved and, using the obtained solution, the manipulated variable value is generated. Then the controller passes to the next iteration.

3. Control systems

The considered control plant is an isothermal CSTR in which a van de Vusse reaction carries out (Fig. 1 a) [4]. Its steady–state characteristics are shown in Fig. 1 b.

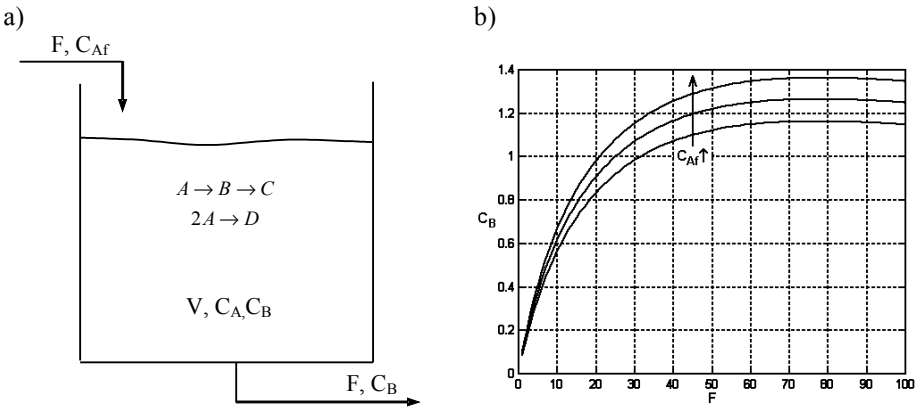


Fig. 1. Isothermal CSTR with van de Vusse reaction; a) diagram of the plant, b) steady–state characteristics of the plant

The reaction scheme is as follows

$$\begin{aligned} A &\longrightarrow B \longrightarrow C, \\ 2A &\longrightarrow D. \end{aligned} \tag{15}$$

The process model contains composition balance equations for both components

$$\begin{aligned} \frac{dC_A}{dt} &= -k_1 C_A - k_3 C_A^2 + \frac{F}{V}(C_{Af} - C_A), \\ \frac{dC_B}{dt} &= k_1 C_A - k_2 C_B - \frac{F}{V} C_B. \end{aligned} \tag{16}$$

where C_A, C_B are the concentrations of components A and B, respectively, F is the inlet flow rate (equal to the outlet flow rate), V is the volume in which the reaction is carried out (it is assumed constant and V =1 l), C_{Af} is the concentration of component A in the inlet flow stream (if it is not stated differently it is assumed that C_{Af0} =10 mol/l). The values of the kinetic parameters are: k_1 =50 1/h, k_2 =100 1/h, k_3 =10 l/h · mol.

The output variable is the concentration C_B of substance B, the manipulated variable is the inlet flow rate F of the raw substance, the disturbance variable is the concentration of component A in the inlet flow stream C_{Af}.

It was assumed that the manipulated variable is constrained

$$F_{\min} \le F \le F_{\max} \tag{17}$$

where $F_{\min}$ =0 l/h, $F_{\max}$ =60 l/h.

For the presented control plant two controllers were designed: the first one – a DMC controller based on a linear model and the second one – an FDMC controller based on a Takagi–Sugeno fuzzy model. The sampling period of both controllers was assumed equal to 3.6 s. The following tuning parameters of both algorithms were assumed: p =70, s =35, λ =0.001.

The DMC algorithm is based on a step response taken in environs of the operating point: C_{B0} =1.12 mol/l, C_{A0} = 3 mol/l, F=34.3 l/h [4]. In the case of an FDMC algorithm, two additional step responses are used (the conventional DMC algorithm was in fact extended): the first one from environs of the point C_{B0} =0.91 mol/l, C_{A0} = 2.18 mol/l, F =20 l/h, and the second one – from environs of the point C_{B0}=1.22 mol/l, C_{A0}=3.66 mol/l, F =50 l/h. Thus, the TS control plant model used by the FDMC controller is composed of three rules. The assumed membership functions are shown in Fig. 2.

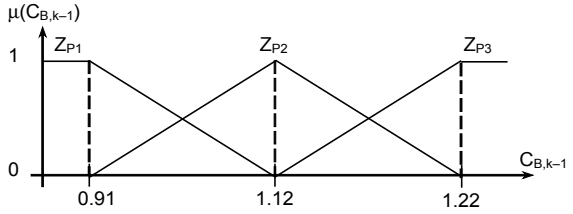


Fig. 2. Membership functions of the fuzzy DMC controller

4. Simulation Experiments

During the experiments the operation of both control systems (the first one – with DMC and the second one – with FDMC algorithms) was compared. First, the set–point value $\bar{C}_B$ was changed towards high values of the concentration C_B, to 1.22 mol/l, precisely. The response obtained in the control system with the FDMC algorithm (solid lines in Fig. 3) is better than the one acquired by the conventional DMC algorithm (dashed lines in Fig. 3) – the output variable much faster reaches the demanded set–point value.

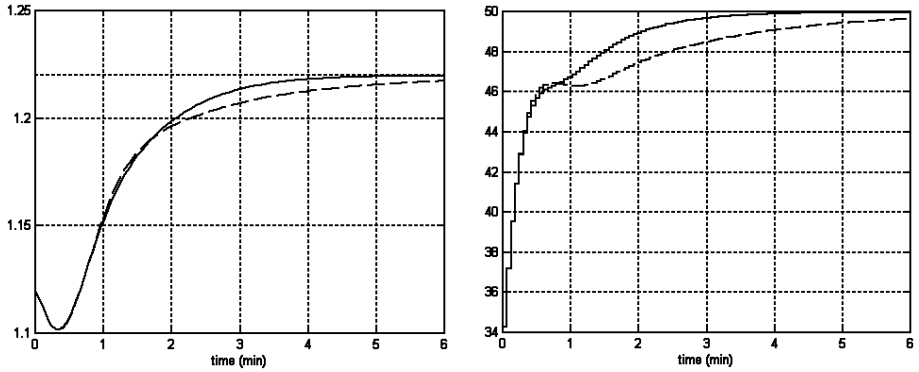


Fig. 3. Responses of the control system with: DMC algorithm (dashed line), FDMC algorithm (solid line) to the change of C_B set–point to $\bar{C}_B = 1.22$; left – output variable C_B, right – manipulated variable F

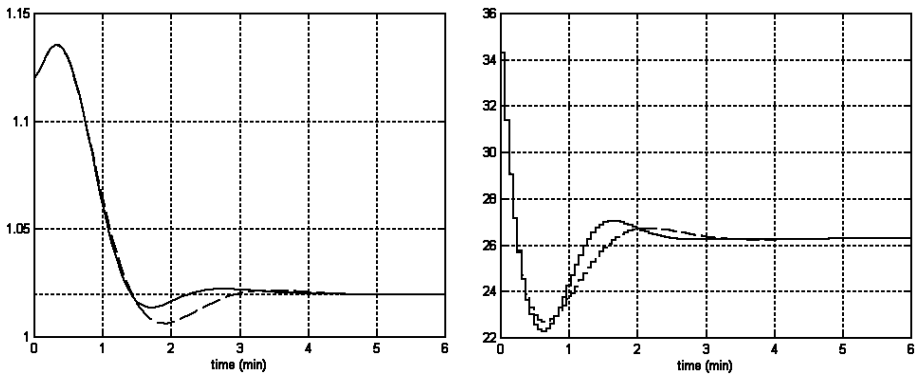


Fig. 4. Responses of the control system with: DMC algorithm (dashed line), FDMC algorithm (solid line) to the change of C_B set–point to $\bar{C}_B = 1.02$; left – output variable C_B, right – manipulated variable F

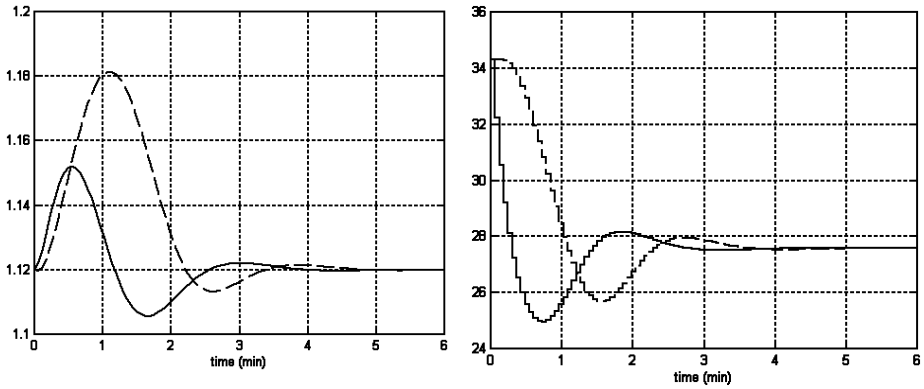


Fig. 5. Responses of the control system with FDMC algorithm to the increase of disturbance C_{Af} by 1 mol/l; disturbance measurement mechanism: not used (dashed line), used (solid line) in the algorithm; left – output variable C_B, right – manipulated variable F

It can be also seen that both responses at the beginning of control action (for C_B values close to the 1.12 mol/l) are practically the same. It is caused by the fact that the responses, the conventional DMC controller is based on, were used as a local model in one of the rules of the TS model exploited by the FDMC controller. This fact illustrates well that the FDMC controller can be designed by simply extending the existing DMC controller (usually well tuned to work near the operating point it was designed for) if needed.

In the second experiment, the set–point value $\bar{C}_B$ was changed towards low values of the concentration C_B, i.e. to 1.02 mol/l. As in the previous experiment, the response obtained in the control system with FDMC algorithm (solid lines in Fig. 4) is better than the one generated by the control system with the conventional DMC algorithm (dashed lines in Fig. 4) – the output variable also reaches the set–point

value faster in the case of the control system with the fuzzy algorithm. Moreover, the overshoot is significantly lower in the control system with the FDMC algorithm. The obtained results clearly show the profits gained thanks to the usage of the fuzzy approach.

There was also performed an experiment with the change of disturbance C_{Af} – concentration of component A in the inlet stream. In the case when there was a disturbance measurement used (solid lines in Fig. 5) the obtained settling time is smaller than in the case without disturbance measurement (dashed lines in Fig. 5). Moreover, in the first case there is much smaller overshoot. Similar results were obtained in the case of operation near other operating points. In such cases the further the original operating point the system operates, the bigger difference of operation between DMC and FDMC algorithm is, with advantage of the latter one.

5. Summary

Application of the FDMC algorithm to the example control plant brought improvement of the control system operation comparing to the case when the algorithm based on the linear model was used. It was achieved despite relatively little effort needed to design a fuzzy algorithm. It was possible because model the FDMC algorithm is based on (a Takagi–Sugeno fuzzy model with step responses used as local models) can be obtained in an easy way. The mechanism of taking measured disturbance into consideration in the algorithm (also relatively simple to apply), brought further improvement of algorithm performance.

References

1. Blevins T. L., McMillan G. K., Wojsznis W. K., Brown M. W.: *Advanced Control Unleashed,* ISA (2003).
2. Camacho E. F., Bordons C.: *Model Predictive Control,* Springer–Verlag (1999).
3. Cutler C. R., Ramaker B. L.: *Dynamic Matrix Control – a computer control algorithm,* Proc. Joint Automatic Control Conference, San Francisco, CA, USA (1979).
4. Doyle F., Ogunnaike B. A., Pearson R. K.: *Nonlinear model-based control using second-order Volterra models,* Automatica, 31 (1995), 697–714.
5. Maciejowski J. M.: *Predictive control with constraints,* Prentice Hall, Harlow (2002).
6. Marusak P.: *Predictive control of nonlinear plants using dynamic matrix and fuzzy modeling* (in Polish). PhD Thesis, Warsaw, Poland (2002).
7. Marusak P., Tatjewski P.: *Fuzzy Dynamic Matrix Control algorithms for nonlinear plants,* Proc. 6th International Conference MMAR 2000, Miedzyzdroje, Poland (2000) 749–754.
8. Rossiter J. A.: *Model-Based Predictive Control,* CRC Press, Boca Raton (2003).
9 . Takagi T., Sugeno M.: *Fuzzy identification of systems and its application to modeling and control,* IEEE Trans. on Systems, Man and Cybernetics 15 (1985) 116–132.
10. Tatjewski P.: *Advanced Control of Industrial Processes; Structures and Algorithms,* Springer–Verlag, London (2007).

Proceedings of the International Multiconference on
Computer Science and Information Technology, pp. 337 – 346
ISSN 1896-7094
© 2007 PIPS

Preferences in Evolutionary Multi-Objective Optimisation with Noisy Fitness Functions: Hardware in the Loop Study

Piotr Woźniak

Technical University of Lodz, Control Theory Group, 18/22 Stefanowskiego St.,
90-924 Lodz, Poland
pwozniak@p.lodz.pl

Abstract. Multi-objective optimisation (MOO) is an important class of problem in engineering. The conflict of objectives in MOO places the issue of compromise in a central position. Since no single solution optimises all objectives, decision-making based on human preference is a part in solving MOO problems. In this paper application of the evolutionary MOO to the dynamic system controller design by use of the hardware in the loop is presented. Thanks to this approach problems of un-modelled plant dynamics and uncertainty of parameters are alleviated because no mathematical model is needed. The a-priori search of one solution does not require knowledge of a whole Pareto front.

1 Introduction

Real-world optimisation problems are characterised by the existence of multiple, often conflicting, objectives. In the absence of information about the relative importance of the various criteria, such problems typically admit multiple optimal solutions where any improvement in one objective can only be obtained at the expense of degradation in other objectives. Edgeworth and Pareto captured this notion mathematically in the criterion widely known as Pareto optimality.

The general MOO problem may be stated as follows. Without loss of generality, the minimisation of objectives is considered in the discussion throughout this paper.

Definition 1

Find the vector of decision variables $X = \{x_1, x_2, \ldots, x_n\}$ which satisfies the m inequality constraints:

$$g_i(X) \geq 0,\ i = 1,2,\ldots,m$$

p equality constraints:

$$h_i(X) \geq 0,\ i = 1,2,\ldots,p$$

and minimises the k objective functions:

$$f(X) = \left[f_1(X) \quad f_2(X) \quad \ldots \quad f_k(X) \right] .$$

In general the objectives are non-commensurable and in conflict with each other.

Definition 2. A vector of decision variables X_j dominates X_i in the Pareto-optimality sense if and only if X_j performs better in at least one objective and at least as good as X_i in the rest

$$X_i < X_j \Leftrightarrow f_n^j < f_n^i \text{ for } 1 \le n \le k \text{ and } f_m^j \le f_m^i \text{ for } \forall m \in [1,k], s.t.\ m \ne n .$$

If neither of the two candidate solutions dominates the other, they are non-dominated. The Pareto-optimal set of a MOO problem consists of solutions which dominate other solutions in the feasible region. The image of this set in the objective space is the Pareto-front.

The immediate effect, when applying the criterion of Pareto optimality to high-dimensional problems, is that a large number of solutions within any generation are non-dominated and, therefore, qualitatively indistinguishable from each other. It is empirically evidenced in [7, pp. 404–405]. Therefore, when the number of objectives is large, Pareto dominance-based ranking procedures become ineffective in sorting out the quality of solutions and selection operators play a minor role to compensate for this effect, which translates into a lack of selective pressure.

The dimensionality reduction techniques may be applied to alleviate this issues. This approach takes advantage of possibility that some objectives may behave in a non-conflicting manner near the Pareto-optimal region although a conflict exists elsewhere. In such cases, the Pareto-optimal front will be of dimension lower than the number of objectives. This technique, has been recently successfully applied to the EMOO electric synchronous machine design [1, 2].

Another possibility, typical for engineering approach to the MOO, is by search for set of solutions in a region of interest to the DM without approximating points on the complete high-dimensional Pareto-optimal front. Thus, the final solution of the Evolutionary MMO (EMOO) problem results, not only from the search process, but also from the decision process. As such, this paper provides a basis for online solutions selection with hardware in the loop (HiL) setup with Permanent Magnet Synchronous Machine (PMSM).

The DM preferences in the EMOO are discussed in the next Section. In Section 3 the robustness issues of the EMOO of the HiL setup. Design problem is introduced in Section 4 along the hardware overview. Results of the research are presented in Section 5.

Section 6 concludes the presentation.

2 Issues in the EMOO algorithms

Evolutionary algorithms are global parallel search and optimisation methods based around Darwinian principles, working on a population of potential solutions to a problem. Every individual in the population represents a particular solution to the EMOO problem. The population is evolved over a series of generations (iterations of the algorithm) to produce better solutions. The quality of the solution is evaluated by the fitness assignment i.e. functions which characterises the individual's performance in the problem domain. Most of published results on EMOO algorithms base on explicit objective functions. In this paper we consider the approach when each fitness evaluation is run on the hardware (as introduced in Section 4).

The relative degree of the fitness value determines the level of propagation of the individual's genes to the next iteration. In the EMOO algorithm NSGA-II [7] in use here, the rank of a certain individual corresponds to the number of individuals in the current population by which it is dominated. All non-dominated individuals are assigned rank zero, while dominated ones are penalised according to the population density of the corresponding region of the trade-off surface.

2.1 Preferences in the EMOO

Solutions on the Pareto-optimal set perform better in some objectives and worse in other. Since no single solution optimises all objectives, Decision-Making (DM) based on subjective human preference is a natural aspect in solving EMOO problems. In real problems only a set of solutions in a region of interest to the DM is required. One may consider introducing some component to overall fitness measure for each solution. This preference related fitness component zeroes on a subset of the Pareto-optimal solutions considered of high relevance to the DM for the particular EMOO problem. The vagueness and context-dependence of the DM's preference structure result, unsurprisingly, in various mathematical models as well as techniques proposed for the incorporation of preferences into EMOO [3-5, 8, 10].

The integration of the DM preferences in the EMOO process must allow the selection. Two different classes of methods can be defined, taking into account the way how a search and decision processes are integrated with EMOO [3]:

i) A-priori methods: The DM preferences are defined in terms of an aggregating function which combines objectives values into a single utility value.

ii) A-posteriori methods: The approximation of the Pareto optimal set is supplied to the DM, which selects the most preferred among the alternatives using e.g. Weighted Metrics approach [6].

In this paper we consider the a-priori approach based on Reference Point Method [8]. We consider it as most suitable in the problem under investigation although for this methodology it is necessary to know beforehand the criteria space, and also the DM needs to have some knowledge about the location of regions of the best solutions. No need for the whole Pareto front approximation is the main advantage over all a-posteriori approaches.

2.2 Reference Point Method

In the Reference Point Method the aim is to reach a Pareto front region located near a specific pre-defined reference point or points. This method can be applied using or not a weight vector. The main ideas of this method are the following:

i) Solutions closer to the reference points, in the objective space, are to be more preferred,

ii) Solutions within a ε-neighbourhood to a near-reference point solution are to be preferred, in order to maintain a diverse set of solutions near each reference point.

To allow the identification of the solutions located near the reference point the normalised Euclidian distance (d_{ij}) between each solution of the best non-dominance level and the reference point is calculated

$$d_{ij} = \sqrt{\sum_{i=1}^{k} w_i \left(\frac{f_i(X) - \bar{z}_i}{f_i^{max} - f_i^{min}} \right)^2} \qquad (1)$$

where : f_i^{max} and f_i^{min} are the maximum and the minimum value of the objective function for criterion *i*, and $\bar{z}_i$ and w_i are the *i*-th component of the reference point and weight vector, respectively.

The distance (1) takes into account simultaneously the relative importance of the different criteria, quantified through the weight vector, and the distance between the solution and the reference point. The solutions with lower distance to the reference point and, simultaneously, with high weight vector, will be selected preferentially.

This procedure was been implemented [9] through the modification of the NSGA-II algorithm [5] niching strategy:

Step 1: For each reference point, the distance (1) of each solution of the front is calculated and the solutions are sorted in ascending order of distance. This way, the solution closest to the reference point is assigned a rank of one,

Step 2: After computations in Step 1 are performed for all reference points, the solutions with a smaller preference distance are preferred in the tournament selection,

Step 3: To control the extent of obtained solutions, an ε - neighbourhood idea is used in the niching operator. First, a random solution is picked from the non-dominated set. Thereafter, all solutions having a sum of normalised difference in objective values of ε or less from the chosen solution are assigned an artificial large preference distance. This way, only one solution within an ε-neighbourhood is emphasised. Then, another solution from the non-dominated set is picked and the above procedure is performed. The use of the ε-based selection strategy ensures a spread of solutions near the preferred Pareto-optimal regions.

The main change in NSGA-II was made on the crowding operator, since in this case the aim is not to obtain a great diversity of solutions along the Pareto frontier but only a subset of these solutions that minimise the distance to the reference point.

3 Robustness issues in the EMOO based control system design

The performance of a particular control design is fundamentally tied to the accuracy of the model upon which it is based. This is especially true for iterative control design and optimisation procedures. The substitution of Hhardware in the loop (HiL) for the software model in the EMOO opens up new possibilities for design based on real world performance indices.

In real optimisation problems, a wide range of uncertainties have to be taken into account. Generally, uncertainties in EMOO can be categorised into four classes. Each

of them is presented with additional comments concerning the HiL setup to be presented in Section 4.2.

i. Noise; in the HiL it comes from measurement errors from sensors (e.g. the Sensors Block from the system presented on Fig. 2),
ii. Errors in fitness approximations based on mathematical modelling; introducing the HiL approach we eliminate this type of uncertainty,
iii. Time-varying parameters of fitness; in the considered HiL based control system design changes in environment may be neglected because in this study evaluations are available in less than 1 [s] that is in time shorter than all time constants in the system [11],
iv. Perturbations of parameters and environmental parameters after the optimal solution has been determined; this type is not considered in this paper, however it may be easily included in the HiL by the appropriate design of experiments sampling approaches.

In general noisy or uncertain scenario the proximity of other fitness values, even if only close on one objective, can influence how the rank is assigned. Limits on objectives, constraints on the chromosomes, and sharing can all be implemented easily within this ranking framework, allowing interactive DM with any noisy systems possible.

At the current stage of research no disturbance was introduced.

4 Hardware in the Loop in the EMOO of the Permanent Magnet Synchronous Motor speed control design

The structure of a PI speed controller considered it this paper is given by

$$u(t) = k_p e(t) + \frac{1}{T_i} \int_0^t e(\tau) d\tau \qquad (2)$$

where u(t) is the output of the controller at time t,
e(t) is the error between the actual and the desired speed of the motor shaft,
k_p, T_i are constants of the controller, and decision parameters of the EMOO problem.

Depending on the choice of k_p, T_i the response of a controller to a given error signal will vary significantly.

Various methods exist to tune the gains of a PI controller off-line to attain the prescribed transient response and steady-state error criteria [12] because there is no analytical way of finding the optimal set of constants (k_p, T_i). Empirical methods such as Ziegler-Nichols tuning can be used. Up-to-date search methods generally involve some form of iterative approach to achieve performance criteria such as rise-time, overshoot and settling-time [12]. For that reason it was decided to use a EMOO approach to find several possible parameter settings that would result in good performance.

4.1 Objective functions

When a controller is designed for a specific system it is often desired that multiple design requirements are met. The drive system is no exception. When evaluating the performance of the PI controller parameters for the speed control loop a step response is considered and different performance measures are investigated [11].

The fitness functions to be minimised are of the form:

i) the speed tracking error denoted as ITAE [12]

$$\text{ITAE} = \int \left| e_p(t) \right| t\,dt \tag{3}$$

ii) composite objective based on the time period when output current of the converter is on constraints. This objective (denoted hereafter as I_{constr}) is related to the power consumption of the system [11].

$$\text{Int}(i) = \int \left| i(t) \right| dt \tag{4}$$

In this way, it is required the search be performed basing on two non-commensurable objectives. They may be visualised in form of fitness landscapes over constrained parameters' space as presented on Fig.1. The values for the sample visualisation are obtained from a sequence of step response tests for $0 \leq t \leq 1$ [s].

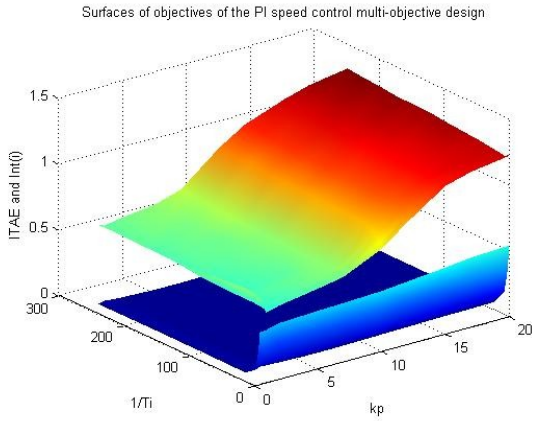


Fig. 1. Fitness landscape for objective functions (3) and (4).

4.2 Hardware overview

In the application under consideration here, the software implemented PI controller to be optimised (Fig. 2) performs tracking of a speed demand for the PMSM. It is achieved by calcultion of a current demand for Power Module. Its value depends on the shaft speed (from the Speed Sensor), and the Load (from the Manometer).

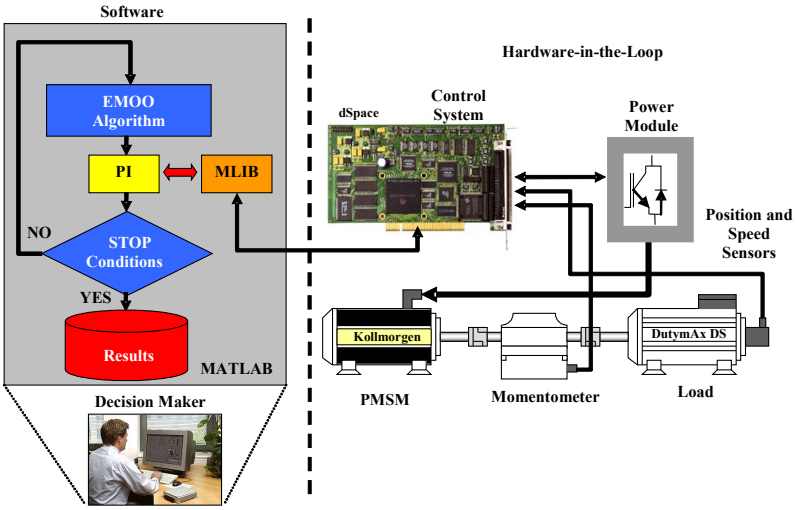


Fig. 2. On-line PI speed controller EMOO design with the hardware in the loop

The actual view of the experimental installation is presented on Fig. 3.

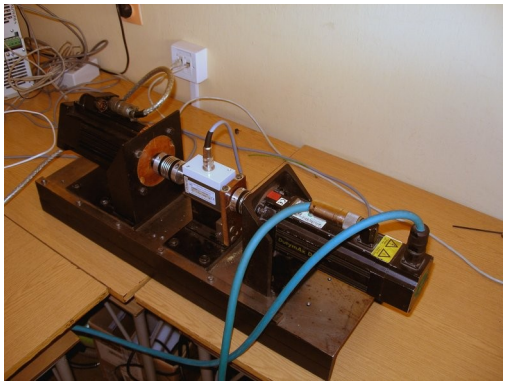


Fig. 3. Actual view of the experimental installation.

The measurement of fitness functions were realised in each iteration of the EMOO as a set of step response tests. There is one test for each member of population under consideration. Thanks to the symmetry in the PMSM operation it is possible to alter the sign of successive tests. It enables cutting down the time of experiments.

Sample sequence of six, 0,8 [s] long tests is presented on Fig. 4.

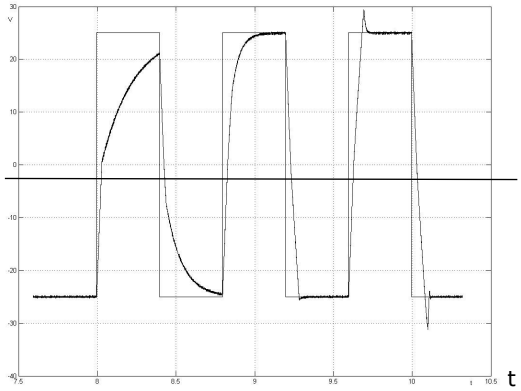


Fig. 4. Registered sequence of six 0,8 [s] long, speed step-response tests with altering sign for $v_{desired}$ = 25 [rad/s]

4.3 EMOO setup

The parameters of the optimisation algorithm NSGA-II implemented during EMOO of the problem defined by (2)-(4) under constraints:

i) in the parameters space defined by inequalities $0{,}25 < k_p < 20$; $25 < 1/T_i < 200$,

ii) in the H-i-L installation (not presented; for details see [11]),

are presented in Table 1.

Table 1 . Basic parameters of the NSGA-II algorithm

Algorithm parameter		Value
Population	*N*	60
Generations	*G*	50
Pool size	*N*/2	30
Tour size		2
Crossover probability		0.9
Mutation probability		0.5

5 Results

The result of measurement's noise influence on the EMOO speed control design is presented after normalisation to [0,1] on Fig. 5 in form of a Pareto set. This approximation obtained using NSGA-II [4] without DM preferences modifications (as discussed in Section 2.2) presents significant influence of noise which prevents even spread of non-dominated solutions along the front. To alleviate this effect further research on relation between the bounded noise and quality of the whole Pareto set approximation is needed.

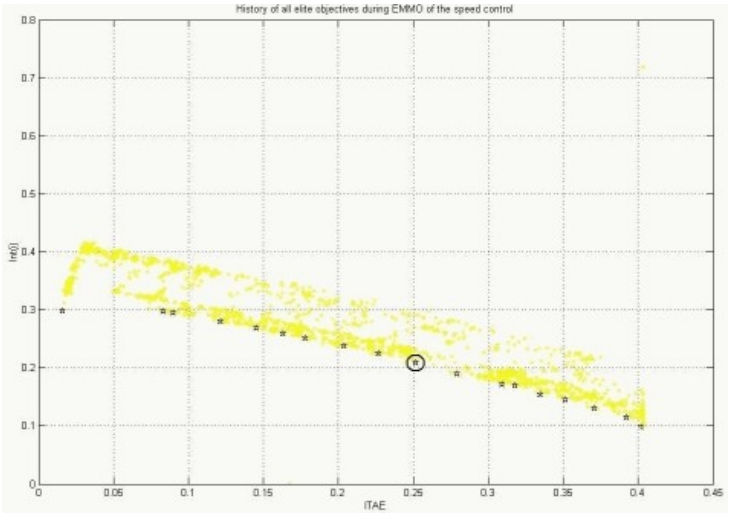


Fig. 4. Single solution selected from the Pareto front approximation (stars) with all elite solutions from 100 iterations .

There are several techniques for selection of one exact, or with some acceptable tolerance, Pareto-optimal solution. For this study the Reference Point Method modified NSGA-II algorithm is used to select single Pareto-optimal solution k_p=2.01, $1/T_i$=29.3 [1/s]. The step response for final values gives the speed step response presented in Fig.5.

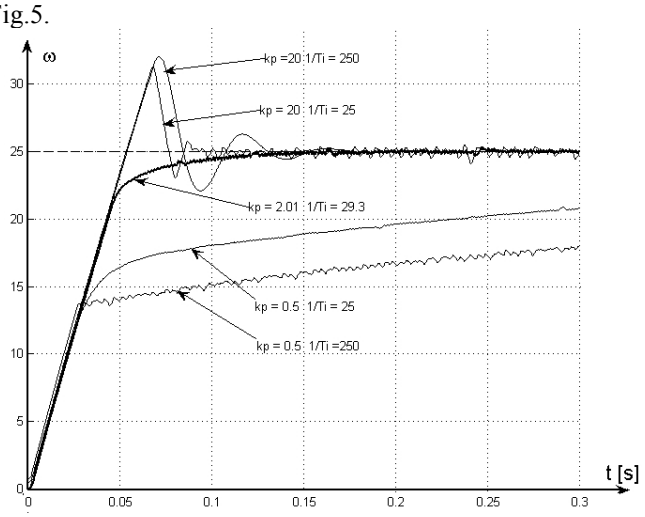


Fig. 5. Comparison of speed step response for the selected single Pareto-optimal solution (bold line) and four extreme decision pairs.

Further research is needed to evaluate influence of the measurement noise on the radius of solution set under DM preferences.

6 Conclusions

This paper investigates the design of a dynamic controller for the drive system directly onto hardware. A multi-objective evolutionary algorithm is applied to the task of controller development, while an objective function ranks the system response to find the Pareto-optimal set of controllers. The measurement noise present during each evaluation at run-time deteriorate the quality of Pareto set approximation. To prevent decline of results the Decision Maker preferences are included in the EMOO process in form of the a-priori type Reference Point Method.

This approach demonstrates very good properties providing the designer the set of optimisation parameters in the area surrounding the reference point.

References

1. Woźniak P., Witczak P.: *Dimensionality Reduction in Evolutionary Multiobjective Design of Permanent Magnet Generator,* In: Proc.of Fourth Int. Conference on Evolutionary Multi-Criterion Optimization EMO 2007 (2007) 63-68.
2 Woźniak P.: *Dimensionality Reduction in Evolutionary Multiobjective Design: Case Study,* In: Proc.of ACM-SIGEVO on Genetic and Evolutionary Computation GECCO'07 (2007) 913.
3 Coello C.A.: *Handling Preferences in Evolutionary Multiobjective Optimization: A Survey,* in Proc. Congr. Evolutionary Computation (2000) 30–37.
4 Cvetkovic D., Parmee I. C.: *Preferences and Their Application in Evolutionary Multiobjective Optimization,* IEEE Trans. on Evol. Comput., 6, 1 (2002) 42-57.
5 Branke J., Deb K.: *Integrating User Preferences into Evolutionary Multi-Objective Optimization,* in Yaochu Jin (editor), Knowledge Incorporation in Evolutionary Computation, Springer (2005) 461-477.
6 Miettinen K. M.: *Nonlinear Multiobjective Optimization,* Kluwer, Boston (1999).
7 Deb K.: *Multi-Objective Optimization using Evolutionary Algorithms,* Wiley, Chichester (2001).
8 Wierzbicki A. P.: *The use of reference objectives in multiobjective optimization,* In Fandel, G., Gal, T. Eds, Multiple Criteria Decision Making Theory and Applications. Springer, Berlin (1980) 468-486.
9 Deb K., Sundar J., Bhaskara U., Chaudhuri S.: *Reference Point Based Multi-Objective Optimization Using Evolutionary Algorithms,* ISSN International Journal of Computational Intelligence Research, 2, 3 (2006) 273-286.
10 Ferreira J. C., Fonseca C. M., Gaspar-Cunha A.: *Methodology to Select Solutions from the Pareto-Optimal Set: A Comparative Study,* In: Proc.of ACM-SIGEVO on Genetic and Evolutionary Computation GECCO'07 (2007) 789-796.
11 Woźniak P., Sobieraj T.: *Multi-objective control design by experiment-based evolutionary methods : case study,* approved for presentation at the Conference on Control for Power Electronics and Electrical Drives, SENE'07 (2007).
12 Ogata, K.: *Modern Control Engineering,* 3 rd Ed., Prentice Hall, 1997.

Proceedings of the International Multiconference on
Computer Science and Information Technology pp. 347–356
ISSN 1896-7094
© 2007 PIPS

On Dimensionality of Latent Semantic Indexing for Text Segmentation

Radim Řehůřek

Faculty of Informatics, Masaryk University
xrehurek@fi.muni.cz

Abstract. In this paper we propose features desirable of linear text segmentation algorithms for the Information Retrieval domain, with emphasis on improving high similarity search of heterogeneous texts. We proceed to describe a robust purely statistical method, based on context overlap exploitation, that exhibits these desired features. Ways to automatically determine its internal parameter of latent space dimensionality are discussed and evaluated on a data set.

1 Introduction

Text segmentation has recently enjoyed increased research focus. Segmentation algorithms aim to split a text document into contiguous blocks, called *segments*, each of which covers a compact topic while consecutive blocks cover different topics. Applications include finding topic boundaries in text transcriptions of audio news, improving text navigation or intrinsic plagiarism (or anomaly) detection. It can also be used to improve Information Retrieval (henceforth IR) performance, which is main target application for the method described in this paper.

To see how text segmentation might improve IR performance, consider standard IR scenario. Here documents are transformed into Vector Space and indexing techniques are employed to allow efficient exact and proximity queries. Given the widely heterogeneous documents that a general IR system may expect, some of these documents may be monothematic and compact, dealing with a single topic. Others can be a mixture of various topics, connected not thematically but rather incidentally (for example, documents containing news agglomerated by date, not topic). Some may cover multiple topics intentionally, such as complex documents involving passages in different languages. The problem here is that once the documents are converted into Vector Space, all structural information is lost. The resulting document vector shifts away from any one topic included in the original document. Still, user queries are typically monothematic and in this way the chance of high similarity match between user query and document vector decreases. This can result in missed hits. Thus having basic retrieval blocks correspond to single topics rather than whole documents seems like a methodologically sound step.

1.1 Motivation

There are practical considerations that are important in real-world IR systems. Driven by need to understand system's behaviour (especially unexpected behaviour) and ability to make extensions to the system during development cycle, it is advantageous to keep the system architecture as simple, clear and robust as possible. Based on these concerns, three important properties of text segmentation for IR systems may be identified:

1. Domain independence. As little external knowledge about document content as possible is required for segmentation. For large collections, even semi-automatic techniques (i.e., techniques that require some kind of human intervention during segmentation process) are problematic.
2. Language independence. Although techniques for automatic language detection exist, using separate algorithms for different types of input data is difficult from maintenance point of view. An algorithm should ideally work based solely on document content. Additionally it should be able to deal with the case where input text is not a strictly well-behaved part of a language, but rather a real-world document. An example is robustness towards commonly appearing inserted content such as small ASCII drawings, inserted passages in different languages, short tables, poorly OCR-ed documents and so on.
3. Granularity. Desirable is an option that allows one to set a customizable level of granularity as to what constitutes a "sufficient" topic shift. This allows the system maintainer to set the segmentation policy, based on expected query granularity. There is no point clogging the system with many small segments that are conceptually identical with respect to user interests, just as it's no good to keep widely varying topics in one single document.

In addition to this list, technical aspects such as effectiveness are also important. Segmentation must perform well enough to allow text segmentation of large documents in reasonable time. This limits our choices to algorithms based on readily identifiable surface cues, such as those based on text cohesion and lexical repetition.

2 Existing methods

Different text segments should more or less correspond to different topics. Paragraph breaks could be indicative of a topic shift. However, simply splitting at new line breaks is problematic. In many documents the new line breaks are not indicative of paragraphs, but rather of the place where text meets right margin of the page. This may be caused by OCR, certain text editors or retransmission through e-mail clients. More sophisticated segmentation algorithms which take into account document content are required.

An early linear segmentation algorithm called TextTiling is due to Hearst [4]. The basic idea is to take note of lexical repetition. A window of fixed length is

being gradually slid through the text, and information about word overlap between the left and right part of the window is converted into digital signal. Shape of the post-processed signal is used to determine segment breaks. Complexity of this method is linear in the length of the document and requires virtually no linguistic knowledge or tuning beyond the choice of parameters for window size, step for window slide and segmentation threshold[1]. These features make TextTiling a good candidate for IR purposes.

A more recent approach is that of Choi [1]. His C99 algorithm computes a sentence similarity matrix (again using lexical cohesion), and applies ranking to get rid of unreliable absolute values. Divisive clustering on the rank matrix is applied to obtain final segmentation. Choi's results show that C99 is vastly superior to the previous methods of Hearst, various modifications of Reynar's maximization algorithm [9] and the Segmenter algorithm [5].

3 LSITiling

Our proposed algorithm, called LSITiling, is based on and extends Hearst's TextTiling algorithm. It uses the same sliding window to obtain context similarity signal, then analyses this signal to arrive at final segmentation. The difference lies in what constitutes a context. Where TextTiling relies on text surface features of character token units to determine context overlap, LSITiling uses context overlap in a conceptual space. The reasoning behind this enhancement is following: segmentation aims at discerning individual topics. These may be vaguely defined as chunks of text pertaining a single idea. People, however, commonly use different words to express the same idea. Indeed, it is a recommended stylistic guideline to change vocabulary and avoid lexical repetition. TextTiling (as well as other algorithms) go a step in the direction of lexical independence by stemming the text — that is, unifying tokens that share a common word stem. Apart from failing point two of our three point wish-list (language independence), this does not address basic common language phenomena like synonymy.

Our algorithm computes the context similarities in *topic space*. Topic space is a space where each dimension represents a concept present in original data. In a general sense, LSITiling is thus a Vector Space method. Construction of the concept space (also called *embedded* or *latent* space) is done by applying an IR technique called Latent Semantic Indexing (LSI). LSI was chosen for its established role in IR, but other decomposition schemes, such as the recently proposed Non-negative Matrix Factorization [6], are also applicable. LSI is basically a straightforward statistical method which projects data onto a lower dimensional space, where this projection is optimal in the least squares sense. Translated into our text domain, it uses word co-occurrence both of first and higher degrees to derive co-occurrence patterns, called *concepts*, which define dimensions in the new concept space.

In our algorithm, LSI is applied to the domain of text segmentation as follows: let input be a single text document we wish to segment. Next we construct a

[1] See Hearst [4] for exhaustive algorithm details.

corpus of *pseudo-documents*. This is a critical step — although previous work by Choi [2] or Foltz [3] also sought to apply LSI to text segmentation, they split the input document into individual sentences and/or paragraphs and used those as corpus of pseudo-documents. Additionally, Foltz added other domain-specific documents such as encyclopaedia entries and book excerpts to the corpus of pseudo-documents to increase quality of the latent model. Here our approach differs in that we consider *overlapping* pieces (or chunks) of the input document. In practice, we choose as pseudo-documents exactly those chunks as considered by the TextTiling algorithm as contexts. All terms are identified (no stemming is performed) and *Term Frequency * Inverse Document Frequency*, or $TF * IDF$, matrix is constructed. On this matrix, Singular Value Decomposition is computed[2]. A number of latent dimensions is chosen and each pseudo-document is projected into the new, reduced latent space.

Dimensionality of the latent space determines how much data variance is retained. Using only the single, most dominant concept would result in each pseudo-document being represented by a single number. Keeping more dimensions keeps more information about the pseudo-document's content, allowing more refined representation in terms of adherence to multiple concepts. As the other extreme, keeping all dimensions means all information present in the data is kept. It is however desirable to keep the number of latent dimensions low. This is to reduce data noise — dimensions with low significance correspond to noise and thus affect performance negatively. Also, for the sake of segmentation, we are only interested in the salient document concepts. Here our approach also differs in that we explore effects of very low dimensionality, as opposed to Choi [2] where the golden IR standard of hundreds of dimensions was used. An interesting algorithmical feature of LSI is that concept spaces are nested. This means that once we have obtained a document representation in for example 100 most significant dimensions, we may work with the document in 10 dimensional space by simply omitting all but its first 10 dimensions. This allows us to produce and store a single latent model, and tune concept granularity by only adjusting vector lengths.

Once the new context representations are obtained, context similarity is computed by standard cosine similarity measure:

$$\cos(vec_i, vec_{i+1}) = \frac{vec_i^T \cdot vec_{i+1}}{|vec_i||vec_{i+1}|} \tag{1}$$

By computing cosine similarity between consecutive pseudo-document vectors we obtain a one-dimensional digital time series, which is smoothed and analyzed for local minima in the same way as described in Hearst [4]. A notable distinction between LSITiling signal and TextTiling signal is signal variance. With LSITiling, most of the signal for overlapping pseudo-documents runs at near exact-match (cosine similarity score of 1.0), with occasional drops near suspected segment boundaries. This allows us to reliably set the segmentation threshold at a high number (such as 1.0), as opposed to TextTiling where the

[2] The PROPACK [8] package for computing SVD on large, sparse matrices was used.

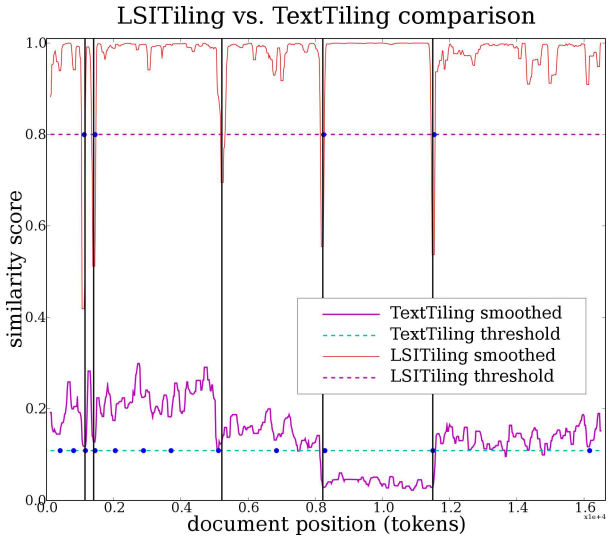


Fig. 1. Comparison of smoothed LSITiling and TextTiling signals on a sample input text of 16,634 tokens. TextTiling threshold is set to signal mean minus half its standard deviation (as proposed by Hearst). The black vertical lines denote true segment boundaries. Compare to blue dots along threshold lines, which denote segment boundaries proposed by respective algorithms. In this example, LSITiling missed one segment boundary. TextTiling missed no boundary but produced seven additional, erroneous boundaries.

threshold is set according to a particular combination of signal mean and variance. See Figure 1 for side-by-side comparison of LSITiling and TextTiling signals over the same sample document.

4 Experiments

In our previous work, we compared LSITiling performance with six other segmentation algorithms. In this paper, we seek to explore relationship between LSI dimensionality and algorithm performance. In the next two sections, we briefly describe evaluation metrics and data sets used to conduct experiments.

4.1 Evaluation

Given input document, its true segmentation and a segmentation proposed by some segmentation algorithm, we seek to numerically capture quality (or "closeness") of the two segmentations. This is a common concept in IR evaluation, however traditional IR evaluation measures of precision and recall are not suitable. This is because they penalize "near-misses" too much, whereas we'd like to penalize errors more softly, depending on distance between true and proposed

segment boundary. We chose the WindowDiff metric proposed in [7]. The idea is to run a window of fixed length through input text, taking note of how many times the number of proposed and real segment boundaries within the window disagree. Length of this window is set to half the average true segment length. Resulting error value is scaled to lie between 0.0 and 1.0 (0.0 being the best and 1.0 the worst score). For a discussion of WindowDiff's advantages over the previously used P_k probability metric as well as its theoretical justification, see [7].

4.2 Data set

Table 1. Overall corpus statistics

corpus stats	document length (tokens)	segment length (tokens)
mean	51549	7003
stddev	32883	5711

Evaluating segmentation performance on real documents is problematic. This stems from the difficulty of obtaining a referential ("true") segmentation. Indeed, given two human judges, three proposed "true" segmentations are likely to appear. One way to overcome this difficulty is to combine results of multiple human judges. Another way is to create an artificial document by merging several smaller texts that deal conclusively with different topics. This way, although the resulting document is artificial, we can evaluate segmentation quality automatically and precisely.

We adopted the latter approach. For this purpose, a collection of 31 text *passages* was assembled[3]. These were picked to represent various topics and styles. 24 of these passages were written in English (both by native and non-native speakers), two in German, one in Czech and one in Dutch. Also present were text passages of unidentifiable language, such as an ASCII art picture or text table drawings. These 31 passages thematically covered law, biology, computer science, art, religion and politics. The texts took many different forms, including book excerpts, scientific papers, paper abstracts, text questionnaire, executive summary, part of a stage play, an interview, product overview or end user license. Two of the passages came from poor book OCR, resulting in much garbled content. All passages were chosen with requirements that

- they deal with mutually exclusive topics, so that evaluation of true segment boundaries is possible,
- and that they are real-world documents, the concatenation of which may conceivably appear in a digital collection.

[3] All data sets are available from author on request

Apart from lack of pair-wise semantic overlap between passages, no other constraints were placed and no text postprocessing was done. Passages had widely varying lengths and formatting, mimicking our requirements placed on segmentation algorithm mentioned above. From these passages, a corpus of 525 documents was assembled. Each document was created by merging one to fourteen passages, which were picked randomly without repetition. Each test document is thus made up from one to fourteen true segments, and the segmentation algorithm then tries to retrieve these segments given their concatenation as its only input. Corpus statistics are summarized in Table 1. Average length is 7003 tokens per segment or 51549 tokens per document. Notable is the high passage length deviation, implying text segments of very different lengths.

One objection to constructing our test corpus in this way may be that it produces artificial documents. While this is true, we note that the alternative, tagging a real corpus by hand by human judges, is quite problematic both theoretically and practically. We further note that using artificial corpora is common in segmentation evaluation research and our approach is in this respect comparable to other works in the field.

4.3 Algorithm parameters

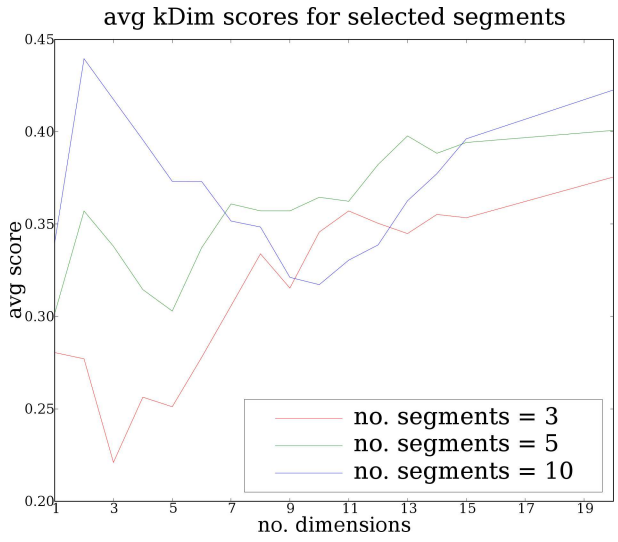


Fig. 2. Average *kDim* scores for documents with selected number of true segments. Scores for *kDim* values above 20 did not improve and were clipped from the figure to improve readability.

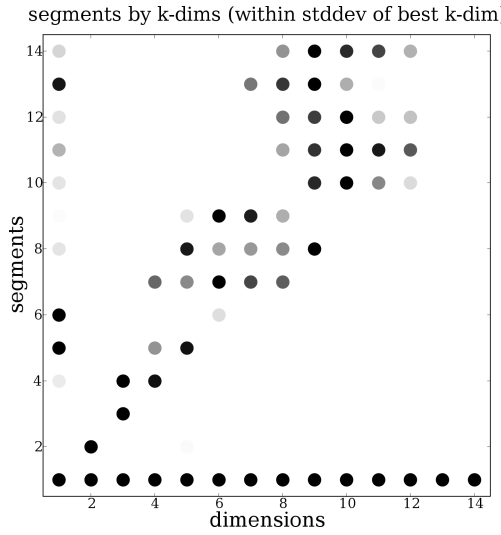


Fig. 3. Best *kDim* scores per number of segments. See text for explanation.

The choice of value for the latent space dimensionality parameter is not straightforward. Our next series of tests therefore sought to establish a relationship between the data and this parameter. Ideally, we would like to make setting of this parameter transparent to the user, so that LSITiling requires no additional parameters to these of TextTiling. To establish relationship between the number of document segments and the dimensionality (also referred to as *kDim*), we evaluated LSITiling for each document of the corpus, with *kDim* values between 1 and 100. As an example, Fig. 2 shows average scores for all documents that have 3, 5 or 10 true segments. For these documents we force LSITiling to return respectively 3, 5 or 10 proposed segments as well, by splitting the input document at the 3, 5 or 10 gaps with most dissimilar left and right contexts. From this graph we may observe that the optimal *kDim* value appears to equal the true number of segments. Going through the same procedure for all other segments number (recall that the corpus has between one and fourteen true segments per document) and plotting the results we obtain Fig. 3. The plots denote all *kDim* values for which the score was within one standard deviation from the optimal value, with plot intensity decreasing as distance to optimal increases (solid black dots denote best scores). From this graph we may observe three prevailing trends. The horizontal line at true segments equal to 1 is trivial — it simply shows that for one true segment, with LSITiling forced to return one true segment, the segmentation is (trivially) perfect. Second trend is diagonal — it shows that the relationship between number of segments and *kDim* may in fact be as simple as linear. Third trend is the vertical line at small *kDim*. This is not trivial and shows that basing segmentation on very small fixed

number of latent dimensions may be justified. This is in contrast with [2], where the authors use hundreds of latent dimensions in their proposed LSI algorithm.

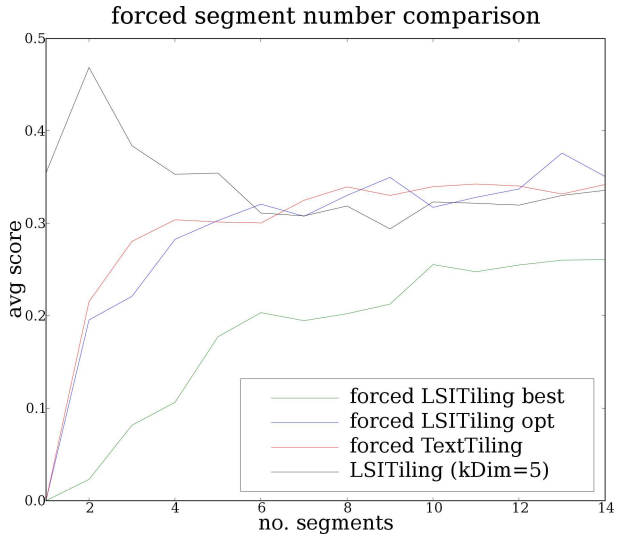


Fig. 4. Comparison of $kDim$ selection strategies

Figure 4 summarizes these results with respect to number of document segments. It plots the original LSITiling result ($kDim$ equal to 5) against results obtained by fixing the number of hypothesized segments to the true number of segments. *LSITiling best* denotes the results obtained by "magically" picking the best $kDim$ for each number of segments. It is included for illustrative purpose only, because obviously in practice we do not have a referential segmentation to determine what the best $kDim$ is. *LSITiling opt* plots the results obtained by setting $kDim$ to the number of true segments (the diagonal trend). Although not straightforward, it is feasible to determine the number of segments for new, unseen texts. Clusterization techniques (such as the one used in C99) can be employed here, and in particular Spectral Graph Analysis, close in spirit to SVD, seems promising. Also included in the graph for reference are TextTiling scores, where the algorithm is forced to return segmentation with a fixed number of segments in the same way LSITiling was. We can see that for low number of true segments, all forced algorithms perform better than the default LSITiling. This means that default LSITiling overestimates the number of true segments here. However, for higher number of segments, the default LSITiling performs comparably to *LSITiling opt* and forcing it to return a fixed number of segments offers on average no improvement.

5 Conclusion and future work

In this paper we have presented an algorithm for linear text segmentation based on Latent Semantic Indexing. The algorithm was devised to satisfy Information Retrieval needs for robustness, independence of language, independence of domain and of adjustable segmentation granularity. Its target application is one where heterogeneous texts of widely varying lengths can be expected. Here it exhibits encouraging performance and outperforms other commonly used algorithms. Its robustness draws from design which does not rely on linguistic tools such as sentence or paragraph boundary disambiguation, language dependent stemmers, stop-lists and thesauri. For the same reason, its crude and statistic-based behaviour is mainly suited for refinement of Information Retrieval systems, where a missed or superfluous boundary is not critical for overall system usability.

We have also explored how LSITiling's internal parameter of space dimensionality may be tuned automatically. This is in contrast with previous works in the field which rely on setting the dimensionality to magical multiples of 100, based on results from LSI research in other IR areas. Future work will seek to determine exact influence of (the amount of) context overlap used in LSI training, which will help fully understand the algorithm's behaviour.

Acknowledgments

This work has been partly supported by the Ministry of Education of ČR within the Center of basic research LC536.

References

1. Choi F. Y. Y., 2000. Advances in domain independent linear text segmentation. In *Proc. of the North American Chapter of the Association of Computational Linguistic*, pp. 26–33, Seattle.
2. Choi F. Y. Y., Wiemer-Hastings P., Moore J., 2001. Latent semantic analysis for text segmentation. In *Proc. of the 6th Conference on Empirical Methods in Natural Language Processing*, pp. 109–117.
3. Foltz P. W., 1996. Latent Semantic Analysis for text-based research. In *Behavior Research Methods, Instruments and Computers*. 28, pp. 197–202.
4. Hearst M. A., 1994. Multi-paragraph segmentation of expository text. In *Proc. of the ACL'94*. Las Crces, NM.
5. Kan Min-Yen, Klavans J. L., and McKeown K. R., 1998. Linear segmentation and segment significance. In *Proc. of WVLC-6*, pp. 197–205, Montreal.
6. Lee D. D., Seung H. S., 1999. Learning the parts of objects by non-negative matrix factorization. Nature, 401(675), pp. 788–791.
7. Pevzner L. and Hearst M., 2002. A critique and improvement of an evaluation metric for text segmentation. *Computational Linguistics*, 28(1):19–36.
8. `http://sun.stanford.edu/~rmunk/PROPACK/`, March 2007.
9. Reynar J. C., 1998. Topic segmentation: Algorithms and applications. Ph.D. thesis, Computer and Information Science, University of Pennsylvania.

Proceedings of the International Multiconference on
Computer Science and Information Technology, pp. 357 – 364
ISSN 1896-7094
© 2007 PIPS

Information System Based on Natural Language Generation from Ontology

Wojciech Górka, Łukasz Bownik, Adam Piasecki

Centrum Elektryfikacji i Automatyzacji Górnictwa "EMAG"
ul.Leopolda 31, 40-189 Katowice

{wgorka, lbownik,apiasecki}@emag.pl

Abstract. Knowledge, facts and dependencies are usually recorded as texts. This method of gathering knowledge causes a situation when some dependencies between certain concepts are not visible. The ontology-based recording of knowledge enables to easily integrate data from many sources and to determine dependencies between them. In the course of the specific targeted project a computer system was developed. The system (which operates on polish language) gives the user access to information from a wide range of topics. The adopted method of communication is based on a natural language and makes use of information assets accumulated in ontologies.

1. Introduction

Storing knowledge and facts is related to the introduction of concepts, the properties of these concepts, and relationships between concepts. Storing knowledge in the form of a natural-language sentence (descriptive text) contains concepts, dependencies and properties. However, due to its nature, this way of recording limits the possibilities to process knowledge as well as to compare and connect similar concepts. Thus it is not possible to automatically combine knowledge from two different sources with the purpose to obtain some extra cross-sectional information based on two separate documents. Such cases refer particularly to the knowledge that resembles data structures where there is focus on certain dependencies between entities. An example is a description of a device and sub-assemblies the device consists of (catalogue of products, catalogue of sub-assemblies). The description will comprise not only typical information on a given sub-assembly but also the dependencies, e.g. which of other sub-assemblies is able to replace the given one, what other sub-assemblies it consists of, what material it is made of, etc. Similarly, scientific research results that contain parameters and their mutual dependencies can be described with the use of such a structure. This kind of solution enables to easily connect data from various sources and to find new dependencies.

Although this way of data description can be easily processed by a computer, it is less readable for the user. An ideal situation for the user would be the possibility to "ask" the structural knowledge base with the use of a natural language and obtain answers in the form of grammatical sentences. The goal is not to hide from the user

the fact he/she "talks" to the computer but to formulate answers and interpret questions in a more ergonomic way[1].

2. Operation Concept of the Application

The objective of the specific targeted project was to develop a computer system (basing on polish language) which provides the user with information on a multi domain issues (eg. issues from public administration, medical health, tax) The number of topics comprised by the system was limited by the range of ontology on which the system operation was based. The applied communication method was based on a natural language. Thus the user should be able to ask questions to the system in natural language. According to the assumptions, the answers should be given in a readable form, i.e. to the best of abilities, as grammatical sentences in a natural language.

The use of the ontology was to provide easy knowledge management and flexibility of the solution, as well as to enable the construction of a structural form of recording facts which, at the same time, is adapted to the natural-language dialogue with the user.

3. Ontologies

According to Webster's Revised Unabridged Dictionary the word "*ontology*" means: "*That department of the science of metaphysics which investigates and explains the nature and essential properties and relations of all beings, as such, or the principles and causes of being*" [8].

In computer science term "*ontology*" refers to formal description of part of the world that could help in knowledge sharing and reuse. An important definition of ontology was also introduced by Tom Gruber[9]: "An *ontology is a specification of a conceptualization*" A conceptualization is an abstract simplified view of the world that we wish to represent for some purpose.

In order to made ontology specification, in 1997 World Wide Web Consortium (W3C) began the standardization process for ontology description by publishing Resource Description Framework (RDF) [2]. This standard became W3C's official standard in 1999 and was updated in 2004 with the RDF Schema (RDF-S) specification [3].

RDF allows to record triples of concepts in a simple way. Each triple is a subject-predicate-object expression which define piece of information. Such expressions build a network of definitions (each object can be a subject in a different triple).

RDF-S introduced the possibility to build meta-concepts: classes, sub-classes, properties. It also launched a non-standard way of defining the name of the notion (label) and its description (comment). Next stage to extend the semantic web standards was to increase the expressiveness of languages intended for ontology recording. W3C published the OWL (Web Ontology Language) standard [4]. It allows, among others, to express the number of concept sets, to show how one

concept belongs to or differs from the other, to identify necessary and sufficient conditions for a given concept. Greater expressiveness of a language allows to verify concepts added to the ontology and to search out certain facts and features indirectly. Additionally, this language makes it possible to integrate two ontologies by means of associating their identical concepts.

Therefore, ontology description standards allow to describe concepts and the network of concepts.

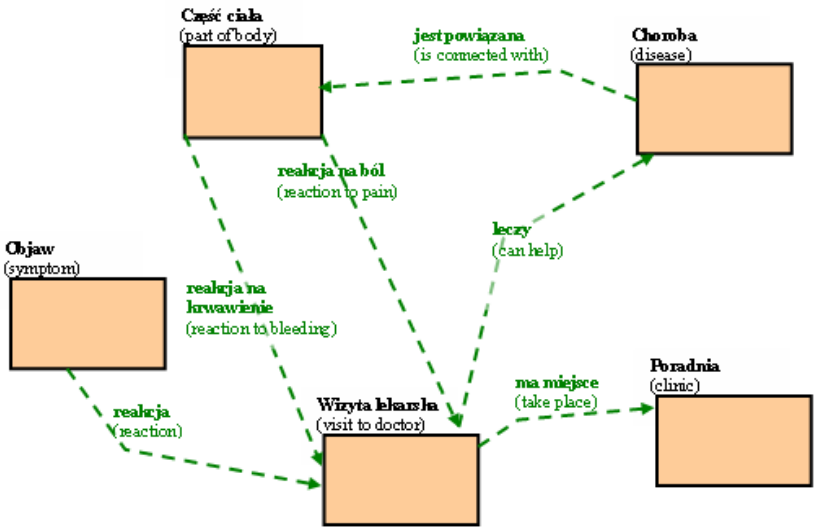


Fig. 1. Ontology graph – square elements are concepts and arrows are properties

4. Ontologies and Natural Language

Searching a knowledge base is, first of all, browsing through particular concepts (subject or object in mentioned triples), reading their properties (predicates in triples) and navigating between associated concepts. The concepts are usually nouns in terms of grammar. The properties define the concepts or act as links between two concepts, thus they can be verbs or adjectives.

Some of concepts from ontology showed in example (see Fig. 1) can be written as:

```
<Cz    cia a> <reakcja na ból> <Wizyta lekarska>
```

(eng.: <part of body> <reaction to pain> <visit to doctor>)

```
<Choroba> <jest powi zana> <Cz    cia a>
```

(eng.: < disease> <is connected with> <part of body>)

It is possible to write triples that show dependencies between some individuals - "materialized" concepts (not showed in Fig. 1):

```
<G owa> <reakcja na ból> <Wizyta u Neurologa>
```

(eng.: <head> <reaction to pain> <visit to neurologist>)

Natural languages have different ways of building sentences. In some languages, the position of a word in the sentence is strictly determined. This facilitates the sentence analysis which, in turn, allows to precisely determine the meaning of the sentence.

Polish is not a positional language. Verb, subject, attribute, etc. can occur in different positions in the sentence[5]. However, Polish has fixed connections between parts of speech. These connections determine dependencies between particular parts of the sentence, i.e. the grammatical form of one part of the sentence enforces the grammatical form of the other part[6]. Unfortunately, these dependencies do not have strict character. They depend on the style of the sentences and their types too. For example, questions will have a different word order and different dependencies between forms.

Due to high complexity of these dependencies they were not used in the developed system. Thus, the questions were interpreted and the answers generated on the basis of the defined concepts and properties. Each concept and property was also equipped with other meaningful words. Additionally, morphological analyser was used to lemmatise words.

5. Interpreting the Users' Questions

When the user asks a question, uses certain concepts – meaningful words. The identified words indicate proper concepts or properties in the ontology. The Apache Lucene[7] full-text browser was used for searching elements (concepts and properties) in the ontology, based on the user's questions. On this basis the answers are searched out.

The user's question may indicate:

- one or a few concepts,
- concept and property,
- property.

When a single concept or a few concepts are found, the answers engine will return the description of the given concept.

If the user's question indicates a concept and a property, system treats it as a question about some aspect of concept pointed by property, or another concept which is connected with given concept using property. In such case, possible triple combinations are searched out in the ontology. In the triples the concept may be a subject or an object, while the property is a predicate. Ontology allows to record the hierarchy of concepts (including inheritance) and relations between them. On the basis of this hierarchy, in the course of searching out a proper concept-property pair, more general concepts (placed one level higher in the hierarchy) are also used. This results from the fact that in his/her question the user can use a concept which is too

specific, i.e. does not have the indicated connection, while a more general concept does have the given property. If adequate triples are found, the third element of the triple, indicated by the concept and the property, is presented.

In the case when only the property was searched out, it is necessary for the user to specify the concept (see example in Fig. 2). For this purpose, the engine generates an additional question to the user on the basis of the indicated property. Therefore, it is necessary to assign a question template to each property. The template will be used in case the search results are specified.

At each stage of asking, the user receives an answer in the form of meaningful words which can be used. These words are generated based on the names and key words associated with concepts and properties. When a property is specified, a hint is generated based on the concept-property pairs that can be searched out.

The ergonomics of working with the system depends on the way the information is organized according to the search-out accuracy. Each of the searched out concepts and properties has its significance on whose basis it is put into a proper place in the answer. The significance of the concept is affected by the value calculated on the basis of a full-text analysis.

Ontology (written as triples):
<brzuch> <reakcja na ból> <wizyta u ginekologa>
(eng.: <stomach> <reaction to pain> <gynaecologist>)
<brzuch> <reakcja na ból> <wizyta u lekarza rodzinnego>
(eng.: <stomach> <reaction to pain> <GP>)

Question template:
<reakcja na ból> => "Co cię boli"
(eng.: <reaction to pain> => „What hurts you")

Answer template:
<reakcja na ból> => "W przypadku bólu <?> proponuje"
(eng.: <reaction to pain> => „In case of pain in <?> try ")

The user enters the question:
Coś mnie boli *(eng.: I have a pain)*

The system asks for specification:
Doprecyzuj: Co Cię boli ? *(eng.: Let me ask you: What hurts you ?)*

User answers:
Brzuch *(eng.:stomach)*

System answers :
W przypadku bólu brzuch proponuje: *(eng.: In case of pain in stomach try)*
- Wizyta u ginekologa *(eng.: gynaecologist)*
- Wizyta u lekarza rodzinnego *(eng.: GP)*

Fig. 2. Using ontology to precise user question. *User says that something hurts him ("Coś mnie boli"). System searches out property "<reakcja na ból>"and asks question – "Co Cię boli" and then (after user's answer – "Brzuch") gives answer.*

6. Generating Answers for the User

By analyzing the user's question and the answers selected by him/her, the system will be able to describe a concrete concept. The concept description consists of a textual description, list of properties and list of connections with other concepts (properties that lead to other concepts). It is important that these elements should be presented in a readable way. Based on the searching path, it is possible to select a proper section of the concept description so that the user could see only the section he/she is interested in.

If the answer to the user's question comes straight from the indicated concept, this very concept is presented. Description of concept is based on connections (concept properties) with other concepts. System uses answer templates associated with properties to generate such description (see example in Fig. 3).

Ontology (written as triples):

<głowa> <reakcja na skaleczenie> < wizyta u lekarza rodzinnego >
(eng.: <head> <reaction to injury> <GP>)
<głowa> <reakcja na ból> < wizyta u lekarza rodzinnego >
(eng.: <head> <reaction to pain> <GP>)
<głowa> <reakcja na ból> < wizyta u neurologa>
(eng.: <head> <reaction to pain> <neurologist>)
<głowa> <ból wskazuje na> <migrena>
(eng.: <head> <pain indicates> <migraine>)

Answer templates:

<reakcja na ból> => "W przypadku bólu <?> proponuje"
(eng.: <reaction to pain> => „In case of pain in <?> try ")
<reakcja na skaleczenie> => "W przypadku skaleczenia <?> proponuje"
(eng.: <reaction to injury> => „In case of injury of <?> try ")
<ból wskazuje na> => "Ból <?> wskazuje na"
(eng.: <pain indicates> => „Pain in <?> indicates ")

System answers, describing concept: "głowa":

W przypadku skaleczenia głowa proponuje: *(eng.: In case of injury of head try)*
- Wizyta u lekarza rodzinnego *(eng.: GP)*
W przypadku bólu głowa proponuje: *(eng.: In case of pain in head try)*
- Wizyta u lekarza rodzinnego *(eng.: GP)*
- Wizyta u neurologa *(eng.: neurologist)*
Ból głowa wskazuje na: *(eng.: Pain In head indicates)*
- Migrena *(eng.: migraine)*

Fig. 3. Using ontology to describe concept "głowa" basing on knowledge stored in ontology.

There is also possibility that pair concept-property does not indicate a different concept but a textual description (see example in Fig. 4). Textual descriptions help producing answers to user, when storing knowledge and generating answers using triples would be too complicated.

The hierarchy of concepts is also presented to the user. The presentation includes more specific concepts (children), more general concepts (parents), and the concepts with the same degree of specification (siblings). Thanks to this, the user can easily get familiar with the surroundings of the given concept. The hierarchy of concepts was not presented in the textual form because generalizations and specifications have

different meanings for different concepts and, additionally, the hierarchy can be extended (many specifications, many concepts at the same level) – all these factors would make the description unreadable.

Ontology (written as triples):

<zmiana lekarza> <koszt> <Pacjentowi przysługuje bezpłatnie zmiana lekarza, pielęgniarki i położnej podstawowej opieki zdrowotnej...>

(eng.: <change the doctor> <payment> <(text containing description of payment in case of changing the doctor)>)

Answer templates :

<koszt> => "Zagadnienie kosztu związanego z <?> jest następujące"

(eng.: <payment> => „Payment in case of <?> is as follows: ")

System answers, describing concept: "zmiana lekarza":

Zagadnienie kosztu związanego z zmiana lekarza jest następujące :

(eng.: Payment In case of change the doctor is as follows)

Pacjentowi przysługuje bezpłatnie zmiana lekarza, pielęgniarki i położnej podstawowej opieki zdrowotnej nie częściej niż dwa razy w roku kalendarzowym. W przypadku każdej kolejnej zmiany, pacjent wnosi opłatę w wysokości 80 zł. Nie dotyczy to sytuacji zmiany miejsca zamieszkania pacjenta bądź zaprzestania udzielania świadczeń przez wybranego świadczeniodawcę lub innych przyczyn niezależnych od pacjenta.

Fig. 4. Using ontology to describe concept "zmiana lekarza" basing on knowledge stored in ontology as textual data.

7. Conclusions

The presented solution is an attempt to develop a service providing a universal method of searching and presenting the structural data sources. Not all types of information assets fit this model. Therefore the solution can find application first of all in such cases where the knowledge has an organized, structural character by nature.

In the presented solution the stage of identifying the user's question was simplified. It is possible to extend it by analyzing the sentence on the basis of rules and connections of the Polish language .

Another solution to improve searching results could be organized net of words that allows to find out synonyms and meaning of given word. Such a solution could be SKOS described in [10], WordNet[11] and its equivalent for Polish language [12], used with combination with full-text browser.

In all presented examples, there was used very simple way to fill in templates – just replace "<?>" element with concept name. Future work would be application of tool generating proper form of words used in templates.

References

1. Davies J., Studer R., Warren P., *Semantic Web Technologies,* Wiley 2006
2. W3C, *Resource Description Framework,* http://www.w3.org/RDF
3. W3C, *Resource Description Framework Schema,* http://www.w3.org/TR/rdf-schema/
4. W3C, *OWL Web Ontology Language,* http://www.w3.org/TR/owl-features/
5. Vetulani Z.., *Komunikacja człowieka z maszyną. Komputerowe modelowanie kompetencji językowych,* Akademicka Oficyna Wydawnicza Exit.
6. Saloni Z., Świdziński M., *Składnia współczesnego języka polskiego,* Wydawnictwo Uniwersytetu Warszawskiego, Warszawa 1981.
7. Apache, Lucene, http://lucene.apache.org/
8. *Webster's Revised Unabridged Dictionary,* http://www.dict.org/bin/Dict
9. Gruber T., *What is ontology?,* http://www-ksl.stanford.edu/kst/what-is-an-ontology.html
10. SKOS, *Simple Knowledge Organization System,* http://www.w3.org/2004/02/skos/
11. WordNet, *Lexical database for the English language,* http://wordnet.princeton.edu/
12. plWordNet, http://plwordnet.pwr.wroc.pl/main/

Proceedings of the International Multiconference on
Computer Science and Information Technology, pp. 365 – 374
ISSN 1896-7094
© 2007 PIPS

Intelligent System for Docking Ligands to Protein Active Sites

Zbigniew Starosolski[1], Andrzej Polański[2]

[1] Silesian University of Technology, Institute of Automatics,
Akademicka 16, 44-100 Gliwice, Poland
zbigniew.starosolski@polsl.pl
[2] Silesian University of Technology, Institute of Informatics,
Akademicka 16, 44-100 Gliwice, Poland
Andrzej.polanski@polsl.pl

Abstract. We presented computational packed for analysis of interaction of molecular ligand-protein complexes. Packed was invented as an open project allows modyfication in simply manner, composed as set of functional blocks of standardised i/o data. The final information from package can be chosen by user interactive from all signals between blocks. Is based on Matlab high-level language environment that enables to perform other various computational analysis based on Matlab toolboxes. The block diagram of package and an example of its use is presented.

1 Introduction

Methods for computational prediction of interactions between proteins and ligands are developing at a fast rate due to a very wide area of potential applications. Many projects in biological and biochemical research lead to establishing experimental values of chemical affinities between chemical molecules (ligands), and active sites of proteins. These experimental data serve for improving and tuning parameters of molecular models for interactions between proteins and ligands. On the other hand, planning of many experiments largely relies on the preprocessing stage involving recruiting best candidates for experimental verification among numerous families of chemical compounds. This preprocessing stage involves computational prediction of energies of interactions between protein active sites and ligands based on models of molecular mechanisms [3, 7, 13, 16, 19, 27].
There are already many systems of modeling and predicting of molecular interactions between proteins and ligands the most popular [1, 2, 8, 10, 12, 17, 18, 20-22, 24], both commercial and freeware. However, due to fast increase in the knowledge on proteins and their molecular interactions they must be updated very often and, especially for new experimental data, it may happen that the software system at use lacks necessary parameters, or does not allow one to perform the required computational project.
In computational projects involving modeling or predicting molecular interactions it is necessary to make choices between different models to pick up the model, which suits best for the interactions under study, to search for appropriate sets of parameters or to pursue procedures for estimating parameters. It is therefore highly desirable to have computational system with a lot of flexibility in choosing between different

models, different algorithms, different sets of parameters, and so forth. Such a system rather cannot be a closed software package, it should be an open project with possibilities of replacing some of its elements by others. It is also desirable to equip the system with some elements of artificial intelligence, that is with elements of adapting and learning, adjusting to changing data structure and learning by using results of modeling computations in future decisions.
In this paper we present an open software system for modeling molecular interactions between proteins and ligands, based on Matlab environment. Our system is very flexible and we develop it towards equipping it with some capabilities of adaptation and learning. In order to present our system we fist describe in more detail the problem of protein-ligand docking. Then we describe the structure of the software system and its elements. We also show example of its use.

2 Description of protein-ligand docking problem

Processes at the molecular level that govern docking ligands to proteins, at the basic level can be formulated by using laws of quantum chemistry, where time evolution of molecular systems is expressed in terms of spatial wave functions of atoms. In practical computations, however, we are using approximations, where dynamics of the system are modeled by using atoms represented by pointlike masses which move in the molecular force fields. Molecular forces are determined by electrostatic and chemical bonding interactions between atoms.

2.1 Molecular field model

Potential energy is a function of a atomic position (x,y,z) normally in Cartesian space. The equation of the potential energy of the system of particles in the molecular force field, commonly used in molecular modeling [16, 27] is presented below:

$$E=\frac{1}{2}\sum_{i=1}^{m} k_{\theta i}(\theta_i-\theta_{0,i})^2+\frac{1}{2}\sum_{i=1}^{n} k_{bi}(b_i-b_{0,i})^2+\frac{1}{2}\sum_{i=1}^{k} v_i[\,1+\cos(n_i\omega_i-\gamma_i)\,]+ \\ +\sum_{i,j} 4\varepsilon_{i,j}\left[\left(\frac{\sigma_i}{r_{ij}}\right)^{12}-\left(\frac{\sigma_j}{r_{ij}}\right)^{6}\right]+\sum_{i,j}\frac{1}{4\pi\varepsilon_0}\frac{q_i q_j}{r_{ij}} \tag{1}$$

In the above equation first three terms describing change of potential energy describes bonded atom interaction: bond length, angles, and dihedral angle of molecule respectively. Next two terms describes non-bonded atoms interaction: van der Waals interaction and electrostatic intersection between atoms.

First term of bonds interaction is a harmonic potential describing interaction of all the pairs of atoms in molecule connected with covalent bond. Each chemical bond has ideal value $b_{0,i}$, and a constant k_{bi} represent strength of it. This term shows all deviation of length b_i from ideal for molecule with m covalent bonds. Second term of equation (1) represent differences of bond angles θ_i from ideal values $\theta_{0,i}$, for all n angles in molecule. Constant $k_{\theta i}$ results from atom type connected with angle. Third

term is energy of the torsion angle deviation describe presence of steric barriers between atoms isolated of one covalent bond. It is modeled as periodic function, n_i is a period, an γ_i – is phase and depends on type of chemical bonds, and type of atoms, constant v_i represent the force of steric barriers. Fourth and fifth terms modeling non-bonded interaction in molecules: van der Waals and electrostatical interactions. Van der Waals , between two atoms arises from a balance between repulsive and attractive forces best modeled as Lennard-Jones function. Constants are given for all parts i,j : well depth ε_{ij}, atom type related constants σ_i, geometic distance between atom i and atom j represented as r_{ij}. Electrostatic potential is calculated as a sum of interaction between nonbonded atoms i and j with its point charges according Coulomb's low, where q_i are the point charges which is attributed to atoms i, ε_o – dielectrical constant of environment.

2.2 Scoring systems for modeling energies of protein-ligand interactions

The general idea of constructing scoring function is to obtain proper values of the energy interaction (binding affinities) between analyzing molecules. Such energy is experimentally determined during several chemical reactions and can be readout from adequate databases. An assumption is: if scoring function is precise for known molecular complexes it is also probable that for unknown molecular complexes gives adequate values of binding affinity.

When describing interactions between molecules we should take into account not only their potential energy, described by equation (1) but also kinetic energies of different molecules related to their thermal motions in the solution. One approach to this interaction between potential and kinetic energies is by simulations with some elements of randomness. The second approach, more adequate to computations supporting experiments with measuring biochemical affinities between molecules, is by using models with some heuristic parts aimed at taking into account the averaged influence of moving (water) particles on the protein-ligand bond. Many such models were presented in the literature [3], [24], [26]. Some of them are presented below.

Bohm

$$\Delta G_{binding} = \Delta G_0 + \Delta G_{hb} \sum_{h-bonds} f(\Delta R, \Delta\alpha) + \Delta G_{ionic} \sum_{ionic} f(\Delta R, \Delta\alpha) + \\ + \Delta G_{lipo} |A_{lipo}| + \Delta G_{rot} NROT \quad (2)$$

In above equation $\Delta G_{binding}$ is obtained as a composition of sums. Each sum represents particular molecular interaction, where terms $f(\Delta R, \Delta\alpha) = f_2(\Delta R)\, f_1(\Delta\alpha)$ are the penalty function:

$$f_1(\Delta R) = \begin{cases} 1 & \Delta R \le 0.2A \\ 1 - \quad (\Delta R - 0.2)/0.4 & \Delta R \le 0.6A \\ 0 & \Delta R > 0.6A \end{cases}$$
$$f_2(\Delta\alpha) = \begin{cases} 1 & \Delta\alpha \le 30^o \\ 1 - (\Delta\alpha - 30)/50 & \Delta\alpha \le 80^o \\ 0 & \Delta\alpha > 80^o \end{cases} \quad (3)$$

All above the coefficients are estimated during calibration processes. Each specifies certain purpose. ΔG_0 – describes the contribution to the binding energy which does not directly depend on any specific interaction with protein, ΔG_{hb} – describes the contribution from an ideal hydrogen bond (neutral), ΔG_{ionic} – describes the contribution from an ideal hydrogen bond (charged), ΔG_{lipo} – represents the contribution from lipophilic interactions, |Alipo| - the surface of contact lipophilic, ΔG_{rot} – describes the loss of energy due to freezing of internal degrees of freedom in the ligand, NROT – The number of rotable bonds NROT is taken as the number of acyclic sp3-sp3 and sp3-sp2 bonds.

There are several sets of coefficients for characteristics chemical compounds and specify solutions (environment), more detail descriptions could be found in [3, 25, 26]

Chemscore

$$\Delta G_{binding} = \Delta G_{hb} \sum_{h-bonds} f(\Delta R, \Delta\alpha) + \Delta G_{metal} \sum_{metal} f(\Delta R, \Delta\alpha) + \\ + \Delta G_{lipo} \sum_{lipo} f(\Delta R) + \Delta G_{rotor} \sum_{rotor} H_{rot} + \Delta G_0 \quad (4)$$

This equation has the same form as [3] but with one difference in term H_{rot} which depends on number and distance of heavy atoms on both sides on rotational bonds:

$$H_{rot} = 1 + (1 - 1/Nrot) \sum_r \left(P_{nl}(r) - P_{nl}'(r) \right) / 2 \quad (5)$$

Where : *Nrot* – number of frozen rotational bonds during docking process, P_{nl}(r) and P_{nl}'(r) are the percentages of non-lipophilic heavy atoms on either side of the rotatable bond.

Score

This function give a value of negative logarithms of dissociation constant pK_d which can also describes binding affinity and is represented as [24]

$$pK_d = K_0 + K_{vdW} + K_{metal} + K_{hbond} + K_{desolvation} + K_{deformation} \quad (6)$$

Each of component assume the responsibility for specific contribution between ligand –protein interaction: K_o – loss of transitional and rotational entropy thought binding ligand – protein phenomena, K_{vdW} – for Van der Waals interaction, K_{metal} – metal – ligand interaction, K_{hbond} – for hydrogen bonding, $K_{desolvation}$ – for desolvation effect, $K_{deformaton}$ – for deformation of ligand.

The components are represented as also as a sum of next terms for example one of the more important is K_{hbond} compound. It is a sum of six components, where *i*-describe the certain ligand atoms, *j* – atoms of protein. There are summed over all ligand and protein atoms like presented below:

$$K_{hbond} = K_{SHB} + K_{MHB} + K_{WHB} + K_{SWH} + K_{MWH} + K_{WWH} \quad (7)$$

In this equation there are distinction between *SHB* -strong, *MHB*- moderate and *WHB*- weak hydrogen bond, and next taking into account water interaction in hydrogen binding: *SWH* – strong, *MWH* moderate, *WWH* week.

$$K_{hbond} = \sum_i \sum_j SHB(d_{ij}) + \sum_i \sum_j MHB(d_{ij}) + \sum_i \sum_j WHB(d_{ij}) + \sum_i \sum_j SWH(d_{ij}) + \sum_i \sum_j MWH(d_{ij}) + \sum_i \sum_j WWH(d_{ij}) \quad (8)$$

Where d_{ij} - geometrical distances between atom of ligand *i* and atom of protein *j*

2 Structure of the Matlab-based system for modeling protein-ligand interactions

Computations concerning protein-ligand interactions are complex and must be organized in a multi stage manner. Arrangement of the computational stages can be different and the overall efficiency of the algorithm depends on its structure.

The Matlab environment has efficient procedures for operation on large matrices that feature we especially use for the changing conformation structure of molecules (ligand, protein) in 3D coordinate space. The block diagram of the Matlab-based system for modeling protein-ligand geometry and computing energies of protein-ligand interactions is given in figure 1. Generally the blocks of the system can be divided into three categories, data input, computational algorithms and data output. Below we discuss the blocks of the system in more detail.

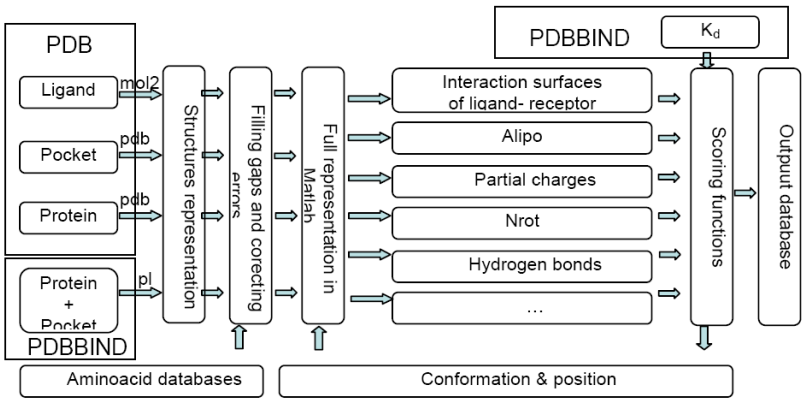


Fig. 1. Block diagram of the Matlab-based system for modeling protein-ligand geometry and computing energies of protein-ligand interactions

3.1 Internet based input data

Feeding data to computational projects may require browsing of internet libraries of chemical compounds and resources of Protein Data Bank (PDB) and database. Matlab environment offers the possibility of programmed browsing of internet depositories and downloading data from them.

An example of using Matlab for downloading data on complex: HIV1-protease in complex with a cyclic sulfamide inhibitor: pdb_index='*1hvr*'

```
Pdb_load(pdb_index,format_type)
```

We are specially interested for such PBD database records that include maximum information for further processing for intelligent system, and can be used in future as a knowledge database. The most interested for us is records from part of PBD database called PDBsum. The majority of 3D coordinates is obtained from X-ray technique, less data is from Nuclear Magnetic Resonance (NMR) technique which has excellent resolution. We can use data from both techniques that include information of docking molecular complexes. We get 3D coordinates of protein pocket with docked ligand into, and information of existing hydrogen bonds between ligand and pocket. This records has extension: *.pl*

For obtaining further information of molecular complexes we use publicity available database named PDBbind from upload chemical types of ligand bonds, also hydrogen coordinates for all complex. There is also a collection of experimental measured Ki, Kd, pKi. If the is no such file deposit in PBD database it means there is no crystal structure of selected protein, there is possibility to predict protein structure based on known amino acids sequences, and similarity between already crystallized structures [5, 11, 14]. It is also possible to obtain such data from Meta-Server [4]. If we are interested in ligands – which are not in PBD database we can always try to obtain it form publicity available ligand database, where are 1,159,274 entries [23].

3.2 Internal parameters

Internal parameters of the system are those, which rather do not change from one computational project to the next one. Our internal parameters are stored in the internal database of amino acids, their structures and chemical properties, bond types and partial charges of their atoms, ideal value of: bonds, atomic angles, dihedral angles [1].

Using internal database of structure amino acids we correct errors and gaps in protein chains (protein pocket) and create full representation of molecular complexes in Matlab environment.

3.3 A collection of protein-ligand interaction models

Procedures for protein-ligand docking models, described in section 2.2 are available as separate blocks easy to replace one by another if necessary. Procedures are elaborated in such a way that their input and output data are standardized, which makes easy their exchange. The basic input data to protein-ligand interaction models are relative positions between ligands and protein pockets and the basic output is the energy or score of their interaction.

3.4 Other computational procedures

Other computational procedures allow us to make several manipulations with the settlement described above. The first type of computations is changing the position of ligand relatively to according to some algorithm, most often according to the algorithm of minimization of the energy. For the second type of computational algorithm is changing the conformation and structure of the ligand. The third type is preparing different types of output data, based on the performed computations. Types of output are listed in the next section. All the preformed computation is based on Matlab environment which was optimized for large matrix calculations and that assure us the low computational complexity for ours project.

3.5 Output

Thanks to flexibility of our system, actually all of the values, input, intermediate or output can be made available. Some of the output formats for which we have developed appropriate procedures are presented in next section

4 Example of the use of the system

For the example we choose well known complex, primary proposed by [15], inhibitor of HIV protease. We calculate he estimate the optimal coordinate of hydrogens producing hydrogen bonding between ligand: $C_{41}H_{38}N_20_3$ and binding site of HIV protease. The schematic of crystal structure can be obtained from PDBdatabase.

Three dimension visualization of this example is shown on fig.2, where in a center of protein is placed docked ligand $C_{41}H_{38}N_20_3$. Always is difficult to present 3D data on paper, to see all necessary details. To show this detail we presented selected feature on the right side on fig.2. Next simulation shows the dependency between angle and distance of hydrogen atoms and bond energy. The possible atom placement is mark by asterisks fig.3 (left) and the reference energy is presented on fig.3.(right). Optimal position is there where energy reaches minimum value.

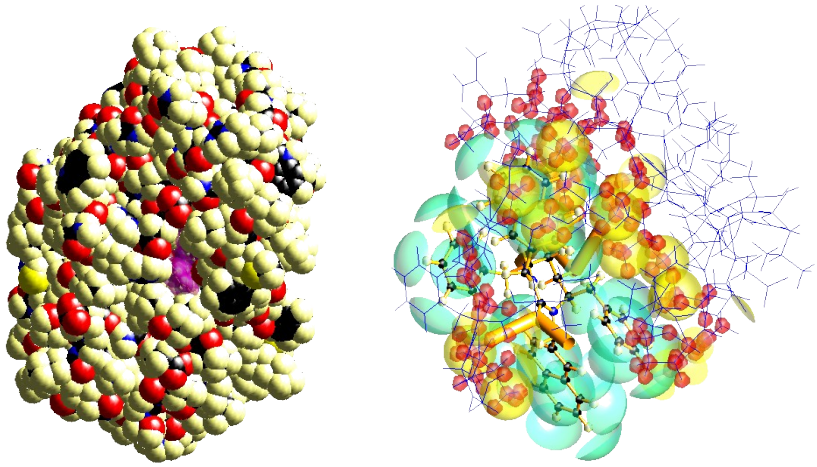


Fig. 2. Left: 3D visualizations of complex of HIV protease with ligand $C_{41}H_{38}N_20_3$ (in the middle), right selected features of complex focused of binding site.

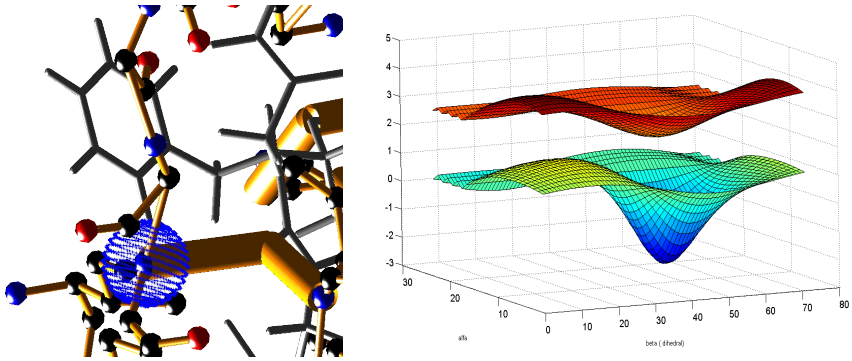


Fig. 3. Left: Asterisks shows positioning for hydrogen bonds, right: upper surface – length of bond [Å], lower- the value of energy of hydrogen bond during positioning [kcal/mol].

PDB records shows seven hydrogen bonds stabilizing the position of ligand in the binding site, in our simulation we take six of them, not taking into account, difficult to explicate bond: *ASP25(B)OD1* and *O5*. The values of energy are presented in Table 1.

Table 1 . . Energy of hydrogen bonds for optimal placement of ligand into binding site

Hydrogen bond (protein –ligand)	Calculated optimal energy kmol/mol
ILE 50(A) N – O1	-2,0772
ILE 50(B) N – O1	-1,8770
ASP25 (A) OD1 – O5	-2,1626
ASP25 (B) OD2 – O5	-2,0637
ASP25 (B) OD1 – O4	-2,1515
ASP25 (A) OD2 – O4	-2,1517

6 Conclusion

In this paper we presented package of procedures for calculation of molecular ligand – protein interaction. It is an open source project which allows easy modification and application of new parameters, ideas, algorithms and etc. Project is implemented in Matlab environment that easily allows combining Matlab standard toolboxes with our package what is in our opinion also its advantage. Project is still in development and grows in new procedures to catch the recent scientific findings.

7 Acknowledgment

This paper was supported by the European FP6 grant GENEPI low RT

References

1. AMBER: homepage: `http://amber.scripps.edu/`
2. AUTODOCK homepage `http://autodock.scripps.edu/`
3. Bohm J.: *The development of a simple empirical scoring functions to estimate the binding constant for a protein-ligandd complex of known three-ddimensional structure,* Journal of Computer-Aided Molecular Design, 8,pp.243-256, 1994.
4. Bujnicki J. M. et al,: *Structure prediction meta server,* Bioinformatics 17 (8), (2001) 750-75.
5. Bujnicki J. M.: *Practical Bioinformatisc Nucleic Amid and Molecular Biology,* (2005) Vol. 15.
6. W. D. Cornell et al. *A Second Generation Force Field for the Simulation of Proteins, Nucleic Acids, and Organic Molecules,* J. Am. Chem. Soc. 117 (1995) 5179-5197.
7. Dauber-Osguthorpe et al.: *Structure and Energetics of Ligand Binding to proteins: E. coli dihydrofolate reductase-trimethoprim, a drug-receptor system,* Proteins: Str. Funct. Genetics 4 (1988) 31-47.
8. GLIDE:homepage `http://www.schrodinger.com/ ProductDescription.php?mID= 6&sID= 6&cID=0`
9. Garrett M. et al.: *Automated Docking Using a Lamarckian Genetic Algorithm and Empirical Binding Free Energy Function,* Journal of Computation Chemistry, 19 (1998) 1693-1667.
10. GOLD: Genetic Optimalization for Ligand Docking:homepage `http://www.ccdc.cam.ac.uk/`
11. Ginalski K., Grishin N. V, Godzik A., Rychlewski L.: *SURVEY AND SUMMARY, Practical lessons from protein structure prediction,* Nucleic Acids Research, Vol. 33, No. 6 (2005) 1874–1891.

12. HyperChem: homepage http://www.hyper.com/
13. Jorgensen W. L., Tirado-Rives J.: *The OPLS Potential Functions for Proteins. Energy Minimizations for Crystals of Cyclic Peptides and Crambin,* J. Am. Chem. Soc. 110 (1988) 1657-1666.
14. Kolinski A, Bujnicki J. M.,: *Generalized protein structure prediction based on combination of fold-recognition with de novo folding and evaluation of models,* Proteins, 61 Suppl 7:84-90,(2005).
15. Lam P. Y. et al.: *Rational design of potent, bioavailable, nonpeptide cyclic ureas as HIV protease inhibitors,* Science vol. 263 pp. 380-384,Springer-Verlag (1996).
16. MacKerell A. D. et al. *An All-Atom Empirical Energy Function for the Simulation of Nucleic Acids,* J. Am. Chem. Soc. 117 (1995) 11946-11975.
17. MOE (Molecular Open Enviroment) : homepage http://www.chemcomp.com/
18. NAMD: http://www.ks.uiuc.edu/Research/namd/
19. Ott K-H., Meyer B.,: *Parametrization of GROMOS force field for oligosaccharides and assessment of efficiency of molecular dynamics simulations,* J. Comp Chem 17, (1996) 1068-1084.
20. Rarey M. et al.: *A fast flexible docking method using an incremental construction algorithm,* J. Mol. Biol. 261 (1996) 470-489.
21.SYBYL homepage
http://www.tripos.com/index.php?family=modules,SimplePage,comp_informatics
22. TINKER: homepage http://dasher.wustl.edu/tinker/
23 von Grotthuss M, Pas J, Rychlewski L: *Ligand-Info, searching for similar small compounds using index profiles,* Bioinformatics. May 22; 19 (8) (2003).
24. VEGA: homepage http://www.ddl.unimi.it/vega/index2.htm
25. Wang R.; Gao Y.; Lai L.: *SCORE: A new empirical method for estimating the binding affinity of a protein-ligand complex,* J. Mol. Model. 4 (1998) 379-394.
26. Wang R. et al.: *An Extensive Test of 14 Scoring Functions Using the PDBbind Refined, Set of 800 ,Protein-Ligand Complexes,* J. Chem. Inf. Comput. Sci. 44 (2004) 2114-2125.
27. Weiner S. J., et al.: *A New Force Field for Molecular Mechanical Simulation of Nucleic Acids and Proteins,* J.Am. Chem. Soc., 106 (1982) 765-784.

Proceedings of the International Multiconference on
Computer Science and Information Technology pp. 375–388
ISSN 1896-7094
© 2007 PIPS

Integrating Fuzzy Logic and Chaos

Zhong Li and Wolfgang A. Halang

Faculty of Mathematics and Computer Science
FernUniversität in Hagen, Germany
zhong.li@fernuni-hagen.de,
Website: http://www.fernuni-hagen.de/IT/welcome.html

Dedicated to Professor Janusz Kacprzyk for his 60th birthday

Abstract. Motivated by the current studies on the interactions between fuzzy logic and chaos theory, for instance, fuzzy modeling of chaotic systems using Takagi-Sugeno (TS) models, linguistic descriptions of chaotic systems, fuzzy control of chaos, complex fuzzy systems, and a combination of fuzzy control technology and chaos theory for an engineering practice, this survey paper aims to provide some heuristic research achievements and insightful ideas to attract more attention on the topic, interactions or relationship between fuzzy logic and chaos theory, which are related at least within the context of human reasoning and information processing.

1 Introduction

The naissance of fuzzy logic and the prosperity of scientific research on chaos theory occurred almost at the same time in the 1960s, a decade full of confusion, when scientists faced difficulties in dealing with imprecise information and complex dynamics. A set theory and then an infinite-valued logic of Lotfi A. Zadeh were so confusing that they were called fuzzy set theory and fuzzy logic in 1965; a deterministic system found by Edward N. Lorenz in 1963 to have random behaviors was so unusual that it was lately named a chaotic system [1, 2].

Since then, fuzzy set theory and chaos theory have independently developed along their own ways and matured as sciences (although still evolving). They have provided many insights into previously intractable and inherently imprecise or complex nonlinear natural phenomena.

In particular, fuzzy systems technology has achieved its maturity with widespread applications in many industrial, commercial and technical fields, ranging from control, automation, and artificial intelligence to image/signal processing, pattern recognition, and electronic commerce. Chaos, on the other hand, was considered as one of the three monumental discoveries of the twentieth century together with the theory of relativity and quantum mechanics. As a very special nonlinear dynamical phenomenon, chaos has reached its current outstanding status from being merely a scientific curiosity in the mid-1960s to an applicable technology in the late 1990s.

Why do we bring together the two seemingly unrelated concepts to study their interactions and relationships? On the one hand, finding the intrinsic relationship between fuzzy logic and chaos theory is certainly of significant interest and of potential importance. The past twenty years have indeed witnessed some serious explorations of the interactions between fuzzy logic and chaos theory, leading to such research topics as fuzzy modeling of chaotic systems using Takagi-Sugeno models [3], linguistic descriptions of chaotic systems [4, 5], fuzzy control of chaos [6], and a combination of fuzzy control technology and chaos theory for various engineering practices [7]. On the other hand, the reason to study the interactions between fuzzy logic and chaos theory lies in that they are related at least within the context of human reasoning and information processing. In fact, fuzzy logic resembles human approximate reasoning using imprecise and incomplete information with inaccurate and even self-conflicting data to generate reasonable decisions under such uncertain environments, while chaotic dynamics play a key role in human brains for processing massive amounts of information instantly. It is believed that the capability of humans in controlling chaotic dynamics in their brains is more than just an accidental by-product of the brain's complexity, but rather, it could be the chief property that makes the human brain different from any artificial-intelligence machines [8]. It is also believed that to understand the complex information processing within the human brain, fuzzy data and fuzzy logical inference are essential, since precise mathematical descriptions of such models and processes are clearly out of question with today's scientific knowledge.

What is necessary to mention is that Lotfi A. Zadeh has integrated fuzzy logic and chaos theory into the concept of *soft computing* (SC), where he states that SC consists of fuzzy logic (FL), neural network theory (NN) and probabilistic reasoning (PR), with the latter subsuming parts of belief networks, genetic algorithms, chaos theory and learning theory. It is noted that SC is not a melange of FL, NN and PR. Rather, it is an integration in which each of the partners contributes a distinct methodology for addressing problems in their common domain. In this perspective, the principal contributions of FL, NN and PR are complementary rather than competitive.

This paper aims to provide some heuristic research achievements and insightful ideas to attract more attention on the topic, through reviewing the current studies on the interactions between fuzzy logic and chaos theory, including fuzzy modeling of chaotic systems using Takagi-Sugeno (TS) models, linguistic descriptions of chaotic systems, fuzzy control of chaos, complex fuzzy systems, and a combination of fuzzy control technology and chaos theory for an engineering practice.

2 Fuzzy Definition of Chaos

The term *chaos* associated to an interval map was first formally introduced into mathematics by Li and Yorke in 1975 [9], where they established a simple

criterion for the existence of chaos in one-dimensional difference equations, i.e., "period three implies chaos".

However, a definitive, universally accepted, and completely rigorous mathematical definition of chaos is not yet available in the scientific literature to provide a fundamental basis for studying such exotic phenomena. Instead, various alternative, but closely related definitions of chaos have been proposed, along with mechanisms giving rise to such behavior. Among those, the original definition of Li and Yorke and its fine tuning by Devaney seem to be the most popular. These definitions of chaos have been generated to difference equations in $\mathcal{R}^n$, Banach spaces and complete metric spaces. In particular, an application of the definition in complete metric spaces to chaotic dynamics on the metric space $(\mathcal{E}^n, D)$ of fuzzy sets on the base space $\mathcal{R}^n$ may lay a foundation for further study on the interactions between fuzzy logic and chaos theory [10].

It is shown by the result of Li and Yorke that such chaotic behavior could arise in quite simple systems and could be generated by quite simple mechanisms. In particular, the "period three implies chaos" result in scalar difference equations involves *noninvertible, continuous* maps rather than more demanding diffeomorphisms. Consequently, one may ask to what extent the one-dimensional result of Li and Yorke carries over to a higher dimensional difference equation

$$x_{k+1} = f(x_k), \quad k = 0, 1, 2, \ldots, \tag{1}$$

with a continuous map $f : X \to X$, where X is a closed subset of $\mathcal{R}^n$ for $n \geq 2$.

A counterexample shows that the result does not carry over to higher dimensions without some suitable restriction on the class of maps f [11]. To determine a suitable class of maps for which the result of the Li and Yorke might hold in higher dimensions, we found that the one-dimensional maps for which the difference equation (1) have cycles of period three all have graphs with a hump, i.e., which fold over on themselves, namely, they are not one-to-one maps. This suggests that attention might profitably be restricted to maps that are not one-to-one. This was done by Marotto [12] who showed that difference equations on $\mathcal{R}^n$ defined in terms of continuously differentiable maps with *snap-back repellers*, so consequently not one-to-one, behave chaotically in the sense of Li and Yorke. His proof used the inverse function theorem for one-to-one local restrictions of the maps and the Brouwer fixed point theorem, but otherwise paralleled the proof of Li and Yorke for one-dimensional maps. This was extended in 1981 to maps with a saddle point by Kloeden [13] and Shiraiwa and Kurata [14]. Further, the result of Kloeden for the case of saddle points in a finite dimensional Euclidean space $\mathcal{R}^n$ can be easily extended by using the Schauder fixed point theorem to a Banach space. Even more generally, some criteria for chaos of difference equations in general complete metric spaces have been given in [15].

What we concern here is to generalize the Li-Yorke and Marotto definitions to be applicable to maps from a space of fuzzy sets into itself, namely the metric space $(\mathcal{E}^n, D)$ of fuzzy sets on the base space $\mathcal{R}^n$. The result to be given below is essentially an adaptation of the result in a Banach space, which is possible because the metric spaces of fuzzy sets under consideration can be embedded as a cone in a certain Banach space.

Here, we simply give the Kaleva fixed point theorem without giving some basic terminologies [16].

Theorem 1. (Kaleva)
Let $f : \mathcal{E}^n \to \mathcal{E}^n$ be continuous and let $\mathcal{X}$ be a non-empty compact convex subset of $\mathcal{E}^n$ such that $f(\mathcal{X}) \subseteq \mathcal{X}$. Then f has a fixed point $\bar{u} = f(\bar{u}) \in \mathcal{X}$.

Consider an iterative scheme of fuzzy sets

$$u_{k+1} = f(u_k), \quad k = 1, 2, \ldots, \tag{2}$$

where f is a continuous map from the space of fuzzy sets $\mathcal{E}^n$ into itself. Using the Kaleva fixed point theorem, sufficient conditions will be given below for a map on fuzzy sets to be chaotic.

Theorem 2. (Kloeden [17])
Let $f : \mathcal{E}^n \to \mathcal{E}^n$ be continuous and suppose that there exist non-empty compact subsets $\mathcal{A}$ and $\mathcal{B}$ of $\mathcal{E}^n$, and integers $n_1, n_2 \geq 1$ such that

(i) *$\mathcal{A}$ is homeomorphic to a convex subset of $\mathcal{E}^n$,*
(ii) *$\mathcal{A} \subseteq f(\mathcal{A})$,*
(iii) *f is expanding on $\mathcal{A}$, that there exists a constant $\lambda > 1$ such that*

$$\lambda D(u, v) \leq D(f(u), f(v))$$

for all $u, v \in \mathcal{A}$,
(iv) *$\mathcal{B} \subset \mathcal{A}$,*
(v) *$f^{n_1}(\mathcal{B}) \cap \mathcal{A} = \emptyset$,*
(vi) *$\mathcal{A} \subseteq f^{n_1+n_2}(\mathcal{B})$,*
(vii) *$f^{n_1+n_2}$ is one-to-one on $\mathcal{B}$.*

Then the map f is chaotic.

It is noted that difference equations generated by Poincaré section maps provide a link between the dynamics of discrete-time dynamical systems and continuous-time dynamical systems. However, it has been much more difficult to give a mathematically rigorous proof of the existence of chaos in a continuous-time nonlinear autonomous systems. Even one of the classic icons of modern nonlinear dynamics, the Lorenz attractor, now known for 40 years, was not proven rigorously to be chaotic until 1999 by Warwick Tucker of the University of Uppsala in his Ph.D. dissertation [18]. A commonly agreed analytic criterion for proving the existence of chaos in continuous-time systems is based on the fundamental work of Shil'nikov, known as the Shil'nikov method or Shil'nikov criterion [19], whose role is in some sense equivalent to that of the Li-Yorke definition in the discrete setting. The Shil'nikov criterion guarantees that complex dynamics will occur near homoclinicity or heteroclinicity when an inequality (Shil'nikov inequality) is satisfied between the eigenvalues of the linearized flow around the saddle point(s), i.e., if the real eigenvalue is larger in modulus than the real part of the complex eigenvalue. Complex behavior always occurs when the saddle set is a limit cycle.

3 From Chaos to Fuzziness

This section discusses fuzzy modeling of chaotic systems, meaning to transform chaotic systems into fuzzy formulations.

Fuzzy system models basically fall into two categories, which differ fundamentally in their abilities to represent different types of information. The first category includes linguistic models, which have been referred to so far as Mamdani fuzzy models. They are based on collections of IF-THEN rules with vague predicates and use fuzzy reasoning [20, 21]. In these models, fuzzy quantities are associated with linguistic labels, and a fuzzy model is essentially a qualitative expression of the underlying system. Models of this type form a basis for qualitative modeling that describes the system behavior by using natural language [22]. A corresponding fuzzy logic controller is a prototypical example of such a linguistic model, in which its rules give a linguistic expression of the control strategy in a common sense.

The second category of fuzzy models is based on the Takagi-Sugeno (TS) method of reasoning [23–25]. These models are formed by logical rules that have a fuzzy antecedent part and a functional consequent. They are combinations of fuzzy and nonfuzzy models. Fuzzy models based on the TS method of reasoning integrate the ability of linguistic models for qualitative knowledge representation with great potential for expressing quantitative information.

3.1 Fuzzy Modeling of Chaotic Systems based on Mamdani Model

Fuzzy logic allows to model processes in a linguistic manner. The basic configuration of a fuzzy logic system is composed of a fuzzyfier, a fuzzy rule-base, a fuzzy inference engine, and a defuzzyfier, where the fuzzy rule-base consists of a collection of fuzzy IF-THEN rules, and the fuzzy inference engine uses these fuzzy IF-THEN rules to determine a map from fuzzy inputs to fuzzy outputs based on fuzzy composition rules.

A systematic approach for modeling chaotic systems using Mamdani model is never available, Baglio *et al.* [4, 5] have, however, subtly derived Mamdani fuzzy models of some typical chaotic systems. To do so, a good description of chaotic systems is required. The definition of chaotic behaviors involves the three fundamental concepts of transitivity, density of periodic orbits, and sensitivity to initial conditions [26]. Furthermore, from a qualitative point of view, chaos can be defined by monitoring the time evolution of trajectories emanating from nearby points on the attractor. In a chaotic system, points that are close to each other repel themselves so that the flow stretches. Then, a folding action must take place for the chaotic behavior to combine with the boundedness of the attractor. The stretching and folding features of the flow are responsible for the sensitivity to initial conditions, and characterize the chaotic behavior.

To develop a fuzzy model of the evolution of a chaotic signal x, two variables can be considered as inputs, i.e., the center value $x(k)$, which is the nominal value of the state x at the instant k, and the uncertainty $d(k)$ on the center value. In terms of fuzzy description, this means that the model contains four

linguistic variables, i.e., $x(k)$, $x(k+1)$, $d(k)$ and $d(k+1)$. The whole set of rules has to determine the values $x(k+1)$ and $d(k+1)$ from the values $x(k)$ and $d(k)$.

Take the logistic map $x(k+1) = \mu x(k)(1-x(k)),\ \mu = 4$, which shows a single-scroll attractor, as an example to illustrate the modeling procedure. In this single scroll system, x tends to move out from the trivial equilibrium point $x_1^* = 0$ until x begins to oscillate around the nontrivial equilibrium point $x_2^* = 3/4$. The increasing amplitude of the oscillations forces the trajectory to enter again the neighborhood of $x_1^* = 0$, where, due to its instability, the above process repeats. The linguistic variables of the system, $(x(k), x(k+1), d(k), d(k+1))$, take five linguistic values: zero (Z), small (S), medium (M), large (L) and very large (VL). The fuzzy sets associated to these linguistic values are shown in Fig. 1. They are constructed in such a way that the equilibrium point $x_2^* = 3/4$ is between the fuzzy set M and the fuzzy set L.

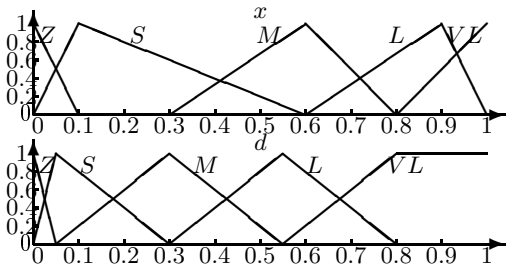


Fig. 1. The fuzzy sets for x (left) and d (right): logistic map

In other words, when x is smaller than the nontrivial equilibrium point $x_2^* = 3/4$, it tends to increase and, when x is very large, it tends to decrease, which can be summarized in the following rules.

$$R_1: \quad \text{IF } x(k) \text{ is } S \text{ THEN } x(k+1) \text{ is } M;$$
$$R_2: \quad \text{IF } x(k) \text{ is } VL \text{ THEN } x(k+1) \text{ is } Z.$$

In this way, a complete set of fuzzy rules to generate a single-scroll chaotic system is summarized in Table 1

The trajectory of the fuzzy system in the phase space is shown in Fig. 2 with the center-of-sums defuzzification method and the product as t-norm, which is obviously similar to that of the original logistic map.

3.2 Fuzzy Modeling of Chaotic Systems based on TS Model

As mentioned above, the TS fuzzy model adopts linear functions rather than fuzzy sets in the consequence part, thus, linearization methods are often used in modeling, where the following theorem can be used to convert the nonlinear terms in the nonlinear systems to weighted linear sums of some linear functions [27].

Table 1. Fuzzy rules implementing a single scroll chaotic system

$x(k)/d(k)$	Z	S	M	L	VL
Z	Z/Z	Z/M	Z/M	S/VL	L/L
S	M/Z	M/M	M/M	M/VL	L/S
M	L/Z	L/M	L/M	L/VL	VL/S
L	M/Z	M/M	M/M	M/VL	Z/S
VL	Z/Z	Z/M	Z/M	Z/VL	Z/L

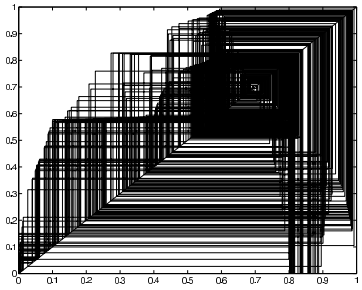


Fig. 2. The logistic map generated by the fuzzy model

Theorem 3. *Consider the following nonlinear term:*

$$f_n = x_1 x_2 \cdots x_n \,, \tag{3}$$

where $x_i \in \left[M_1^i \,, M_2^i\right]$. *Formula (3) can exactly be represented by a linear weighted sum of the form*

$$f_n = \left(\sum_{i_2, i_3, \ldots, i_n = 1}^{2} \mu_{i_2 i_3 \cdots i_n} \cdot g_{i_2 i_3 \cdots i_n} \right) x_1, \tag{4}$$

where

$$g_{i_2 i_3 \cdots i_n} = \prod_{j=2}^{n} M_{i_j}^j \,, \quad \mu_{i_2 i_3 \cdots i_n} = \prod_{j=2}^{n} \Gamma_{i_j}^j \,,$$

in which $\Gamma_{i_j}^j$ *is positive semi-definite for all* $x_j \in [M_1 \,, M_2]$, *defined as follows:*

$$\Gamma_1^j = \frac{-x_j + M_2^j}{M_2^j - M_1^j} \,, \quad \Gamma_2^j = \frac{x_j - M_1^j}{M_2^j - M_1^j} \,.$$

Proof. It can be easily proved using inductive reasoning, thus, the proof is omitted here.

Herewith, for most of chaotic systems, their exact TS fuzzy models can be easily derived to be with only two fuzzy rules. Here, the word "exact" means that the defuzzified output of the TS fuzzy models are mathematically identical to that of the original nonlinear systems.

For instance, the Lorenz equations,

$$\frac{d}{dt}\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} -\sigma x + \sigma y \\ rx - y - xz \\ xy - bz \end{bmatrix}, \tag{5}$$

where $\sigma, r, b > 0$ are parameters, can be expressed as the following TS fuzzy model:

$$\text{Rule 1: IF } x(t) \text{ is about } M_1 \text{ THEN } \frac{d}{dt}\begin{bmatrix} x(t) \\ y(t) \\ z(t) \end{bmatrix} = A_1 \begin{bmatrix} x(t) \\ y(t) \\ z(t) \end{bmatrix}$$

$$\text{Rule 2: IF } x(t) \text{ is about } M_2 \text{ THEN } \frac{d}{dt}\begin{bmatrix} x(t) \\ y(t) \\ z(t) \end{bmatrix} = A_2 \begin{bmatrix} x(t) \\ y(t) \\ z(t) \end{bmatrix}$$

where

$$A_1 = \begin{bmatrix} -\sigma & \sigma & 0 \\ r & -1 & -M_1 \\ 0 & M_1 & -b \end{bmatrix}, \quad A_2 = \begin{bmatrix} -\sigma & \sigma & 0 \\ r & -1 & -M_2 \\ 0 & M_2 & -b \end{bmatrix},$$

and the membership functions are

$$\Gamma_1 = \frac{-x + M_2}{M_2 - M_1}, \quad \Gamma_2 = \frac{x - M_1}{M_2 - M_1},$$

where $\Gamma_i, i = 1, 2$, are positive semi-definite for all $x \in [M_1, \quad M_2]$.

4 From Fuzziness to Chaos

It is clear to see that fuzzy modeling of chaotic systems implies that fuzzy systems can be also chaotic. In contrast to the last section, "from fuzziness to chao" means to make originally stable or non-chaotic fuzzy systems chaotic. This is of practical significance, since chaos can actually be useful under certain circumstances, and there is growing interest in utilizing the very nature of chaos [28–31].

One simple, yet mathematically rigorous control method from the engineering feedback control approach was developed, where a linear state-feedback controller with an uniformly bounded control-gain sequence can be designed to make all Lyapunov exponents of the controlled system strictly positive and arbitrarily assigned. Moreover, such a controller can be designed for an arbitrarily given, n-dimensional dynamical system that could originally be nonchaotic or even asymptotically stable. The goal of chaotification is finally achieved with a simple modulus operation or a sawtooth (or even a sine) function. The design criterion is to use the definition of chaos given by Devaney or Li-Yorke [9],

while for the n-dimensional case the Marotto theorem [12] was used for a proof. For the continuous-time case, a general approach to make an arbitrarily given autonomous system chaotic has also been proposed recently [32–35]. Here, the main tool to use is time-delay feedback perturbation on a system parameter or as an exogenous input [33].

These chaotification techniques can be applied to TS fuzzy systems, where the so-called parallel distributed compensation (PDC) technique is employed to determine the structure of a fuzzy controller [38–43].

It is remarked that modeling of chaotic systems based on Mamdani models discussed above provides a means to generate chaos from Mamdani fuzzy systems in a linguistic manner.

5 Fuzzy Control of Fuzzy Chaotic Systems

For many years, the main feature of chaos, i.e., the extreme sensitivity to initial conditions, made chaos undesirable, and most experimentalists consider such characteristics as something to be strongly avoided [36, 37]. In addition to this feature, chaotic systems have two other important ones. First, there are infinite many unstable periodic orbits embedded in the underlying chaotic attractor, and second, the dynamics in the chaotic attractor is ergodic, which implies that during its temporal evolution the system ergodically visits any small neighborhood of every point in each of the unstable periodic orbits embedded within the chaotic attractor.

Owing to these properties, a fuzzy chaotic system, the fuzzy formulation of a chaotic system, can be seen as shadowing some periodic behavior at a given time, and erratically jumping from one to another periodic orbit. Thus, when a trajectory approaches ergodically a desired periodic orbit embedded in the chaotic attractor, one can apply small perturbations to stabilize such an orbit. Therefore, we can say that the extreme sensitivity of a chaotic system to changes in its initial conditions may be very desirable in practical experimental situations [37]. It suffices to note that, due to chaos, using the same chaotic system one is able to produce infinite many desired dynamical behaviors (either periodic or not periodic) only with properly chosen tiny perturbations. This property is not shared by non-chaotic systems, because the perturbations needed therein for producing a desired behavior must, in general, be of the same order of magnitude as the unperturbed dynamical variables.

Generally, chaos control approaches can be divided into two broad categories: feedback and nonfeedback (or say, open-loop) control approaches. Feedback control methods do not change the controlled systems and stabilize unstable periodic orbits embedded in chaotic attractors, while nonfeedback control methods slightly change the controlled system, mainly by a small tuning of control parameter, changing the system behavior from chaotic attractor to periodic orbit, which is close to the initial attractor.

It is known that the nonfeedback approach is much less flexible, and requires more prior knowledge of motion. To apply such an approach, one does

not have to follow the trajectory. The control can be activated at any time, and one can switch from one periodic orbit to another without returning into the chaotic behavior. This approach can be very useful in mechanical systems, where the feedback control systems are often very large (sometimes larger than the system controlled). The extremely simple, easily implementable, low-cost, and reliable nonfeedback approaches are widely applied in many physical experiments and industrial processes today, particularly for nonlinear dynamical systems, as a unified feedback control approach has not been fully established for general nonlinear dynamical systems. Roughly speaking, the nonfeedback approaches include the entrainment and migration control method [44–46], control through external forcing [47, 48], while the feedback approaches include the Ott-Grebogi-Yorke (OGY) method, engineering feedback control method, and Pyragas's time-delayed feedback control method [49–51]. In addition, chaos control approaches include conventional linear and nonlinear control, adaptive control, neural networks-based control, fuzzy control, and another very important topic, synchronization of chaos.

6 Fuzzy-Chaos-based Applications: An Example

Combination of fuzzy logic and chaos theory may provide a new means in engineering practice. An example is a fuzzy-model-based chaotic cryptosystem introduced in [7, 52–54].

Cryptography concerns the ways in which communications and data can be encoded to prevent disclosure of their contents through eavesdropping or message interception, using codes, ciphers, or other methods, so that only certain people can see the real messages. So far, varieties of cryptographic methods have been proposed to secure Internet communication. For instance, the Data Encryption Standard (DES) is adopted as a U.S. Federal Information Processing Standard for encrypting unclassified information. Others include IDEA (International Data Encryption Algorithm), and RSA (developed by Rivest, Shamir and Adleman). These encryption algorithms are based on number theory. However, none of them is absolutely secure. Therefore, some emerging theories, such as chaos theory, are always desirable to be adopted to strengthen existing cryptography. The reason of applying chaos theory in cryptography lies in its intrinsic essential properties, such as sensitivity to initial conditions (or control parameters) and ergodicity, which meet Shannon's requirements of confusion and diffusion for cryptography. In addition, chaotic signals are typically broadband, noise-like, and difficult to predict. Therefore, they can be used in various context for masking information-bearing waveforms. They can also be used as modulating waveforms in spread-spectrum systems. The idea of chaotic masking is to directly add the message in a noise-like chaotic signal at the end of the transmitter, while chaotic modulation is by injecting the message into a chaotic system as a spread-spectrum transmission. Later, at the receiver, a coherent detector with some signal processing is employed to recover the message. But the signal masking or parameter modulation approach to chaotic communication

only provides a lower level of security as stated in [55]. Using basic cryptosystem theory, a fuzzy-model-based chaotic cryptosystem has been proposed to provide a methodology with a higher level of security. There, Luré type discrete-time chaotic systems are first exactly represented by TS fuzzy models. Then, a superincreasing sequence is generated by using a chaotic signal, which can flexibly be used as an output of the TS fuzzy chaotic drive system, or any state in which the synchronization error approaches zero. In terms of a cryptosystem, the plaintext (message) is encrypted using the superincreasing sequence at the drive system side, which results in the ciphertext. The ciphertext may be added to the output or state of the drive system using the methodologies proposed in [7, 52]. Further, this the ciphertext embedding scalar signal is sent to the response system end. Following the design of a response system, the chaotic synchronization between a drive and a response system is achieved by solving LMIs. By the synchronization, one can regenerate the same superincreasing sequence and recover the ciphertext at the response system end. Finally, using the regenerated superincreasing sequence, the ciphertext is decrypted into the plaintext. The block diagram of the whole cryptosystem is shown in Fig. 3.

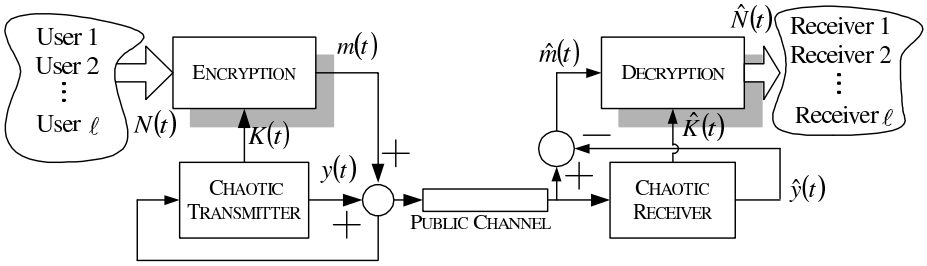


Fig. 3. Block diagram of chaotic encryption methodology

7 Conclusions

This paper has reviewed the current studies on the interactions between fuzzy logic and chaos theory, from the following aspects: fuzzy modeling of chaotic systems using Takagi-Sugeno models, linguistic descriptions of chaotic systems, fuzzy control of chaos, complex fuzzy systems, and a combination of fuzzy control technology and chaos theory for various engineering practices. What needs to emphasize is that in spite of the efforts on exploring the interactions between fuzzy logic and chaos theory, it is still far away from fully understanding their mutual relationships. Although this paper may not give insight into their relations or may raise more questions than it can provide answers, we hope that it nevertheless contains seeds for future brooming research.

References

1. Z. Li, W. Halang, and G. Chen, *Integration of Fuzzy Logic and Chaos Theory*, Springer-Verlag: Heidelberg, 2006.
2. Z. Li, *Fuzzy Chaotic Systems – Modeling, Control, and Applications*, Springer-Verlag: Heidelberg, 2006.
3. H.O. Wang, K. Kanaka and T. Ideda, Fuzzy modeling and control of chaotic systems, *Proc. 1996 IEEE Int. Symp. on Circ. Syst., ISCAS'96*, pp. 209–212.
4. S. Baglio, L. Fortuna and G. Manganaro, Design of fuzzy iterators to generate chaotic time series with assigned Lyapunov exponent, *Electronics Letters*, Vol. 32, No. 4, 1996, pp. 292–293.
5. M. Porto and P. Amato, A fuzzy approach for modeling chaotic dynamics with assigned properties, *The 9th IEEE Int. Conf. on Fuzzy Systems, 2000, FUZZ-IEEE 2000*, Vol. 1, pp. 435–440.
6. O. Calvo and J. L. Cartwright, Fuzzy control of chaos, *Int. J. Bifurc. Chaos*, Vol. 8, No. 8, 1998, pp. 1743–1747.
7. K.-Y. Lian, T.-S. Chiang, C.-S. Chiu and P. Liu, Synthesis of fuzzy model-based designs to synchronization and secure communications for chaotic systems, *IEEE Trans. Syst., Man, and Cybern. – Part B: Cybernetics*, Vol. 31, No. 1, pp. 66–83, 2001.
8. L. Glass and M. C. Mackey, *From Clocks To Chaos*, Princeton University Press, 1988.
9. T. Y. Li and J. A. Yorke, Period three implies chaos, *Amer. Math. Monthly*, Vol. 82, pp. 481–485, 1975.
10. P. Kloeden and Z. Li, Li-Yorke chaos in higher dimensions: a review, *Journal Difference Equations and Applications*, the special issue in Memory of Professor Bernd Aulbach, Vol. 12, No. 3–4, pp. 247–279, 2006.
11. P. E. Kloeden, Cycles and chaos in higher dimensional difference equations, Yu. A. Mitropolsky (Ed.), *Proc. 9th Int. Conf. Nonlinear Oscillations*, Vol. 2, pp.184–187, (Kiev Naukova Dumka) 1984.
12. F. R. Marotto, Snap-back repellers imply chaos in $\mathbf{R}^n$, *J. of Math. Anal. Appl.*, Vol.63, pp. 199–223, 1978.
13. P. E. Kloeden, Chaotic difference equations in $\mathbf{R}^n$, *J. Austral. Math. Soc. (Series A)*, Vol. 31, pp. 217–225, 1981.
14. K. Shiraiwa and M. Kurata A generalization of a theorem of Marroto, *Nagoya Math J.*, **82**, pp. 83–97, 1981.
15. Y. M. Shi and G. Chen, Chaos of discrete dynamical systems in complete metric spaces, *Chaos, Solitons and Fractals*, Vol. 22, pp. 555–571, Nov. 2004.
16. O. Kaleva, On the convergence of fuzzy sets, *Fuzzy Sets and Systems*, Vol. 17, pp. 53–65, 1985.
17. P. E. Kloeden, Chaotic iterations of fuzzy sets, *Fuzzy Sets and Systems*, Vol. 42, No. 1, pp. 37–42, 1991.
18. W. Tucker, The Lorenz attractor exists, *C. R. Acad. Sci. Paris*, Vol. 328, pp. 1197–1202, 1999.
19. C. P. Silva, Shil'nikov theorem—a tutorial, *IEEE Trans. Circuits Syst.-I*, Vol. 40, pp. 675–682, 1993.
20. R. M. Tong, The construction and evaluation of fuzzy models, in *Advances in Fuzzy Sets Theory and Applications*, M. M. Gupta, R. K. Ragade and R. R. Yager (eds.), North-Holland: Amsterdam, 1979, pp. 559–576.
21. W. Pedrycz, *Fuzzy Control and Fuzzy Systems*, Wiley: New York, 1989.

22. M. Sugeno and T. Yasukawa, A fuzzy-logic-based approach to qualitative modeling, *IEEE Trans. Fuzzy Systems*, Vol. 1, 1993, pp. 7–31.
23. T. Takagi and M. Sugeno, Derivation of fuzzy control rules from human operators actions, *Proc. IFAC Symposium on Fuzzy Information*, Marseille, 1983, pp. 55–60.
24. T. Takagi and M. Sugeno, Fuzzy identification of systems and its application to modeling and contorl, *IEEE Trans. Syst., Man, and Cyber.*, Vol. 15, pp. 116–132, 1985.
25. M. Sugeno and G. T. Kang, Fuzzy modeling and control of multilayer incinerator, *Fuzzy Sets and Systems*, Vol. 18, 1986, pp. 329–346.
26. H.-O. Peitgen, H. Jürgens and D. Saupe, *Chaos and fractals – New frontiers of science*, 2nd ed., Springer, 2004.
27. H. J. Lee, *Robust Fuzzy-Model-Based Controller Design for Nonlinear Systems with Parametric Uncertainties*, Master thesis, Yonsei University, Seoul, Korea, 2000.
28. G. Chen and X. Dong, *From Chaos to Order: Methodologies, Perspectives and Applications*, World Scientific: Singapore, 1998.
29. G. Chen, Chaos: Control and anticontrol, *IEEE Circuits Syst. Soc. Newsletter*, Vol. 9, No. 1, pp. 1–5, 1998.
30. S. J. Schiff, K. Jerger, D. H. Duong, T. Chang, M. L. Spano and W. L. Ditto, Controlling chaos in the brain, *Nature*, Vol. 370, pp. 615–620, 1994.
31. W. Yang, M. Ding, A. J. Mandell and E. Ott, Preserving chaos: Control strategies to preserve complex dynamics with potential relevance to biological disorders, *Phys. Rev. E*, Vol. 51, pp. 102–110, 1995.
32. X. F. Wang and G. Chen, Chaotifying a stable LTI system by tiny feedback control, *IEEE Trans. Circuits and Systems-I*, Vol. 47, No. 3, pp. 410–415, 2000.
33. X. F. Wang, G. Chen and X. Yu, Anticontrol of chaos in continuous-time systems via time-delay feedback, *Chaos*, Vol. 10, No. 4, pp. 1–9, 2000.
34. T. S. Zhou, G. Chen and Q. G. Yang, A simple time-delay feedback anticontrol method made rigorous, *Chaos*, Vol. 14, pp. 662–668, 2004.
35. H. G. Zhang, Z. L. Wang and D. R. Liu, Chaotifying fuzzy hyperbolic model using adaptive inverse optimal control approach, *Intl. J. of Bifurcation and Chaos*, Vol. 14, No. 10, pp. 3505–3517, 2004.
36. T. Kapitaniak, *Controlling chaos – Theoretical and practical methods in nonlinear dynamics*, Academic Press, 1996.
37. S. Boccaletti, C. Grebogi, Y.-C. Lai, H. Mancini and D. Maza, The control of chaos: theory and applications, *Physics reports*, Vol. 39, pp. 103–197, 2000.
38. K. Tanaka, T. Ikeda and H.O. Wang, A unified approach to controlling chaos via an LMI-based fuzzy control system design, *IEEE Trans. Circ. Syst.—I*, Vol. 45, No. 10, pp. 1021–1040, 1998.
39. H. Wang, K. Tanaka and M. Griffin, Parallel distributed compensation of nonlinear systems by Takagi and Sugeno's fuzzy model, *Proc. FUZZ-IEEE'95*, pp. 531–538, 1995.
40. H. Wang, K. Tanaka and M. Griffin, An approach to fuzzy control of nonlinear systems: stability and design issues, *IEEE Trans. Fuzzy Syst.*, Vol. 4, pp. 14–23, 1996.
41. Z. Li, J. B. Park, G. Chen, Y. H. Joo and Y. H. Choi, Generating chaos via feedback control from a stable TS fuzzy system through a sinusoidal non-linearity, *Int. J. Bifur. Chaos*, Vol. 12, No. 10, pp. 2283–2291, 2002.
42. Z. Li, J. B. Park, Y. H. Joo, G. Chen and Y. H. Choi, Anticontrol of chaos for discrete TS fuzzy systems, *IEEE Trans. Circ. Syst.-I*, Vol. 49, No. 2, pp. 249–253, 2002.

43. Z. Li, J. B. Park and Y. H. Joo, Chaotifying continuous-time TS fuzzy systems via discretization, *IEEE Trans. Circ. Syst.-I*, Vol. 48, No. 10, pp. 1237–1243, 2001.
44. E. A. Jackson, The entrainment and migration controls of multiple attractor systems, *Phys. Lett. A*, Vol. 151, pp. 478–484, 1990.
45. E. A. Jackson, On the control of complex dynamic systems, *Physica D*, Vol. 50, pp. 341–366, 1991.
46. E. A. Jackson, *Perspectives of nonlinear dynamics*, Vol. 1 and 2, Cambridge University Press, New York, 2nd ed., 1991.
47. Y. Braiman and I. Goldhirsch, Taming chaotic dynamics with weak periodic perturbations, *Phys. Rev. Lett.*, Vol. 66, pp. 2545–2548, 1991.
48. R. Chacón, Geometrical resonance as a chaos eliminating mechanism, *Phys. Rev. Lett.*, Vol. 77, pp. 482–485, 1996.
49. K. Pyragas, Continuous control of chaos by self-controlling feedback, *Phys. Lett. A*, Vol. 170, pp. 421–428, 1992.
50. K. Pyragas, Control of chaos via an unstable delayed feedback controller, *Phys. Rev. Lett.*, Vol. 86, pp. 2265–2268, 2001.
51. J. E. S. Socolar, D. W. Sukow and D. J. Gauthier, Stabilizing unstable periodic orbits in fast dynamical systems, *Phys. Rev. E*, Vol. 50, pp. 3245–3248, 1994.
52. K.-Y. Lian, C.-S. Chiu, T.-S. Chiang and P. Liu, LMI-based fuzzy chaotic synchronization and communiations, *IEEE Trans. Fuzzy Systems*, Vol. 9, No. 4, pp. 539–553, 2001.
53. K.-Y. Lian, T.-S. Chiang, P. Liu, and C.-S. Chiu, Synchronization and secure communication for chaotic systems: the modulated fuzzy system design, *Int. J. Bifur. Chaos*, Vol. 11, No. 5, pp. 1397–1410, 2001.
54. K.-Y. Lian, T.-S. Chiang, C.-S. Chiu and P. Liu, Synthesis of fuzzy model-based designs to synchronization and secure communications for chaotic systems, *IEEE Trans. Syst., Man, and Cybern. – Part B: Cybernetics*, Vol. 31, No. 1, pp. 66–83, 2001.
55. K. Short, Steps toward unmasking secure communications, *Int. J. Bifurcation Chaos*, Vol. 4, pp. 959–977, 1994.

Proceedings of the International Multiconference on
Computer Science and Information Technology, pp. 389 – 398
ISSN 1896-7094
© 2007 PIPS

Automatic Form Filling Based on Ontology-Controlled Dialogue With the User

Łukasz Bownik, Wojciech Górka, Adam Piasecki

Centrum Elektryfikacji i Automatyzacji Górnictwa "EMAG"
ul.Leopolda 31, 40-189 Katowice
{lbownik, wgorka, apiasecki}@emag.pl

Abstract. Semantic Web is now one of the most common directions in the IT-oriented research. Here, the focus is put on issues related to the logical layer of semantic applications, i.e. inference methods within constantly broadening ranges of logic as well as standardization of successive languages with increasing logical expressiveness. The article features the solution to the issue of automatic form filling with the use of ontology-controlled dialogue with the user. This solution resulted in the implementation of a universal programming module. Practical application of semantic technologies based on previous achievements in the field, combining the possibilities of inference machines with the flexibility of the RDF language, allows to achieve certain benefits for the users of an IT system. The Authors assume that the reader possesses some basic knowledge in the range of ontology structure as well as the RDF and OWL languages.

1 Introduction

The basic issue connected with the initiation of public services is the necessity to fill many forms in which certain data are repeated. Most forms require to provide the same data (similar data sets or data dependent on other data included in other parts of the form) whose filling is connected with mental discomfort for the user. The user has to enter the same things again and again. The work described in this paper aimed at developing software that would support the user in the process of filling e-forms through automatic and pseudo-intelligent filling the fields of a form with the information possessed by an IT system. Basic assumptions were the following:

- Modular structure of the software which will enable integration with any IT system,
- Full independence from the domain the form refers to so the software wouldn't include any assumptions related to the meaning and format of the processed data;
- Minimum number of interactions with the user which is both the objective of the work and the qualitative measure of the developed solution.

The result of the conducted research is software in the form of a software library which supports the filling of e-forms based on the dialogue with the user. In the further part of the paper the Authors will show the results of their work, the description of the architecture and the functioning of the developed software, henceforth called a “dialogue module”.

Due to a highly interactive nature of the dialogue module and the limited space of the paper it was not possible to illustrate the functioning of the dialogue module by means of printed screens from the application which uses the dialogue module. Thus in order to demonstrate how the software works the Authors described an example of a user-system interaction while filling a form.

2 Related work

Similar solutions were proposed, among others, in the "RoboForm" [9] and "Mojo Navigator" [10] project but they focus, first of all, on the functionality of passwords wallet integrated with a web browser and having at its disposal the history of values associated with the form fields available on websites. There are also many solutions incorporating a dialog-based method of form filling. However, most of them are strictly based on a hard coded dialogue related to a single concrete form [11].

3 Ontologies

In order to understand the operations of the dialogue module it is necessary to grasp the data ontology and the form ontology ideas which lie at the basis of the process.

3.1 Data Ontologies

Data ontologies define the vocabulary (concepts) used in the course of data exchange between systems. Data ontology is not meant to organize knowledge (which is the case of expert systems) but to determine concepts describing the data which can be understood by many systems. In terms of their concept, data ontologies correspond with XML diagrams and are their functional equivalents.

The first and highest level (Fig.1) contains definitions of the most general vocabulary (concepts) commonly used by many IT systems (e.g. *person, age, name, family name*). The concepts defined at this level are the basis for the definitions of more specialized concepts placed at lower levels of abstraction. **General ontologies should not (must not, to be precise) impose any restrictions** (e.g. in the form of limits as to the format of field value or number of fields in the form) on the defined concepts because each restriction placed at this level of generalization will diminish the universal force of the defined vocabulary and will raise the probability of conflicts with restrictions defined at lower levels (redundant or contrary restrictions). General ontologies must have total horizontal coherence, i.e. mutual coherence within the entire level of abstraction.

The second, middle level contains definitions of concepts shared by IT systems which operate within a given domain (e.g. medicine). These ontologies should use, to the highest possible extent, the concepts defined at a higher level of abstraction (i.e. they should not define their own concepts indicating, for example, *family name*) as well as add the concepts from this very domain (e.g. *clinic, doctor, vaccine*). At this

level of abstraction **only most obvious restrictions** should be defined, i.e. the restrictions which will be forced by all existing and future IT systems operating within a given domain. If it is not possible to explicitly define such restrictions, one should not define them at all because, as it is the case with general ontologies, they may provoke conflicts between ontologies. Domain ontologies must have horizontal coherence within a certain domain, i.e. mutual coherence within each domain at this level of abstraction as well as coherence with general ontologies.

The third and lowest level contains definitions of forms, i.e. specific data packages exchanged between IT systems. A form is a structure of data exchanged between two IT systems. The structure may, but does not have to, be related to the form filled by the system user. Each form there should have a form ontology describing its structure. The ontologies of the form should use, to the highest possible extent, the concepts defined at higher levels of abstraction and **define only such new concepts which are specific to a given form** and cannot be shared with other forms. The form ontology should define all necessary restrictions which will ensure logical coherence and correctness of the form required by the specifications of the IT system. Since for each form there are different rules (restrictions) which determine its coherence and correctness (e.g. one form requires a ID number, the other does not), it is not advisable to define restrictions at higher levels of abstraction. The form ontologies do not have to demonstrate any horizontal coherence (it is practically impossible to provide it). A logical error resulting from horizontal incoherence will never occur because there is separate processing of each ontology in each processing path in the IT system. The form ontologies must be coherent with the applied domain ontologies and general ontologies.

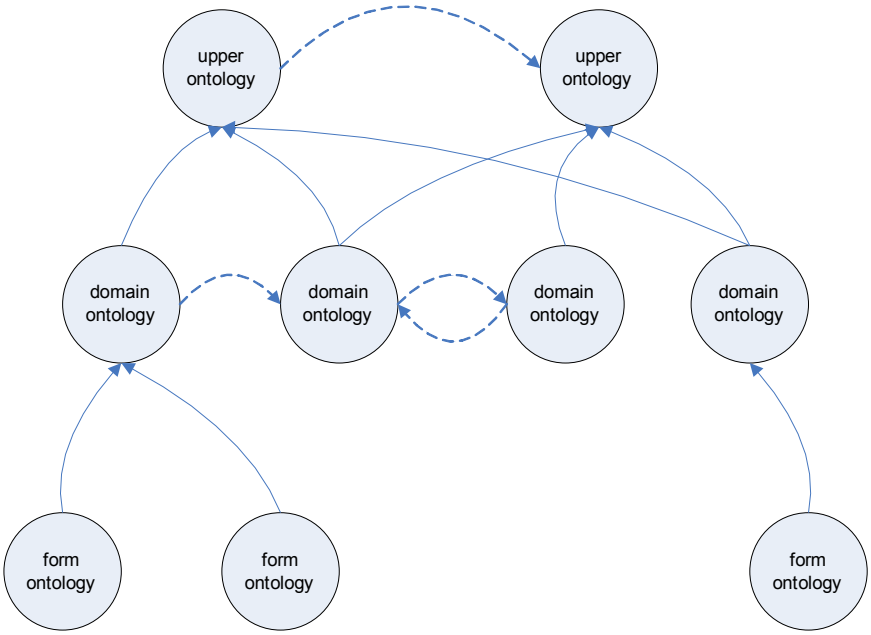


Fig. 1. Layered-tree ontology structure.

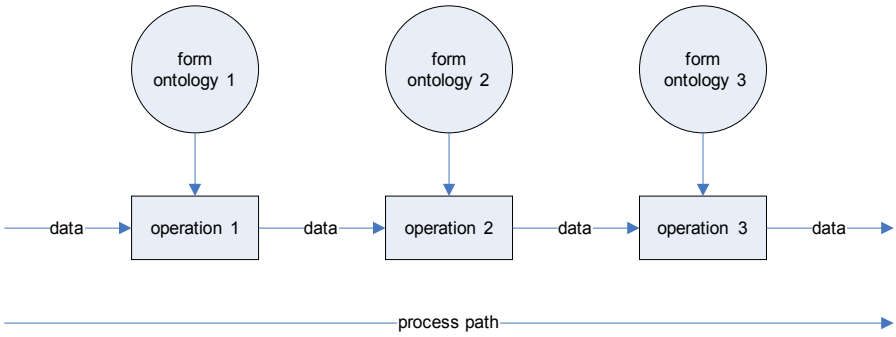


Fig. 2 . The overview of phisical separation of processing form ontologies within single processing path.

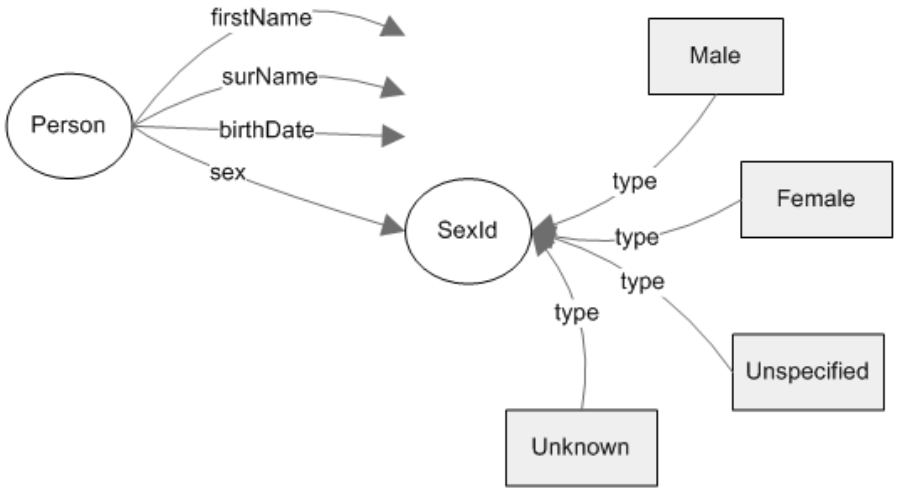


Fig. 3. Example of data ontology.

3.2 Form Ontologies

Form ontologies define the structure as well as coherence and correctness rules of data contained in the described form. A form ontology describes a specific set of data exchanged between IT systems.

The process of constructing a form ontology begins from building the main form class which will lie at the roots of the whole structure. The main class of the form must be a subclass of the Form class. Thanks to this assumption, it is possible to automatically detect the root of the form structure by searching the most specialized subclass of the Form class. The definition of the successive sections of the form is carried out by defining separate classes (e.g. Person class or Address class) linked with object properties to form a tree structure. The definitions of data fields are carried out with the use of datatype properties of classes which make up the sections of the form.

As the form ontology is based on concepts defined in data ontologies which may be much more extensive than the form requires, the *required* and *optional* properties were defined. They connect the desired properties with the classes being the sections of the form (see Fig. 4). In order to force the order of processing the sections and fields of the form, it is necessary to define the processing order for each pair of properties, representing these sections, by means of the *askBefore* property (see Fig. 4).

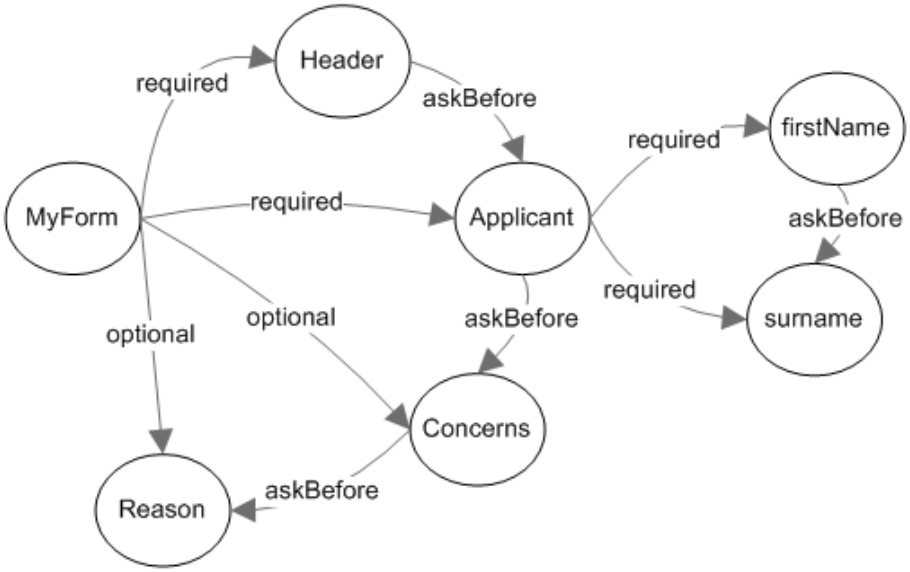


Fig. 4 . A peace of example form ontology.

In order to define the desired restrictions it is necessary to determine new concepts due to the fact that the built-in cardinality concepts have their own defined semantics which does not suit the issue to be solved (first of all, the Open World Assumption, explained further in the article, and no possibility to mark the property as optional).

Form ontologies can contain rules (implications) that extend the functionality of the OWL language [4]. In order to secure logical non-contradiction, the rules must belong to the "DL-safe" set [6, 8]. The most frequent application of the rules is the structure of section labels (enabling, for example, creation of a section label for personal data on the basis of the entered name and surname) and automatic assigning of values to certain fields based on the values of other fields.

4 Character istics of the Dialogue Module

4.1 The Dialogue Module Functionality

The dialogue module works as an information filter. Having a set of input data (which can be empty) and a form ontology, the module "tries" to fill the form with the input data and the data provided by the user. Filling the form is done as a dialogue with the user through minimum interactions with the user assuming that the user is asked questions only if the data are unavailable or doubtful. The usability of the module, measured based on the number of interactions with the user (the fewer the better),

increases proportionally to the volume of input data available in the system. An example of the dialogue functioning will be described in the further part of the module.

4.2 Dialogue module architecture

The dialogue module consists of four basic parts:

- the RDF base enabling access to data and ontologies (now Jena `http://jena.sourceforge.net/`),
- the OWL-Full inference machine responsible for the classification of concepts (now Pellet `http://pellet.owldl.com/`);
- a classic forward chaining inference engine which processes the definitions of extra rules of the form (now Jena);
- a module that controls the form filling process developed as a result of the conducted work (implemented in Java).

The figure below presents a layer structure of the dialogue module. For the module to function properly it is also necessary to have input data stored in the user's profile which was not shown in the figure.



Fig. 5. Dialogue module structure

The efficiency of the dialogue module depends on input data stored in the user's profile. In order to maximize the modular nature it was assumed that the input data are provided as an RDF graph in the process of initializing the dialogue module irrespective of the way they are stored in the system.

4.3 The Dialogue Module Operating Principle

The operating principle of the module is based on the fact that the forms make up a tree structure in which successive embedded sections make branches while fields – leaves of the tree. Thus it is possible to process a form with the use of tree-searching algorithms. To fulfil this task the depth-first search algorithm was selected. This algorithm is the most compliant with the way people fill real forms. The functioning principle of the dialogue module is presented below in a simplified way. While analyzing successive steps one should remember about the iteration-recurrent nature of the algorithm.

1. At the beginning of each dialogue , the dialogue module detects a proper form class in the form ontology and creates its instance.
2. The newly created instance becomes a currently processed node and, at the same time, the top of the tree.
3. For the currently processed node, all required and optional properties are f ound and sorted according to the arrangement determined in the ontology. Then the successively found properties are processed.
4. If the currently processed property points at a section of the form, the section is then processed recursively.
5. In the course of processing the successive properties of the node, the dialogue module tries to detect , in the set of input data, the best adjustments both for the single fields of the form and for the whole sections. The dialogue module asks the user questions only in situations when the desired values do not exist or when it is possible to adjust more than one value to one field or section.

In order to illustrate the functioning of the dialogue module let us check how the module works on a simple form. Assuming that it is necessary to fill the form as follows:

Form:
- Personal data of the user:
  - Name;
  - Surname;
  - Address:
    - City;
    - Street;
    - House No;
- Aim of submitting the form;

If the process of filling the form is invoked for the first time, the dialogue with the user will look as follows:

System: Enter "Name".
User: John
S: Enter "Surname".
U: Brown
S: Enter "City".
U: London
S: Enter "Street".
U: Portland Place
S: Enter "House No".
U: 47
S: Enter "Aim of submitting the form".
U: Application for the permission to use a company car

After filling and accepting the form, the data from the form are stored in the user's profile. At the next interaction between the user and the module, the dialogue will look as follows:

S: I have found personal data for "John Brown" Shall I apply them (Y/N)?
U: Y
S: Enter "Aim of submitting the form".
U: Application to access financial data.

As one can see in the above successive filling of the form, the number of necessary interactions with the user is 6 times smaller. The reduction was possible due to the fact that the dialogue module was able to automatically fill the fields with personal data. Asking the question in the field "Aim of submitting the form" results from the updating policy of the user's profile – this issue, however, is beyond the scope of the paper. Please note that the functioning of the module is based on the real meaning of data defined in an ontology which allows to move data between different forms using common concepts defined in the shared data ontologies.

5 Conclusions

The use of semantic technologies allowed to implement the working software which supports the user in filling any kinds of e-forms. As both R&D and implementation works are underway the usability of the applied solution is assessed more subjectively, based on ergonomics and interaction with some forms, and is not yet supported by statistical data. However, the results obtained up until now show that the direction of the conducted works should be successful.

It is worth mentioning that the solution to the discussed problem cannot be achieved with the use of classic IT solutions which are the basis of most of today's IT systems (relational data bases, XML) and shows the potential which lies in semantic techniques.

References

1. Berners-Lee T., Hendler J., Lassila O., *The Semantic Web*, Scientific American http://www.sciam.com/article.cfm?articleID=00048144-10D2-1C70-84A9809EC588EF21
2. W3C, *Resource Description Framework,* http://www.w3.org/RDF
3. W3C, *OWL Web Ontology Language* , http://www.w3.org/TR/owl-features/
4. Horrocks I., Patel-Schneider P. F., Boley H., Tabet S., Grosof B., Dean M., *SWRL: A Semantic Web Rule Language Combining OWL and RuleML* , http://www.w3.org/Submission/SWRL/
5. Sirin E., Parsia B., Grau B. C., Kalyanpur A., Katz Y., *Pellet: A Practical OWL-DL Reasoner* , http://pellet.owldl.com/papers/sirin07pellet.pdf
6. Parsia B., Sirin E., Grau B. C., Ruckhaus E., Hewlett D., *Cautiously Approaching SWRL* , http://www.mindswap.org/papers/CautiousSWRL.pdf
7. HP Labs, *Jena – A Semantic Web Framework for Java* , http://jena.sourceforge.net/
8. Horrocks I., Parsia B., Patel-Schneider P., Hendler J., *Semantic web architecture: Stack or two towers?* , http://www.cs.man.ac.uk/~horrocks/Publications/download/2005/HPPH05.pdf
9. RoboForm - http://www.roboform.com
10. Mojo Navigator - http://www.mozilla.org/projects/ui/communicator/browser/formfill/
11. http://gospodarka.gazeta.pl/podatki/1,25044,3963883.html

Proceedings of the International Multiconference on
Computer Science and Information Technology, pp. 399 – 407
ISSN 1896-7094
© 2007 PIPS

Genetic Techniques in Modeling Visual Aspects of Non-Natural 3D Objects

Elżbieta Hudyma and Marcin Koszów

Institute of Applied Informatics,
Wrocław University of Technology

Abstract. This paper introduces the idea of using nature-based genetic algorithms to evolve visual aspects of a class of non-natural 3D objects. This pioneer designing approach is presented on example of car models. The problem definition is introduced in genetic algorithms terms and the results of a prototype system are presented. Genetic operators allows users to interactively cross the car models and mutate them. The implemented genetic algorithm is able to generate the models that fits to the provided profile. Alternatively it can be used as a random generator – for creating inspirational car models. More experienced users can directly manipulate the object's genes. The system renders the effects in a real time and allows to watch the objects in a virtual 3D space. Presented approach can be used as well for other class of customizable objects, for example: clothes, furniture, pieces of architecture, etc.

1 Introduction

Generating complex objects with classical 3D modeling systems requires much experience and time. Modeling objects of the same class usually consists of similar steps. Applying changes or adjusting the model often requires repeating all the phases.

The goal of this work is to simplify the modeling process with use of genetic techniques on an example of car models. In the provided prototype system, information needed for object construction is stored in the genotype. The resulting model is built automatically and rendered in a real-time.

Automatic algorithm is capable of creating objects based on a profile sketch given by the user. Genetic operators used in the algorithm are exposed as a set of interactive tools that can be used for crossing and mutating objects. Using them seems to be the most efficient and interesting method of modeling. The gene values can be also accessed directly through the user interface.

This paper also indicates the issues that occur with this approach to modeling. Most significant of these are: genotype structure and domain limitations.

2 Related Work

There are many systems which use genetic algorithms to evolve 3D natural objects. Plant genotype is usually represented in a tree-like structure. There is a great choice of applications based on L-systems (for example in [2]). Advanced systems may use different types of nodes (e.g. leafs, horns, branches, fruits [4]). For this kind of representation, Constructive Solid Geometry seems to be the best way of generating visualizations. Similar, hierarchical structures are also successfully used for modeling animal skeletons. In some solutions the genotype is also responsible for coding neural network which determines the behavior of the object [3]. Depending on the system, Constructive Solid Geometry or Skinning methods are used to generate the result. Human model have a fixed structure so there is no need to store it in the genotype. Genes can contains parameters of the bones (i.e. finger length) and skin deformations (i.e. eyebrows height, muscle size).

Car models and most of man-made classes of objects usually consist of different parts, however the composition of these parts is unchangeable in most cases. Therefore, the genotype should contain properties of the parts and parameters of their dependencies (e.g. relative position).

3 System Overview and Results

The main goal of the spoken system is to provide a set of interactive car modeling tools based on genetic techniques. The design process can be described as a human-based genetic algorithm. The user is an innovator, selector and in fact the executor of each step of the algorithm. He is allowed to use any of the genetic operators on any models from the population. In this mode, the user's likes can be considered the fitness function. The automatic evolution is possible as well, however the user is allowed to step in any time. Algorithm phases and possible interaction points are discussed below.

3.1 Genotype and Graphical Representation

The idea of genotype structure is based on the technical specification of car models. The object is described by 6 cross-sections [Fig.1 - below]. Each cross-section has 4 customizable levels plus one constant floor level [Fig.2 - below]. There are additional genes for wheels parameters. All values are real numbers. Together there are 59 genes.

The result is created with boundary representation – polygon mesh [1]. At the first stage the mesh is created over the cross-sections. Points between the cross-sections are linearly interpolated. The wheels are added next and the mesh is deformed to create fenders. The example models are shown on Figure 3 - below.

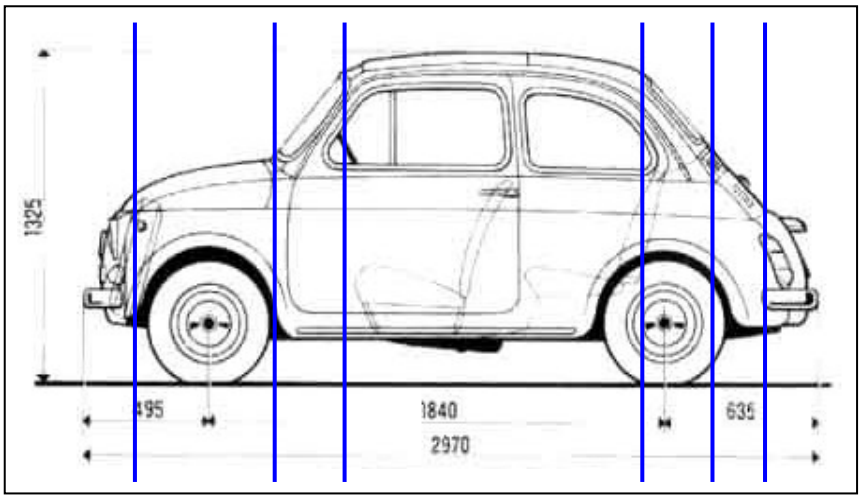


Fig. 1. Six cross-sections of a car model: cowl front, cowl end, cabin front, cabin end, trunk front, trunk end.

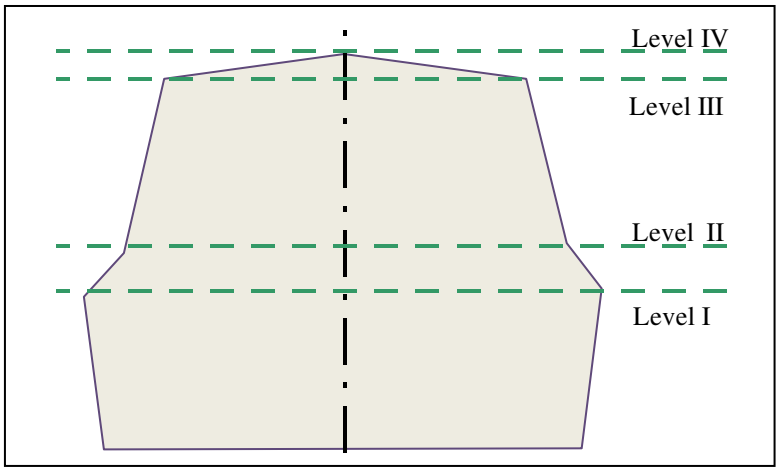


Fig. 2. Cross-section : four customizable levels + constant ground level.

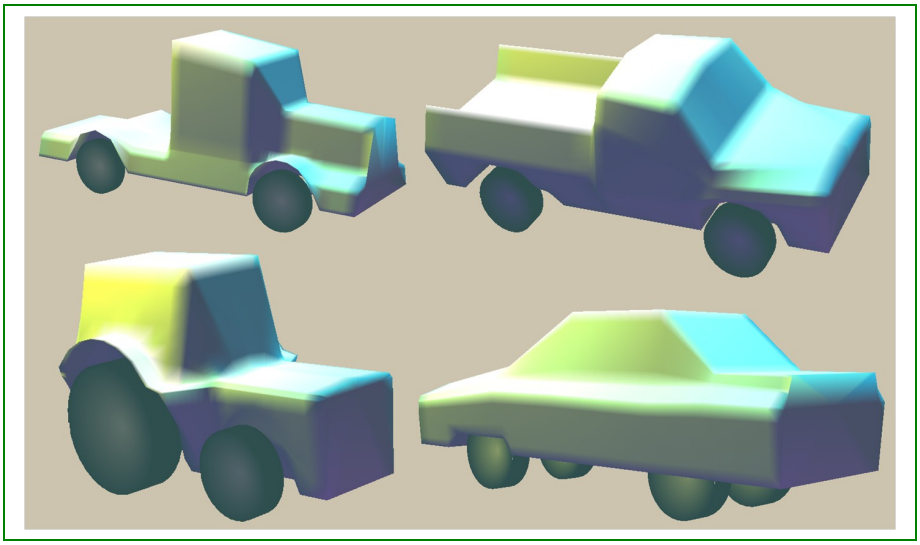


Fig. 3. Example car models designed with genes manipulation.

The genotype is strongly associated to a graphical representation type. For example, besides polygon-mesh used in this project, we also considered Metaballs and Constructive Solid Geometry representations – each of these would require different gene-coding approach.

Genes are the receipt how to create the object and the representation is the fabric. However, even the same genotype, depending on interpretation, can produce different objects, for example when we use Bézier curve point interpolation (instead of linear) when creating the mesh. Thus the module responsible for transforming genotype into the model is the key part of the system. The object is defined by both: genotype and interpreter.

The problem, when manipulating genes directly, are complex relations between genes and esthetical aspects (phenotype). One aspect can be dependent on many genes (e.g. cabin height depends on two cross-section levels) and one gene can affect many aspects (e.g. cowl length and front window angle). This requires the user to know how the genotype is coded.

3.2 Crossing-over

Due to real numbers coding, we decided to use weighted average to generate children. However, to enhance the crossing possibilities two weight values are used. One for the front and one for the back of the object. Values in the middle are linearly interpolated. This allows the child to have the front similar to one parent and the back to the other.

Figure 4 (below) presents possible offspring of two parents for the given weight values [Table 1 - below]:

Table 1 . Weights of parent values for offspring in Figure 4.

P = pick-up **T = tractor**	Back = 1P + 0T	Back = 0.5P + 0.5T	Back = 0P + 1T
Front = 1P + 0T	Pick-up		Pick-up front + Tractor back
Front = 0.5P + 0.5T		Average of Pick-up & Tractor	
Front = 0P + 1T	Tractor front + Pick-up back		Tractor

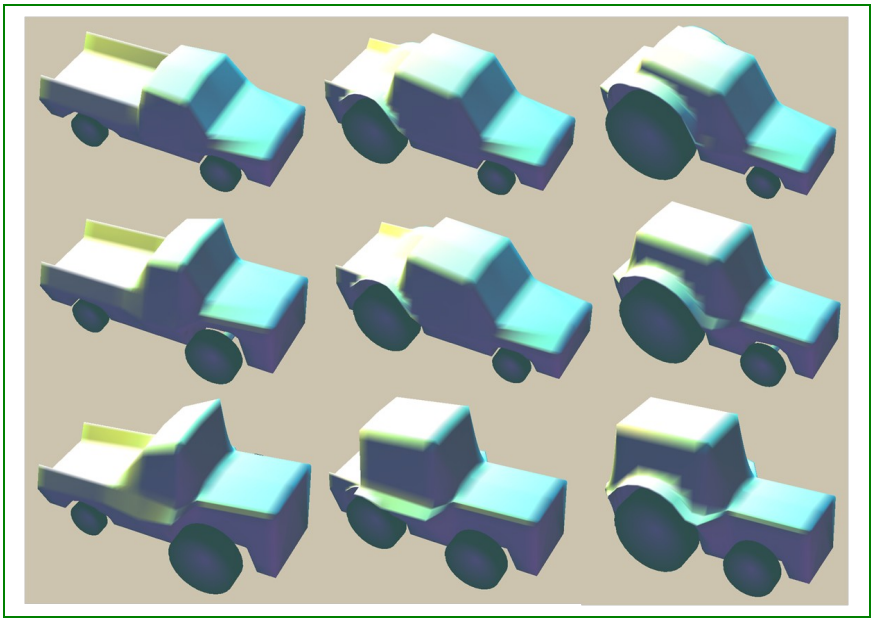


Fig. 4. Example offspring of two parents: Pick-up and Tractor

3.3 Mutation and Repairing Rules

Mutation operator allows to create variations of a given object. Mutations are random changes applied to gene values. Normal distributed random numbers are used in the application. Size of the mutation is controlled by the variance parameter. The example

mutants are shown on Figure 5 - below. During the automated evolution mutations are the source of innovations.

Sometimes the mutations are so unfortunate that the result cannot be called a car. To prevent such a case a set of repairing rules is introduced, for example: “The cabin roof should not be depressed”. Their role is to keep the objects in the domain boundaries. This type of correctness control is simple to manage. Rules can be easily added, deleted or modified. It is also possible to parameterize them in order to choose between a narrow domain of regular models or a great variety of inspirational, concept cars.

The Repairing Rules can be consider as an upper level of mutation – aimed interference to genotype. The reparations lead to characteristic phenomenons. For example it is very hard to achieve a flat roof with regular mutations. However sometimes they produce a depressed roof which then is fixed to be flat (rule: “The cabin roof should not be depressed”). The system designer should be aware of this kind of occurrences.

Repairing also have some disadvantages. First of all, after frequent reparations objects tends to converge towards the domain boundaries. Secondly, some rules are based on many genes, thus they are being broken (and repaired) very often. Some rules can also interfere with the others. They should be designed with a lot of caution. Also some more research is needed on repairing methods and their influence on the algorithm.

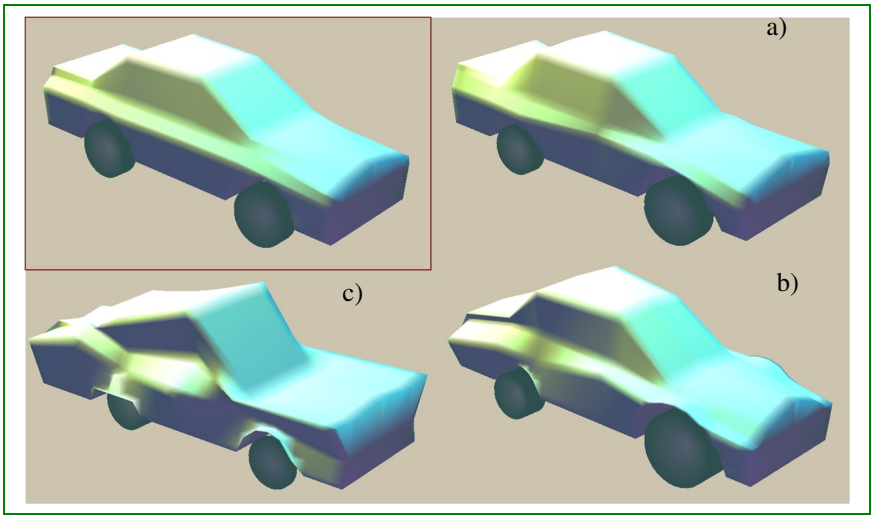


Fig. 5. Reference car model and its selected copies with applied mutations: a) light; b) medium; c) heavy.

3.4 Automatic evolution

Automatic evolution gathers all the techniques mentioned above in one genetic algorithm. The user input is a sketch of a desired car profile and the output are models generated by the algorithm. The examples are presented on Figure 6 – below.

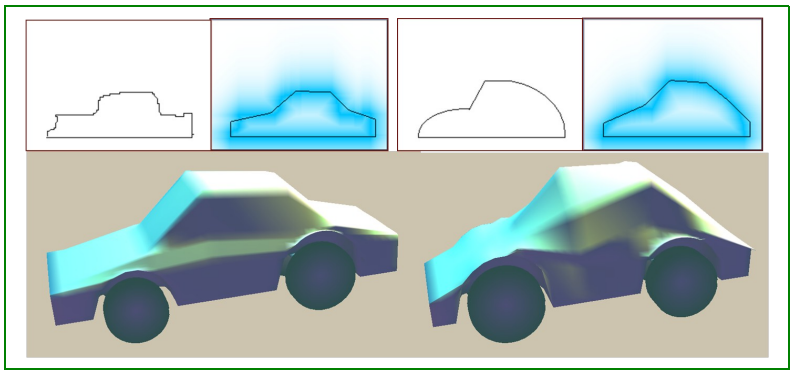


Fig. 6. Results of automatic evolution: provided profile, profile of the result object and visualization of the result.

In order to speed up the evolution process the sketches provided by the user are smoothed. The algorithm analyzes the similarity of processed sketches to the profiles of the generated models.

It is possible to run the algorithm without providing any sketch. Then the evolution turns into random walk. This mode can be used for generating inspirational models. It also helps when defining the domain boundaries and Repairing Rules. Figure 7 - below shows the example results

Figure 7. also shows characteristic mutations. The user may have an impression that the algorithm favors convex cowls. This is because in most of regular car models the cowl is flat or almost flat. Thus the user expects that a convex cowl is something unusual but most mutations will disturb the flat shape.

Here we can identify the possible issue: in the real world the car models are not distributed uniformly in the domain space. The distribution reflects a typical customer's likes, ergonomics and many other factors. The real objects are usually grouped in sub-classes (e.g. small city cars, sport cars, trucks). If these groups are not centered in the search area, the user may have an impression that some of an unusual model features are preferred. The problem is similar to the domain boundaries issue (mentioned in section 3.3), however here the goal is to satisfy the user's subjective judgment.

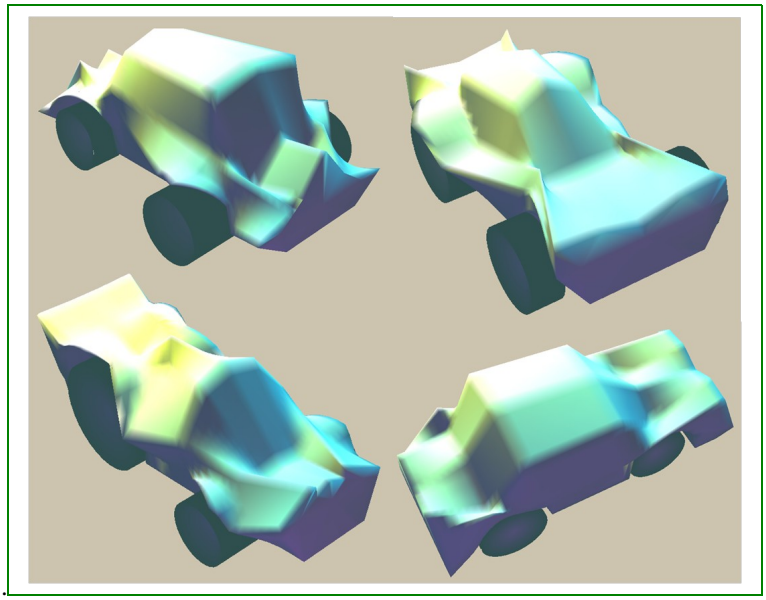


Fig. 7. Random walk results.

A good solution could be to force the non-uniform distribution of mutants. The fitness function could analyze the similarity to a reference model group and punish those objects that differs too much. It would be good idea to parameterize the punishment size so that the user can choose whether unusual models are more or less expected.

4 Conclusions and Future Work

Experience with the presented prototype system indicates that it allows to create and modify the car models more quickly than by standard 3D modeling. The genetic techniques approach is intuitive and allows the users to take part in the designing process even without having much experience in modeling.

The development process of the system indicates, that besides genotype design, the key part is defining the domain boundaries. These usually need to be explored during the development process, so use of easy manageable domain defining rules is a good idea. This method, however, requires much caution. Repairing rules and their influence on algorithm are a possible topic of further research.

One of the other key-parts is gene interpreter - responsible for constructing results. It plays the role of the manufacturer that prepares the product according to given

specification (genotype). Designing genotype and interpreter are two connected parallel processes.

During the whole algorithm development process, special attention should be paid to identify and control subjective esthetical aspects. The cowl convexity has been mentioned in section 3.4, but these aspects may be more complex, for example "aggressive look". These are being judged by the user and as much as possible should be properly recognized and handled by the system. The next step would be to allow the user to directly manipulate these features (phenotype).

As for now it is difficult to model smooth shaped objects in the given solution. The further development of the system involves improving the genotype interpreter to allow this, for example with the use Bézier curve interpolation. Also more details can be contained in the genotype (e.g. mirrors, lights).

Finally, the plan is to design a framework for construction similar evolutionary based solutions for other classes of objects, for example: clothes, furniture, pieces of architecture, jewelers, etc.

References

1. Campbell Neil, *Computer Graphics lectures* (2006), Department of Computer Science University of Bristol.
2. Cristoph Traxler and Michael Gervautz: *Using Genetic Algorithms to Improve the Visual Quality of Fractal Plants Generated with CSG-PL-Systems,* (1998) Institute of Computer Graphics, Vienna University of Technology, Austria.
3. Komosiński Maciej: *The Word of Framsticks: Simulation, Evolution, Interaction (2000)* Institute of Computing Science, Poznan University of Technology.
4. Kratky Andreas: *XFrog – reference manual* (2001).

Proceedings of the International Multiconference on Computer Science and Information Technology pp. 409–419
ISSN 1896-7094
© 2007 PIPS

An investigation of the mutation operator using different representations in Grammatical Evolution

Jonatan Hugosson, Erik Hemberg, Anthony Brabazon & Michael O'Neill

Natural Computing Research & Applications Group
University College Dublin, Ireland
jhugosson@gmail.com, erik.hemberg@ucd.ie, anthony.brabazon@ucd.ie, m.oneill@ucd.ie

Abstract. Grammatical evolution (GE) is a form of grammar-based genetic programming. A particular feature of GE is that it adopts a distinction between the genotype and phenotype similar to that which exists in nature by using a grammar to map between the genotype and phenotype. This study seeks to extend our understanding of GE by examining the impact of different genotypic representations in order to determine whether certain representations, and associated diversity-generation operators, improve GE's efficiency and effectiveness. Four mutation operators using two different representations, binary and gray code representation respectively, are investigated. The differing combinations of representation and mutation operator are tested on three benchmark problems. The results provides support for the continued use of the standard genotypic integer representation as the alternative representations do not exhibit higher locality nor better GE performance. The results raise the question as to whether higher locality in GE actually improves GE performance.

1 Introduction

Grammatical evolution (GE) [15, 14, 20] is a form of grammar-based genetic programming. A special feature of GE is that, unlike genetic programming, it has a clear distinction between the genotype and phenotype. The mapping of the genotype and phenotype is governed by a grammar and this grammar can contain domain knowledge to bias the form a phenotypic solution can take. By separating the search and solution spaces, GE allows the implementation of generic search algorithms without a requirement to tailor the diversity-generating operators to the nature of the phenotype. A substantial literature has emerged on GE and its applications [15, 1, 19, 17]. Some of the more recent developments of GE are focused on the various components of the GE approach including the use of alternative search engines [10, 13], the use of alternative grammar constructs [11, 3, 9, 16], and the examination of different mapping processes in GE [12]. One aspect of GE which has seen less research is the examination of the impact of the choice of genotypic representation, and associated diversity-generation operators, on GE's efficiency and effectiveness. A recent paper by Oetzel and

Rothlauf [19] examined the locality properties of a binary representation in GE and found that a genotypic bit-mutation operator produced non-local changes in the phenotype. The authors of this study proposed that further research be undertaken in order to find other representations and associated mutation operators which would produce higher locality, suggesting that this would increase the performance and effectiveness of GE. This study addresses this research issue by investigating the impact of four mutation operators using two different representations, binary and Gray code representation respectively, on the performance of GE. The combinations are tested using three standard benchmark problems, symbolic regression, the Santa Fe ant trail and the even-5-parity problem.

The remainder of the paper is structured as follows. Section 2 describes GE and provides background on earlier work on representations. Section 3 details the experimental approach adopted and results, and finally section 4 details conclusions and future work.

2 Background

This section provides an introduction to GE and to some prior work on the importance of representation in evolutionary algorithms. GE is an evolutionary algorithm which has some similarities to genetic programming (GP) [7]. Rather than representing the programs as parse trees, as in GP, a linear genome representation is used. A genotype-phenotype mapping is employed such that each individual's variable length binary string, contains in its codons (groups of 8 bits) the information to select production rules from a Backus Naur Form (BNF) grammar, see Fig. 1. Consequently, the genetic operators such as crossover and mutation are applied to the linear genotype in a typical genetic algorithm (GA) [5] manner, unlike in a tree-based GP approach where they are applied directly to the phenotypic parse trees. The grammar allows the generation of programs in an arbitrary language that are guaranteed to be syntactically correct. The user can tailor the grammar to produce solutions that are purely syntactically constrained, or they may incorporate domain knowledge by biasing the grammar. The mapping process creates a clear distinction between the search and solution space.

An important element in a successful application of evolutionary methodologies is a careful co-selection of a representation and associated diversity-generating operators which are well-suited for a specific problem landscape. These choices can radically change the performance of an algorithm. Easy problems in one representation can be hard in another. A substantial literature exists on the importance of representation choice and readers are referred to [18] for a detailed discussion of this issue. In that study three different types of representations are analysed for a GA under influence of the crossover operator. The study focuses on redundancy, scaling of building blocks and the modification of distance between individuals when mapping the genotype to the corresponding phenotype. It suggests that redundancy in a representation typically has a neutral or negative effect on algorithmic performance. In contrast, GE always maps

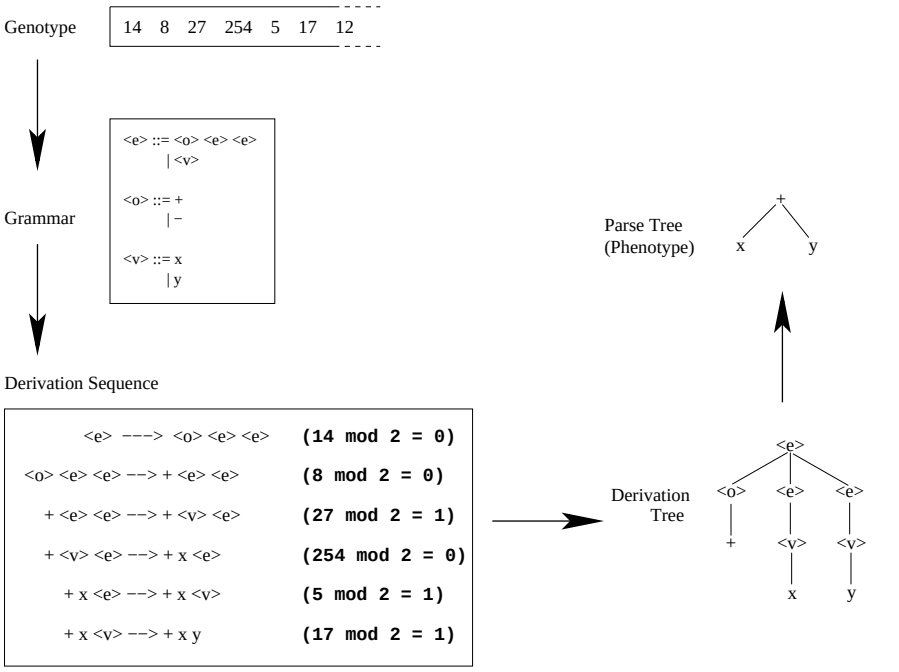


Fig. 1. An example GE genotype-phenotype mapping, where the genotype is used to select production rules from a grammar to produce a derivation sequence. The derivation sequence represents the development of a program from the embryonic non-terminal start symbol (<e>). The derivation sequence can be represented as a derivation tree, which can then be simplified to correspond to the parse tree adopted in standard tree-based GP.

via the BNF grammar to the solution space, producing a much more complex genotype-phenotype mapping than typically exists in a GA. The locality of a genotype-phenotype mapping describes how well genotypic neighbors correspond to phenotypic neighbors. Rothlauf and Oetzel [19] investigate the locality of this mapping in GE and suggest that it has low locality as neighboring genotypes do not necessarily correspond to neighboring phenotypes. Based on this finding, they suggest that further work is needed in order to develop representations with higher locality in order to maximise the efficiency of GE.

In prior investigations on GAs, Hollstien [6] claimed that gray code works slightly better than the binary representation. Gray code has some advantages compared to binary code [18] as it is not affected by scaling and it has perfect locality concerning small changes. Therefore the difficulty of a problem remains unchanged when using mutation-based search. However gray coding may not produce the same effect in GE since GE uses a complex many to one mapping (arising from the BNF grammar) causing neutral mutations whose effects also must be considered. The neutral mutations are believed to cause higher diversity

and thus higher fitness [2, 22, 4, 19]. If neutral mutations are important mutation operators with high neutrality should be advantageous. Yu and Miller, [21], state a hypothesis about the importance of the ratio between adaptive/neutral mutations. The mapping in GE gives it a non uniform representation and thus may take longer to converge. GE is also subject to a *ripple effect* when a standard GA is applied. As the function of a gene depends on the genes that proceeds it, a small genotypic change can lead to a big phenotypic change. Evidence suggests that this effect can promote a useful exchange of derivation sub-sequences during crossover [9].

3 Experimental setup & Results

This section outlines the experimental setup used in this study, details the results of these experiments, and provides a discussion of the key findings.

3.1 Hypothesis and Setup

Given the best fitness value after 50 generations for each mutation operator: μ_0, best fitness using normal integer mutation. μ_1, plus minus mutation. μ_2, binary mutation and μ_3, gray code mutation. The following hypothesis is stated:

H_0**:** None of the representations and mutations proposed gains significant performance to GE in any of the experiments, i.e. $\mu_0 = \mu_1, \mu_2, \mu_3$.
H_1**:** At least one of the mutations gains significant performance for at least one experiment, i.e. $\mu_0 > \mu_1$ or $\mu_0 > \mu_2$ or $\mu_0 > \mu_3$.
α**:** The significance level of the test is 0.05.[1]

Since the mutation operator is believed to be very problem dependent all three experiments are run with the four different mutation operators over 30 runs. Ramped Half-and-half initialization with a maximum tree depth of 8 is used. The mutation probability was 0.01, crossover probability 0.9 and the number of wraps was 30. To raise the diversity within the population, unique children regarding phenotypic difference from its parents are more likely to be produced. Two parents are picked using tournament selection with tournament size ten (population size 500). Steady state replacement is used. The solution quality is measured by cumulative frequency (rate of success) and best fitness value for the last generation (50th generation). To analyze GE's performance a t-test is performed on the best fitness after 50 generations. The data is assumed to come from normal distribution with unknown, but equal, variance. See Fig. 2 for the grammars.

[1] α is the probability of making a type 1 error, i.e. The probability of rejecting H_0 given that H_0 is in fact true.

```
EVEN-5-PARITY
<prog> ::= <expr>
<expr> ::= <expr> <op> <expr> | ( <expr> <op> <expr> ) | <var>
<op> ::= "|" | & | ^
<var> ::= d0 | d1 | d2 | d3 | d4
SYMBOLIC REGRESSION
<prog> ::= <expr>
<expr> ::= <expr> <op> <expr> | ( <expr> <op> <expr> ) | <pre-op> ( <expr> ) | <var>
<op> ::= + | * | -
<pre-op> ::= sin | cos | exp | log | inv
<var> ::= X | 1.0
SANTA FE ANT TRAIL
<prog> ::=<code>
<code> ::=<line>|<code><line>
<line> ::=<condition>|<op>
<condition> ::= if (food_ahead()==1) { <line>} else {<line> }
<op> ::=  left(): | right(): | move():
```

Fig. 2. BNF grammars used in the experiments.

3.2 Description of different mutations

Mutations that change the genotype but not the phenotype are called neutral and can arise due to functional redundancy, implicit neutrality, or mutation on inactive genes [22, 15]. Adaptive mutations are mutations that do change the phenotype. Neutral mutations cause diversity within equally fit individuals in the search space, while adaptive mutations explore the solution space.

In canonical GE, a 32 bit integer representation is used, where each codon is defined by an integer. In a *integer mutation*, the current integer value in a codon is replaced by a new randomly-generated integer. All points in the search space can be reached with the mutation operator. A *plus minus mutation* adds or subtract one from the current integer value of the codon in question. This type of mutation extinguishes the neutral mutations, and narrows the search space given some local optimum. The *binary mutation* used here flips one bit in the codon. Using the Gray code representation of the bitstring a *gray code mutation* will change the codons in a different pattern compared to the binary representation, notice that the gray code changes one bit every time adding or subtracting one. The choice of grammar affects the result of the different operators, for example a smooth grammar might be advantageous for the gray code and plus minus mutation while a grammar with an equal number of production rules for each non-terminal would make the neutral mutations vanish.

3.3 Mutation With Crossover

The amount of statistical analysis for these experiments is quite large, the tables for statistical analysis are omitted, instead significant difference is mentioned in the caption of the figures. Results for mutation with crossover can be seen in Fig. 3.

The only experiments that has a significant difference in the GE performance (best fitness) is the even-five-parity experiment, see Fig. 4. In this experiment

Symbolic regression $f(x) = x^4 + x^3 + x^2 + x$			
Mutation operator	Mean best fit.	Cum. frequency	Standard deviation
Integer mutation	1.1	9	1.4
Binary mutation	1.3	2	0.8
Gray code mutation	1.5	4	1.2
Plus Minus mutation	1.6	1	1.4
No mutation	4.1	4	3.4
Even-five-parity			
Mutation operator	Mean best fit.	Cum. frequency	Standard deviation
Integer mutation	3.2	0	1.9
Binary mutation	4.0	0	2.1
Gray code mutation	2.8	0	2.8
Plus Minus mutation	3.7	0	1.7
No mutation	6.5	0	2.2
Santa Fe ant trail			
Mutation operator	Mean best fit.	Cum. frequency	Standard deviation
Integer mutation	15.6	14	17.4
Binary mutation	13.2	15	15.7
Gray code mutation	12.5	15	15.1
Plus Minus mutation	8.9	19	13.8
No mutation	33	2	12

Fig. 3. Results for experiments using mutation and crossover for 30 runs. The mean best fitness (minimizing), cummulative frequency of success and the standard deviation.

integer mutation outperforms binary mutation while no other significant differences exist. Thus the hypothesis H_0, that none of the proposed operators produces any significant performance advantage, cannot be rejected.

Regarding the mean fitness value, see Fig. 4 and 5, we can draw the conclusion that in all the experiments except even-five-parity, the mutation operator causing the most neutral mutations also has the lowest mean fitness value. Neutral mutations cause phenotypic diversity to decrease and thus produce reduced exploration. The adaptive mutation operators in all experiments have the highest cumulative frequency. Adaptive mutations therefore seems to be advantageous for finding the optimal solution, however maybe neutral mutation with higher mutation rate would perform as well or better using both good exploration and neutral features.

For each experiment no wrapping and no mutations is also tested. Since wrapping is turned off the length of the genotypes are increased to an average of 400 codons. As can be seen in Fig. 3 the canonical GE settings of using mutation and wrapping do produce higher performance than when these mechanisms are 'turned-off'.

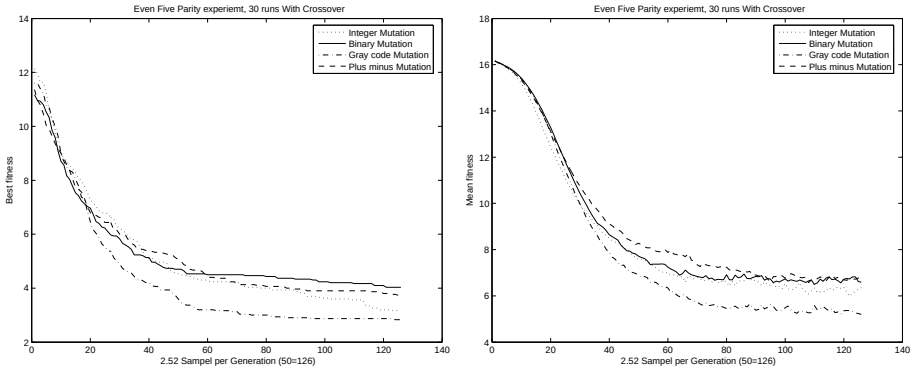


Fig. 4. *Left*: Even-five-parity GE performance, best fitness after 50 generations, for the four mutation operators. There are significant difference between the binary mutation and the integer mutation. *Right*: Even-five-parity mean of the best fitness after 50 generations for the four mutation operators. There are significant difference between the Plus minus mutation and the three others.

3.4 Mutation Without Crossover

In these experiments the crossover probability is set to zero, other settings are as in the previous experiments. The results are shown in Fig. 6.

No conclusions about GE performance and the mutation operators should be drawn since the only factor that seems to matter is the actual rate of exploration. This can be altered by increasing the mutation probability. In the Santa Fe ant trail experiment, it can clearly be seen that the binary and integer mutation are outperformed by plus minus and gray code mutation. Without crossover, neutral mutations cause worse exploration and thus lower variance, diversity, and performance. Since the algorithm has a phenotypic diversity implemented this does not properly justify the way neutral mutations have been shown to work in previous work, see [21, 22]. The best fitness for the plus minus operator is not significantly better than that produced by any other operator. There is no bias induced by the grammar since the Santa Fe ant trail experiment always chooses among three or fewer production rules. The plus minus operator can reach all the points of the search space, but unlike the integer mutation no mutations are neutral and thus the plus minus mutation has a higher rate of exploration. The results do not indicate that any firm conclusions can be drawn regarding locality and mutation. None of the proposed operators appear to produce higher locality, the only factor that seems to be of importance is the exploration rate caused by fewer neutral mutations. It is likely that these result emerge because the GE grammar mapping is so complex that no matter what representation is used, the effect caused by a mutation causes a substantial change in the phenotypic structure.

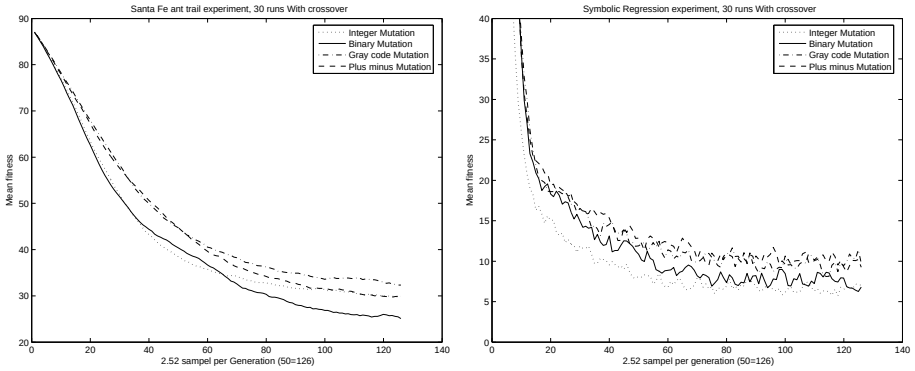


Fig. 5. *Left*: Santa Fe ant trail mean of the best fitness after 50 generations for the four mutation operators. There are significant difference between the Gray code and binary mutation. *Right*: Symbolic regression mean of the best fitness after 50 generations for the four mutation operators. There are significant difference between the binary and Gray code and plus minus mutation respectively.

4 Conclusions & Future Work

The object of this study was to examine the impact of different genotypic representations in GE in order to determine whether certain representations, and associated diversity-generation operators, improve GE's efficiency and effectiveness. Four mutation operators using two different representations, binary and Gray code representation respectively, are investigated. The different combinations of representation and mutation operator were tested on three benchmark problems. The following conclusions are made regarding the mutation operators.

- None of the proposed mutations gain any significant GE performance. Thus it is not possible to reject the hypothesis that at least one of the proposed mutation operators is significantly better than the integer mutation for at least one experiment.
- Using a mutation rate at 0.99, adaptive mutation operators seem to be preferable since they result in higher cumulative frequency. This result could arise because the mutation probability, for optimal performance, should be higher. Then, having the correct rate of adaptive mutations, neutral mutations could be of importance. However it is clear that adaptive mutations are of greater significance than neutral mutations.

Even though no clear improvements can be seen through new mutation operators there is no doubt that mutation increases GE's performance (see Fig. 3). Since GE has a complex mapping from genotype to fitness value, the results do not suggest that changing the representation of the genotype has a significant impact on GE performance. This does not imply that the representation is irrelevant, rather it suggests that the examination of the utility of a specific representation

Symbolic regression $f(x) = x^4 + x^3 + x^2 + x$			
Mutation operator	Mean best fit.	Cum. frequency	Standard deviation
Integer mutation	8.4	0	3.7
Binary mutation	7.9	0	3.8
Gray code mutation	7.4	0	3.8
Plus Minus mutation	7.2	0	3.5
Even-five-parity			
Mutation operator	Mean best fit.	Cum. frequency	Standard deviation
Integer mutation	5.0	0	2.5
Binary mutation	5.5	0	2.5
Gray code mutation	5.0	0	2.3
Plus Minus mutation	5.8	0	2.4
Santa Fe ant trail			
Mutation operator	Mean best fit.	Cum. frequency	Standard deviation
Integer mutation	33.2	2	15.7
Binary mutation	38.3	1	9.1
Gray code mutation	29.8	1	13.7
Plus Minus mutation	26.2	6	16.4

Fig. 6. Results for experiments using only mutation for 30 runs. The mean best fitness (minimizing), cummulative frequency of success and the standard deviation.

cannot be isolated from an examination of the mapping process embedded in the grammar. One way of investigating locality further would be to create a grammar that changes gradually between the properties of each production rules, this mutation could explore the search space more smoothly. Investigation of different mapping methods and how to create operators that perform a more local search, e.g. context sensitive mutations might improve the understanding of GE.

References

1. Brabazon Anthony and O'Neill Michael (2006), *Biologically Inspired Algorithms for financial Modelling*, Springer.
2. Burke E., Gustafson S. and Kendall G. (2004), *Evolutionary Optimization in uncertain Environments—A Survey*, IEEE Transactions on Evolutionary Computation 9(3) 2005, pp. 303–317.
3. Dempsey I., O'Neill M., and Brabazon A. (2005). *Meta-grammar constant creation*, Proc. of GECCO 2005, pp. 1665–1672, ACM Press.
4. Galvan-Lopez E. and Rodriguez-Vasques K. (2002), *The Importance of Neutral Mutations in GP*, PPSN IX, LNCS 4193, pp. 870–879, 2006.
5. Goldberg David E. (1989), *Genetic Algorithms,* Addison Wesley Longman.
6. Hollstein R. B. (1971), *Artificical Genetic Adaption in Computer Control Systems,* PhD thesis, University of Michigan.
7. Koza J. R. (1992). *Genetic Programming: On the Programming of Computers by Means of Natural Selection,* MIT Press.

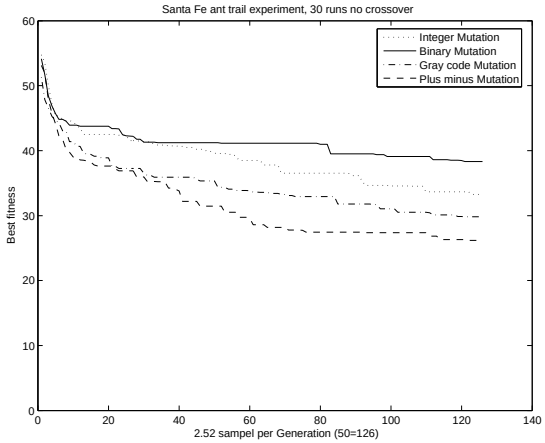


Fig. 7. Santa Fe ant trail GE performance for the four mutation operators without the crossover operator. The Binary mutation is significantly different from the plus minus and the Gray code mutation.

8. Langdon William B. and Poli Riccardo (1998), *Foundations of Genetic Programming,* Springer.
9. Nicolau M. and Dempsey I. (2006), *Introducing Grammar Based Extensions for Grammatical Evolution,* CEC 2006, pp. 648–655.
10. O'Neill M. and Brabazon A. (2004). *Grammatical swarm,* Proc. of GECCO 2004 LNCS 3120, Part 1, pp. 163–174, Springer-Verlag.
11. O'Neill M. and Brabazon A. (2005). *mGGA: The meta-Grammar genetic algorithm,* EuroGP 2005 LNCS 3447, pp. 311–320, Springer.
12. O'Neill M., Brabazon A., Nicolau M., McGarraghy S., Keenan P. (2004). *πGrammatical Evolution,* Proc. of GECCO 2004, LNCS 3103, Part 2, pp. 617–629, Springer-Verlag.
13. O'Neill M. and Brabazon A. (2006). *Grammatical Differential Evolution,* Proc. of ICAI '06, pp. 231–236, CSEA Press.
14. O'Neill M., Ryan C. (2001). *Grammatical Evolution,* IEEE Trans. Evolutionary Computation 2001.
15. O'Neill, Michael and Ryan, Conor (2003), *Grammatical Evolution,* Kluwer.
16. O'Neill M. and Ryan C. (2004). *Grammatical evolution by grammatical evolution. The evolution of grammar and genetic code,* EuroGP 2004 LNCS 3003, pp. 138–149 Springer.
17. O'Reilly, U. M. and Hemberg M. (2007), *Integrating generative growth and evolutionary computation for form exploration,* Genetic Programming and Evolvable Machines, Vol. 8, pp. 163–186, Springer.
18. Rothlauf F. (2002), *Representations for Genetic and Evolutionary Algorithms,* Physica-Verlag, Heidelberg.
19. Rothlauf F. and Oetzel M. (2005), *On the Locality of Grammatical Evolution,* EuroGP 2005 LNCS 3905, pp. 320–330, Springer.
20. Ryan C., Collins J. J., O'Neill M. (1998). *Grammatical Evolution: Evolving Programs for an Arbitrary Language,* EuroGP 1998 pp. 83–95, Springer-Verlag.

21. Yu T. and Miller J. (2002), *Finding Needles in Haystack is Not Hard with Neutrality,* EuroGP 2002, pp. 13–25.
22. Yu T. and Miller J. (2002), *Neutrality and the Evolvability of Boolean Function Landscape,* EuroGP 2001, Vol. 2038, pp. 204–217.

Proceedings of the International Multiconference on Computer Science and Information Technology pp. 421–435
ISSN 1896-7094
© 2007 PIPS

Knowledge based segmentation of fundus eye images

Leslaw Milosz Pawlaczyk

Department of Computer Science,
Silesian University of Technology,
ul. Akademicka 16, 44-101 Gliwice, Poland
leslaw.pawlaczyk@polsl.pl

Abstract. *In this article we describe a new method for segmentation of two anatomical structures visible in fundus eye images: vessels and eye cup. The method is called Universal Segmentation Scheme (USS) and is based on the knowledge stored in rules of an expert system (ES) and PNC2 rule induction system. The USS is a general method which is later adapted for each of the anatomical structures specifically. The results of segmentation are compared to the ground truth images which show the relative segmentation error.*

1 Introduction

Segmentation is a process of dividing a digital image into a set of separate regions which are supposed to represent objects located in a scene. In the article [5] authors presented a detailed analysis of segmentation approaches known at the time. The conclusion given by authors was that the segmentation task cannot be solved unequivocally. This means that each observer will understand differently what sort of output should be presented by segmentation algorithm. That makes this problem difficult to solve for humans and especially for computers. In this article we want to introduce a new segmentation approach called later as the Universal Segmentation Scheme (*USS*). An idea of this solution is to incorporate domain knowledge into USS which should improve the results achieved by older approaches [14, 17]. The domain knowledge is represented by a set of rules which operate on regions extracted by initial image analysis stages. Also a rule induction algorithm verifies the final membership of pixels into output regions.

The author's goal was to create an easily adaptable framework which will have the ability of performing well for different applications (the algorithm should work with other types of images). This ability is achieved by using rule based model for knowledge representation and creating verification rules by training them on a hand labelled set delivered by field domain experts. The system was tested on a set of fundus eye images (*fei*) and specifically customised for the purpose of vessels and eye cup segmentation. Originality of this work relies in the approach to segmentation of *fei* in general. Earlier approaches were

specifically designed for the particular anatomical organ. The proposed framework allowed author to embrace in one general scheme the segmentation tasks for two different targets.

We decided to use *fei* as the source of data because there's a visible need for good quality segmentation algorithms which could simplify work of ophthalmologists. Eye diseases such as glaucoma or diabetic retinopathy are one of the major causes of blindness. Early detection and prevention by analysing *fei* can lead to stopping the progress of a disease and help to save patients eye sight [11].

In section 2 a detailed description of the USS is presented. Section 3 gives insides of the vessels and eye cup segmentation. Each algorithm is illustrated by action flow chart as well as the segmentation results. The final conclusions and possible applications of USS are depicted in section 4.

2 Description of the Universal Segmentation Scheme

The USS can be presented using the steps shown in Fig. 1. Here's a short description of assistant stages and the USS method itself:

- *Creation of domain knowledge base*—definition of extraction and merging rules for the purposes of segmentation.
- *Initial processing*—ensuring a similar appearance by applying the techniques of filtration and image enhancement for the purposes of process automation.
- *Definition of features space*—description which features form the multidimensional space where clustering is performed.
- *Image segmentation using clustering in feature space*—image is segmented by clustering algorithm, in our approach we use FCM [1] as the initial stage for dividing image into regions.
- *Extraction of regions with application of domain knowledge*—the extraction rules are applied to isolate regions.
- *Verification of regions found using knowledge and control strategy*—regions merging rules are applied after extraction stage to refill incomplete pieces of the segmented object.
- *Application of rule induction system in model adaptation in case of requirements change*—this stage allows to implement the last improvements into segmentation using the PNC2 rule induction algorithm (trained on a set of hand labelled images).

2.1 Domain knowledge creation, image preprocessing, definition of features space

Before execution of the main segmentation USS stage we need to define how the domain knowledge would be represented. The knowledge is stored as a set of extraction, merging and erasing rules for regions. An important part of the domain knowledge is also a set of images manually segmented by a specialist—a so called ground truth images [3, 6]. The mentioned images serve as input

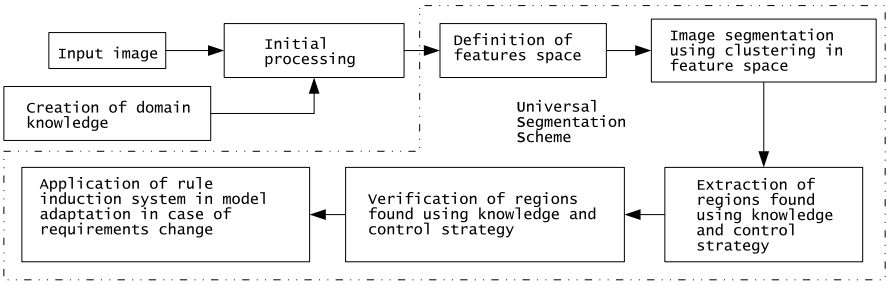


Fig. 1. Universal Segmentation Scheme with additional stages

data for PNC2 algorithm which generates rules used in the last stage as a final classification for pixels to one the groups: object or background.

Image preprocessing techniques which we applied for image enhancement were: brightness equalisation [20], contrast improvement [22], median filtering [8] and histograms specification [20]. The goal of the preprocessing stage is noise reduction and assurance of the similar quality of images. This allowed later to match identical parameters of USS for whole testing set and then automate segmentation process.

Features space is a place where FCM clustering and PNC2 system are working in. Each feature has integer values in range of [0..255] which allows to look at the feature as an image in grey scale. Matching this range allows first of all to asses the feature usefulness. Secondly is helps avoiding the problem of constructing the optimal features vector in terms of assigning correct weights to each of vector's dimensions.

2.2 Image segmentation using clustering in feature space

On the Fig. 2 we can see a detailed operations flow of the image segmentation using clustering in feature space. In a digital image submitted to segmentation the objects searched are located only in some fragments of the scene. Some researchers [17] use this fact to divide an image into rectangular windows. We assume that the input image $f(x, y)$ represented as a set of vectors $g_m(x, y)$ (in features space) is divided similarly as in [17] into windows of size $M \times M$ pixels. Creation of the features space is a task characteristic for a particular application. A sample features space could be the two first channels of Otha colour model.

After that we are able to perform segmentation in windows. From an image space we pick up a successive processing window. Small number of pixels is randomly chosen from the window interior and its colour values are then translated into feature space where they form vectors submitted to clustering. A parameter $patt_{prc}$ is defined which describes what percentage of pixels from window is picked up for later clustering as vectors. We assume that a minimal value of this parameter $patt_{prc} = 5\%$.

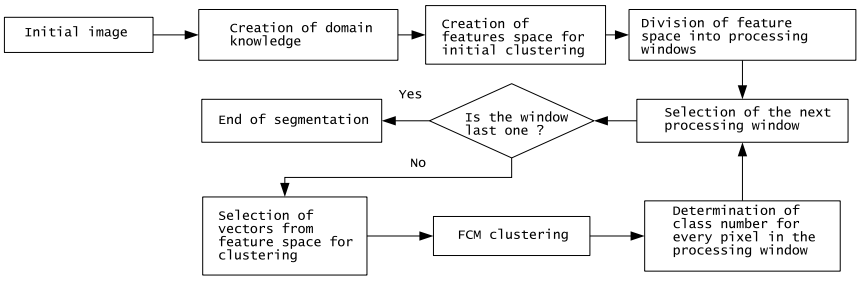


Fig. 2. A scheme of segmentation with application of FCM clustering

After the vectors are chosen clustering is executed. One of the parameters of FCM that we set is number of clusters equal to 4 and $m = 2$. In this case it means that a window after segmentation will be consisted of 4 classes described by different colours. The bigger number of clusters the more granulate the image will be. In the case of application to *fei* we set this number to 4 because the number of distinct anatomical structures is small. Number of clusters is not so important for the next stages as we're using a procedure of connected components counting which are later submitted to a rule base system of regions filtering.

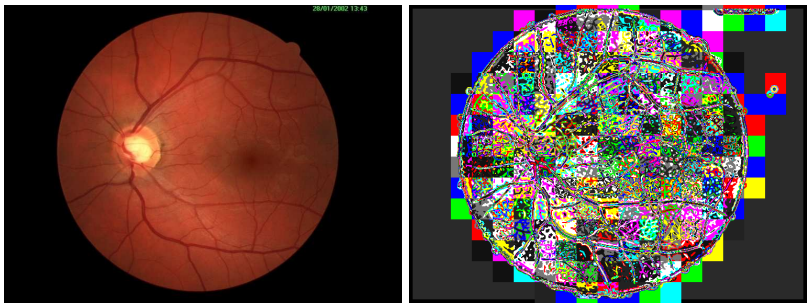
After FCM clustering is done the vectors are divided into separate clusters. Pixels from the window which haven't been represented as vectors are transformed into *additional vectors*. Then they are assigned to a cluster number of the nearest clustered vector using Euclidean metrics.

On the Fig. 3 we can see whole image segmented with windows dimensions 40×40 points. Particular areas with identical classes are depicted by the same colour.

2.3 Extraction of regions with application of domain knowledge

In the previous segmentation step each pixel of the image has been assigned a class. A segmented image consists of compact areas called regions. A region can be eight or four connected [8]. It is a set of pixels $p(x, y)$. If all pixels have at least one pixel lying in their four-neighbourhood the region is four-connected. If any neighbour pixel lies in the analysed pixels eight-neighbourhood then the region is eight-connected. We make an assumption that our regions will be eight-connected.

A region will be defined by a symbol $region_{colour}$, where the attribute *colour* tells about what class the region has. Pixels belonging to the region has an assigned attribute of colour depicted in RGB model as a 24 bit integer value. To extract regions from the Fig. 3b we implement a connected components counting procedure [12]. The aim of the further processing is to find regions in each of the windows used in previous step of USS. The result of this action is a creation of certain collection $Regions[]$.



a) b)

Fig. 3. a) Initial fundus eye image before segmentation, b) the same image partially segmented in windows 40×40 points with the number of clusters equal to 10.

The prepared collection becomes an input data for rules (called extraction rules) which filter regions. As a result of applying these rules (depending on the sort of segmented object) collection *Regions*[] consist of pixels which create the shape of the object. In the case of vessels in *fei* it's necessary to further process them because of errors appearing in the shape of regions such as discontinuities, empty holes or background fragments. The knowledge used in this step is written in the form of rules. Specific rules will be shown later in the article.

2.4 Verification of regions found using knowledge and control strategy

As we mentioned earlier vessels in *fei* require further changes after collection *Regions* has been filtered. In the current step we employ rules of merging and erasing regions [19] with application of specific control strategy which is necessary to define how the collection must be analysed. These rules can be tuned deeply by adjusting its parameters.

Knowledge in the current stage is stored as rules of merging, erasing and the control strategy itself. A proper setting of the mentioned rules is a difficult task, as the efficiency of this step mainly depends on it. Matching too loose criteria of merging will result in addition of unnecessary background artefacts to the collection. On the other hand too restrictive settings will give no improvement in the shape of segmented objects.

As a source of neighbouring regions analysed by merging rules we use regions from the Fig. 3b (image after segmentation using FCM clustering) which are adjacent to the collection. A detailed description of this stage is presented on the example of vessels segmentation in section *Experiments*.

2.5 Application of rule induction system in model adaptation in case of requirements change

A rule induction system PNC2 [9] allows rules to cover examples of classes different than the class returned by the rule itself. It's a cause of classification error on the stage of training. The goal is to minimalise this error. It's achieved by providing additional temporary stage before training, called PNC2 tuning. The rule in PNC2 system is represented as a hypercube.

In this article PNC2 is exploited in the last step of USS, where we check if pixels in previous regions are the correct object pixels. This verification is performed by training PNC2 using ground truth images. These images, after features space generation, give information what features range should characterise pixels belonging to a proper object and which ones belonging to the background. Because PNC2 has the ability of learning we can use it to update segmentation and knowledge model in case when the set of ground truth images is changed.

PNC2 in segmentation of colour images In [3, 7, 6, 10, 13, 15, 16, 18, 21, 25], we can find many different approaches to segmentation which base their functioning on artificial intelligence systems. The knowledge about image can be embraced as rules of an expert system created in cooperation with domain specialist [25]. Another known way is using ground truth type images which are manually divided into objects and background by an ophthalmologist in case of *fei*. Images like this are later exploited as a reference for pictures in the same category.

On the Fig. 4 we can see a general algorithm for segmentation of digital images with application of rule induction system.

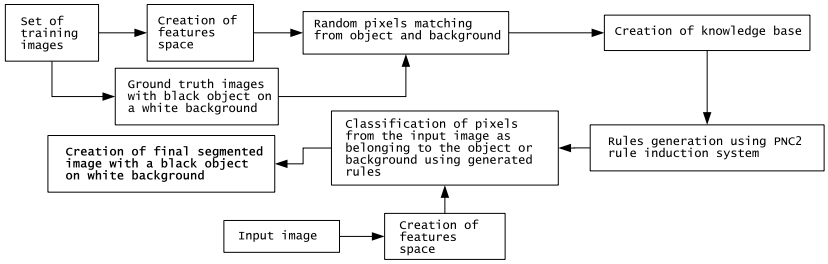


Fig. 4. General scheme of rule induction based image segmentation.

As an input for the algorithm we have a set of digital images belonging to the same class (for example landscape pictures taken on the same location in different light conditions). These images are later fragmented by a domain

specialist, who outline objects with white colour and background as black. Next step is a transformation of pixels into features space.

Because the training set contains images with millions of pixels, it's not reasonable to consider all information given by a features space. The amount of data delivered to the PNC2 shouldn't be too big because of time limitations. This problem can be solved by a randomly chosen small percentage of pixels belonging to the background and object. In our applications we decided to set this values as 0.5% for object and 0.05% for background. Even such a small amount of data gave encouraging results.

When the PNC2 system is properly tuned and trained with data we can test the its efficiency by transforming pixels from a test image into features space and then classify them. The result of classification is assigning each pixel a label; background or object.

The algorithm described above has one significant drawback— we need a good features space representation for pixels to clearly differentiate between object and background features values.

3 Experiments

In this section we'll describe how to apply the USS in order to segment specific anatomical structures on *fei*. As the first one we describe how to extract vessels shape.

3.1 Segmentation of vessels in fundus eye images

Vessels are located in many different places in the image. That's why we can not coarsely define their initial location. This causes that segmentation of vessels is an analysis of the whole image. On the Fig. 5 we can see a general scheme of vessels segmentation. Each of the particular parts is depicted in more details below.

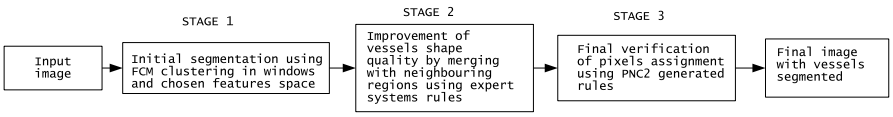


Fig. 5. A general vessels segmentation scheme.

First stage of vessels segmentation In the first stage of vessels segmentation we perform an initial selection of regions in the image which are plausible to be a part of vessels network. On the Fig. 6 we can see a scheme of this method.

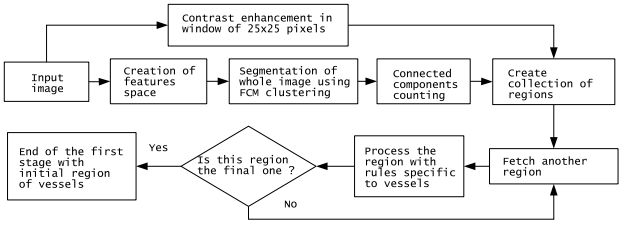


Fig. 6. First stage of vessels segmentation.

The input image is initially filtered to create a features space. In this case our features space composes of one dimensional vector which was prepared with the following techniques:

1. Brightness qualisation using median filter of size 31×31 pixels [20].
2. Contrast enhancement with 25×25 pixels window [8].
3. Denoising with median filter of 5×5 pixels [8].
4. MFR filtering [4].

After its creation the image is segmented using FCM clustering algorithm (part of USS functionality). Subsequently a connected component counting algorithm is performed. These components create collection *Regions*[]. Pixels belonging to the regions have a colour attribute taken from the input image with a local contrast enhancement procedure applied.

Using some specific properties of vessels we propose to use rules analysing subsequent regions from collection *Regions*[] matching appropriate actions depending on the properties of regions. Each rule has its own number defining its priority, the lower number the bigger priority. Here are the rules which support the segmentation of vessels:

RULE (101)
IF AT LEAST 80% PIXELS IN REGION HAVE
VALUE OF RED CHANNEL BETWEEN 4 AND 150
VALUE OF GREEN CHANNEL BETWEEN 4 AND 255
VALUE OF BLUE CHANNEL BETWEEN 4 AND 150
THEN LEAVE REGION,
IN OTHER CASE DELETE REGION FROM THE COLLECTION

RULE (111)
IF AT LEAST 90% PIXELS IN REGION HAVE
LIGHTNESS VALUE LOWER THAN 150
THEN LEAVE REGION,
IN OTHER CASE DELETE REGION FROM THE COLLECTION

All regions which left after processing with these rules are perceived as belonging to the vessels. As the experiments show regions which left are not the only ones which should be included to vessels. The aim of the second stage is to insert neighbouring regions with small differences in red, green and blue channels. On the Fig. 9a we can see vessels found in the first stage of segmentation.

Second stage of vessels segmentation First of all we need to ask ourselves a question: why to correct segmentation results? The answer is a necessity of merging these fragments of the image which weren't recognised in earlier stages. This task applies mostly to vessels because we got a lot of non-continuous regions which shape may be improved using the knowledge stored in the image segmented using the previous windowing technique.

For the purposes of the exact segmentation using regions merging we need to introduce the following definitions:

- Training image—an image created after segmentation using FCM clustering in features space. It consists of colour regions.
- Pattern image—a binarised image with segmented object. In the considered application this will be black vessels in white background.
- Input image—image stored in a bitmap format which we fetch information from about pixels' colours.

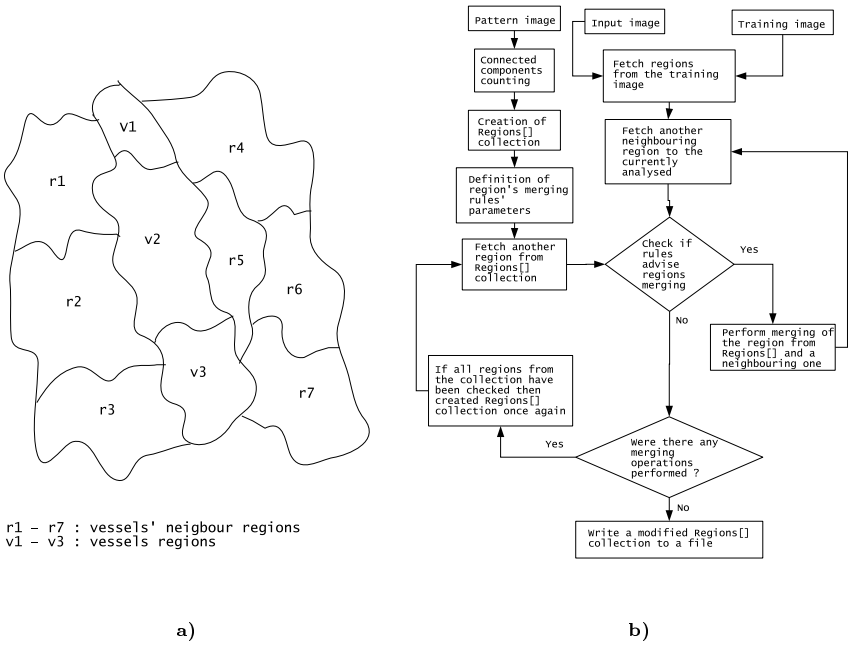


Fig. 7. a) Regions and their neighbouring relations. b) Scheme of the second stage of vessels segmentation.

In [19] authors presented an expert system (*ES*) for improving quality of an image which was segmented by any general segmentation algorithm. The proposed ES advised a particular action which should be performed on the current

object selected from the segmented image. Typical actions were region's division or merging as well as lines merging and deletion. Control strategy was also very important for deciding which parts of the image will be consequently analysed.

Using modified ES rules defined in [19] we propose a specific control strategy which allows to use knowledge stored in the training image to improve quality of any region located in the pattern image. An input image is used as a map of pixels' colours. The pattern image services as a source of one big region depicting the vessels. This big region consist of smaller ones, values of pixel colours are taken from the input image. Upon this basis the subregions create a collection called $Regions[]$.

The aim of the current stage is to fill the gaps between consecutive subregions by merging them with connected components from the training image. The algorithm is reading consequently regions from the collection. An array of pointers to the neighbouring regions is created next for the analysed object.

On the Fig. 7a we can see one of subregions from the $Regions[]$ collection depicted as $v1, v2$ and $v3$ whereas surrounding regions are depicted as $r1, ..., r7$. A procedure of border finding is executed for the mentioned subregion from $Regions[]$. An information of pixel's region assignment is stored within the pixel structure itself. In a case of two regions being merged this information is updated. After detecting regions which surround the analysed one we probe if the merging procedure can be performed.

Here's the set of modified rules from [19] which we apply to analyse if regions may be merged together:

RULE (801)
IF A DIFFERENCE IN ALL RED, GREEN AND BLUE CHANNELS IS SMALL
THEN MERGE REGIONS

RULE (802)
IF SIZE OF A SOURCE REGION IS NOT SMALL
DIFFERENCE IN ALL RED, GREEN AND BLUE CHANNELS IS NOT BIG
THEN MERGE REGIONS

RULE (803)
IF SIZE OF A SOURCE REGION IS SMALL
AVERAGE REGION'S GRADIENT IS BIG
IF A DIFFERENCE IN ALL RED, GREEN AND BLUE CHANNELS IS NOT BIG
THEN MERGE REGIONS

RULE (804)
IF SIZE OF A SOURCE REGION IS SMALL
SOURCE REGION'S ASPECT RATIO IS VERY BIG
DIFFERENCE IN ALL RED, GREEN AND BLUE CHANNELS IS NOT BIG
THEN MERGE REGIONS

RULE (806)
IF SIZE OF SOURCE REGION IS SMALL
NUMBER OF NEIGHBOURING REIONS IS BIG
DIFFERENCE IN ALL RED, GREEN AND BLUE CHANNELS IS NOT BIG
THEN MERGE REGIONS

RULE (807)
IF SIZE OF A SOURCE REION IS VERY SMALL
SIZE OF NEIGHBOURING REION IS VERY SMALL
DIFFERENCE IN ALL RED, GREEN AND BLUE CHANNELS IS NOT BIG
THEN MERGE REGIONS

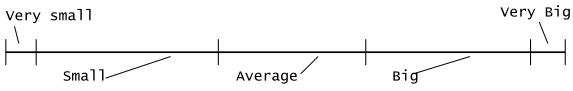


Fig. 8. Graphical representation of definitions: very small, small, average, big, very big.

The definitions introduced above: *VERY SMALL, SMALL, AVERAGE, BIG, VERY BIG* [19] depict in general a *quality* of a parameter. This quality is a ratio between absolute value of the parameter and it's maximum range. The quality is defined by five different symbols. If a parameter is located in a range of symbol meaning then a specific logical function is considered as *TRUE*. A relative location of the symbols is presented on the Fig. 8. The results of vessels segmentation stage 2 can be seen on the Fig. 9b.

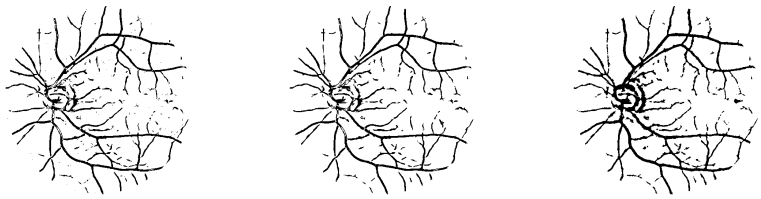


Fig. 9. a) Segmented vessels after stage 1 with the input image from Fig. 3a, b)Segmented vessels after stage 2 with the input image from Fig. 3a, c) Segmented vessels after stage 3 with the input image from Fig. 3a

Third stage of vessels segmentation In the results of the third stage (Fig. 9c) of vessels segmentation—compliant to the scheme in Fig. 4—we use object's region found in the second stage. We mentioned earlier that we use the domain knowledge in a form of PNC2 system rules. In this case we're using six ground truth images. Pixels belonging to an object or background create training data for PNC2 system.

Initially vessels region from stage 2 is enlarged over its border with two surfaces of pixels. The goal of this operation is to increase number of pixels which will be taken as a test set for PNC2. The rule induction system decides which pixels are a part of the object and which are a part of the background.

The usage of knowledge stored in ground truth images is dictated by the fact that pixels may be wrongly assigned in earlier stages.

3.2 Eye cup segmentation

An eye cup is an anatomical structure where the density of optical nerve is highest in the whole *fei*. Any pathological changes of this structure are strictly connected to glaucoma's progress [2]. During segmentation of the eye cup we're using the algorithm for finding the optic disk's region of interest (ROI) [22]. It's a first stage—called initial segmentation, presented on the Fig. 10a.

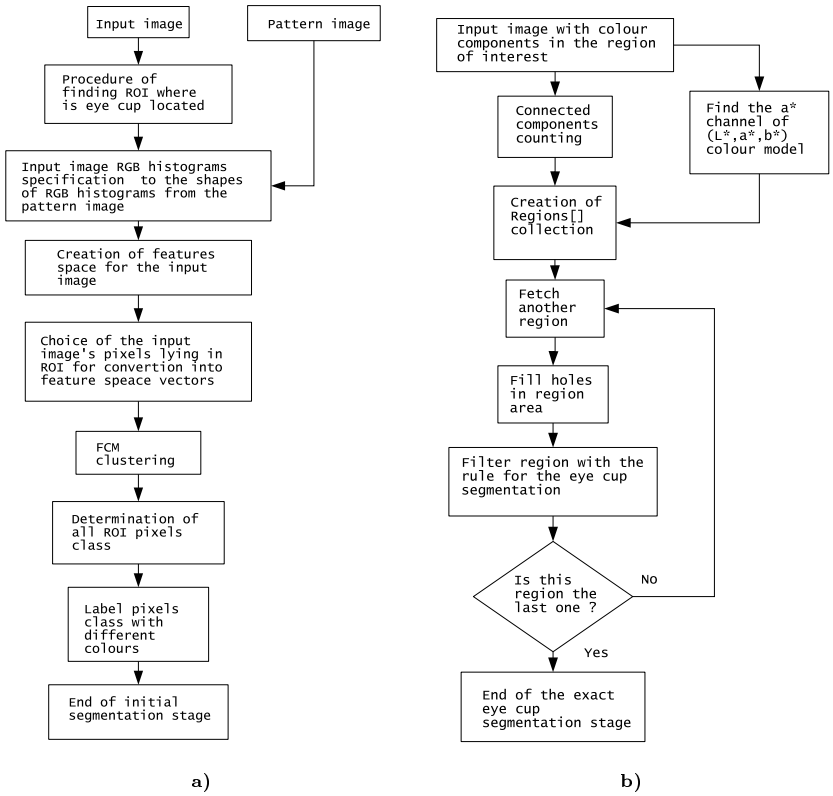


Fig. 10. a) Scheme of eye cup's first segmentation stage b) Scheme of eye cup's second stage segmentation

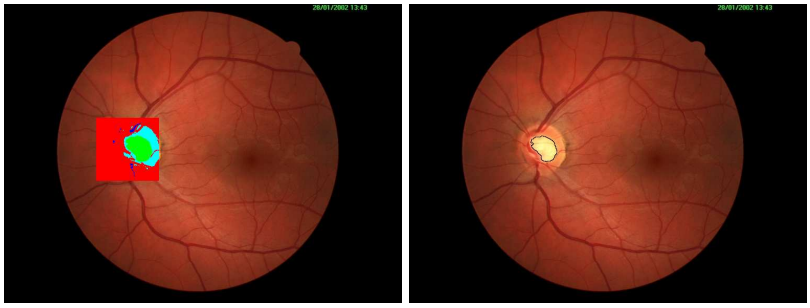
After the ROI is found we use red, green and blue colour channels histogram specification procedure to ensure they have shapes similar to the shapes of pattern image RGB histograms. First of all this operation is performed in order to

secure automatic processing with a set of images which can be later segmented in batch mode. Secondary histograms specification cause the eye cup emerge much better on the image itself which improves segmentation results. The feature space used for FCM clustering is consisting of three Ohta colour model channels. The final stage is existence of colour subregions. These regions are found by applying connected components counting. Similarly to the vessels case they create *Regions*[] collection.

In the stage two (Fig. 10b) an analysis of data from the stage one is performed. A characteristic feature of the eye cup is its small brightness in the channel a* of (L*,a*,b*) colour model comparing to the rest of image [23, 24]. Regions which appeared in the process of connected components counting are later processed by holes filling action. The collection *Regions*[] is analysed by the following rule which marks the proper eye cyp region.

RULE (201)
*IF A REGION HAS THE SMALLEST AVERAGE BRIGHTNESS IN CHANNEL a**
FROM WHOLE REGIONS[] COLLECTION
AND ITS SIZE IS LARGER THAN 100 PIXELS
AND ITS EDGES DON'T MEET WITH ROI EDGES
THEN MARK THE REGION AS THE RESULT EYE CUP

It's obvious that the size of 100 pixels for the minimum eye cup region size is chosen upon the picture size. This size need to be adjusted for the input image size experimentally. On the Fig. 11 we can see the eye cup segmentation results of the two previously mentioned stages.



a) b)

Fig. 11. a) Results of the eye cup segmentation a) stage 1 b) stage 2.

4 Results discussion

The described algorithms for segmentation of vessels and eye cup on *fei* are a detailed version of Universal Segmentation Scheme described in the Fig. 1. The

test data set consists of 10 images. The same set of 10 images was given to an ophthalmologist who manually segmented them. This allowed us to have ground truth images for the evaluation of the results. We used a method of relative segmentation error described in [26]. The average errors were:

- vessels—10.8%,
- eye cup—16.27%

Each of the segmentation algorithms was earlier parameterised to allow the automation of the whole process.

USS can be easily extended to other areas of medical image processing as for example MRI or the colour image segmentation in general. We are now in the process of investigating application of USS in segmentation of colour outdoor images. Advantages of using USS are that the knowledge can be easily changed by proposing new region filtering rules as well as the PNC2 system gives the flexibility of verifying the final results of segmentation by including the ground truth images information. USS is a part of the knowledge controlled segmentation approaches—a fast growing branch of image processing techniques.

References

1. Bezdek J. C.: *FCM: The fuzzy c-means clustering algorithm. Computers and Geosciences,* **10** (1984) 191–203.
2. Budde J. B., Panda-Jonas W. M.: *Ophthalmoscopic Evaluation of The Optic Nerve Head. Survey of Ophthalmology,* **43(4)** (1999) 293–320.
3. Bruijne M., Ginnehen B.: *Automated segmentation of abdominal aortic aneurysms in multi spectral MR images,* UU-CS (Ext. r. no. 2003-014). Utrecht: Utrecht University: Information and Computing Sciences. (2003).
4. Chaudhuri S., Chatterjee S., Katz N., Nelson M., Goldbaum M.: *Detection of Blood Vessels in Retinal Images Using Two-Dimensional Matched Filter,* IEEE Transactions on Medical Imaging. **8(3)** (1989) 263–269.
5. Cheng H. D., Jiang X. H., Sun Y., Wang J.: *Color Image Segmentation: Advances and Prospects,* Pattern Recognition. **34(3)** (2001) 2259–2281.
6. Clark M. C., Hall L. O., Goldgof D. B., Clarke L. P., Velthuizen R. P., Silbiger M. S.: *MRI Segmentation using Fuzzy Clustering Techniques,* IEEE Engineering In Medicine And Biology. **13(5)** (1994) 730–742.
7. Clark M. C., Hall L. O., Goldgof D. B., Velthuizen R., Murtagh F. R., Silbiger M. S.: Automatic Tumor Segmentation Using Knowledge-Based Techniques, IEEE Transactions on Medical Imaging. **17(2)** (1998) 187–201.
8. Gonzalez R. C., Woods R. E.: Digital Image Processing, Prentice-Hall. (2002).
9. Haendel L.: *Data-based learning of fuzzy-rules using an agglomerative cluster algorithm,* In Proc. European Symposium on Intelligent Technologies, Hybrid Systems and their implementation on Smart Adaptive Systems (Eunite). Puerto de la Cruz, Tenerife, Spain. (2001).
10. Hussein K., Agarwal A.: *A Knowledge Based Segmentation Algorithm for Enhanced Recognition of Handwritten Courtesy Amounts,* International Journal of Pattern Recognition. **32(2)** (1999).
11. Kanski J.: *Glaucoma. A color manual of diagnosis and treatment. Butterworth-Heinemann,* Oxford (1996).

12. Kovalevsky V.: *On the topology of Digital Spaces,* Proc. of the Seminar "Digital Images Processing". Technical University of Dresden (1986).
13. Li H., Chutatape O.: *Automatic Location of optic disk in retinal images,* Proc. International Conference on Image Processing ICIP. (2001).
14. Lim Y. W., Lee S. V.: *On the color images segmentation based on the thresholding and the fuzzy c-means techniques. Pattern Recognition,* **23(9)** (1990) 935–952.
15. Linying S., Sharp B., Chibelushi C.: *Knowledge-Based Image Understanding, A rule-based production system for X-ray segmentation,* Proc. of Fourth International Conference on Enterprise Information Systems 1. Cindad Real, Spain (2002) 530–533.
16. Michalski R. S., Kaufman K. A.: *Learning Patterns in Noisy Data: The AQ Approach. Machine Learning and Its Applications,* Springer (2001).
17. Nakagawa Y., Rosenfeld A.: *Some experiments on variable thresholding. Pattern Recognition,* **11(3)** (1979) 191–204.
18. Newman T. S., Tang N., Dong C., Choyke P.: *Slice adaptive histogram for improvement of anatomical structure extraction in volume data,* Pattern Recognition Letters **23(1)** (2002) 25–35.
19. Nazif A. M., Levine M. D.: *Low level image segmentation: An Expert System,* IEEE Transactions on Pattern and Machine Inteligence. **6(5)** (1984) 555–577.
20. Osareh A., Mirmehdi M., Thomas B., Markham R.: *Classification and Localisation of Diabetic-Related Eye Disease,* The 7th European Conference on Computer Vision (ECCV). Springer LNCS **2353** (2002) 502–516.
21. Pakzad K., Heipke Ch.: *Knowledge-Based Interpretation of Moorland in Aerial Images,* IAPRS, Amsterdam (2000).
22. Sinthanayothin Ch.: *Image Analysis for Automatic Diagnosis of Diabetic Retinopathy,* PhD Thesis. King's College London (1999).
23. Stapor K., Pawlaczyk L. M., Chrastek R., Michelson R.: *Automatic detection of glaucomatus changes using adaptive thresholding and neural networks. Proceedings of ICCS,* Cracow (2004).
24. Stapor K., Pawlaczyk L. M., Chrastek R., Niemann R., Michelson R.: *Automatic segmentation and classification of fundus eye images for glaucoma diagnosis,* Workshop Bildverarbeitung Fur Die Medizin. Berlin (2004).
25. Zhang M., Hall L. O., Goldgof D. B.: *A Generic Knowledge-Guided Image Segmentation and Labeling System Using Fuzzy Clustering Algorithms,* IEEE Trans. on Systems, Man, and Cybernetics. **32(5)** (2002) 571–582.
26. Zhang Y. J.: *A Survey on Evaluation Methods For Image Segmentation,* Pattern Recognition. **29(8)** (1996) 1335–1346.

Proceedings of the International Multiconference on
Computer Science and Information Technology pp. 437–446
ISSN 1896-7094
© 2007 PIPS

Genetic Programming for Dynamic Environments

Zheng Yin[1], Anthony Brabazon[1], Conall O'Sullivan[2], Michael O'Neill[1]

[1] Natural Computing Research and Applications Group
University College Dublin, Ireland.
zheng.yin@ucdconnect.ie Anthony.Brabazon@ucd.ie M.ONeill@ucd.ie
[2] School of Business, University College Dublin, Ireland.
conall.osullivan@ucd.ie

Abstract. Genetic Programming (GP) is an automated computational programming methodology which is inspired by the workings of natural evolution techniques. It has been applied to solve complex problems in multiple application domains. This paper investigates the application of a dynamic form of GP in which the probability of crossover and mutation adapts during the GP run. This allows GP to adapt its diversity-generating process during a run in response to feedback from the fitness function. A proof of concept study is then undertaken on the important real-world problem of options pricing. The results indicate that the dynamic form of GP yields better results than are obtained from canonical GP with fixed crossover and mutation rates. The developed method has potential for implementation across a range of dynamic problem environments.

1 Introduction

One of the most studied evolutionary methodologies is that of genetic programming (GP) [3]. GP is a population-based search algorithm. It starts from a high-level statement of what is required and automatically creates a computer programme to solve the problem. GP belongs to the field of *Evolutionary Automatic Programming*. The term is used to refer to systems that adopt evolutionary computation to automatically generate computer programmes. More generally, a computer programme can be considered as a list of rules or as a model.

Many of the most interesting real-world problems arise in a dynamic environment, one where the underlying fitness landscape and the associated optimal solution, changes over time. The challenge in tackling dynamic problems using evolutionary approaches include the making of good parameter choices for the evolutionary algorithm, as well as maintaining sufficient diversity in the population of solutions during the run. In the context of GP, many algorithm design choices are open to the modeller including the form and rate of mutation; the form and rate of crossover; the form and pressure of selection mechanism; the form of replacement mechanism; and the size of population. Choices for these items can impact critically on the algorithm's performance. Good settings for

one problem will not be appropriate for another. Indeed, good settings will not be static during a single GP run. This paper investigates a component of this problem, through application of a dynamic form of GP in which the probability of crossover and mutation adapts during the GP run. This permits GP to adapt its diversity-generating process during a run in response to feedback from the fitness function.

Recent years have seen the application of multiple biologically-inspired algorithms for the purposes of financial modelling [1]. In this study, we test the utility of the new GP system by applying it to the important real-world problem of options pricing. Due to the complexity in developing closed form theoretical models for options pricing, the domain is particularly amenable to techniques such as GP. One of the early applications of GP to the option pricing problem is provided by [6]. Since then there have been many methodological improvements such as seeding the initial population with elements drawn from the Black-Scholes option pricing formula, and the combination of other domain knowledge into the GP set of terminals / non-terminals [7].

2 Dynamic Genetic Programming

As already discussed in sect. 1, there are multiple decision choices open to the modeller when applying GP. Even when attention is restricted to the choice of good crossover and mutation rates, a non-trivial problem results, as these parameters will typically interact.

The parameter choices for crossover and mutation are clearly critical in ensuring a successful GP application. They impact on populational diversity and the ability of GP to escape from local optima. The parameter settings are also linked to the complexity of the problem and the size of the population. If the search space is large and/or the population size is relatively small, then the mutation rate will typically need to increase.

A common approach in tuning a GP is to undertake a series of trial and error experiments before making parameter choices for the final GP runs. However, this approach is highly problematic as it can be time consuming and it is impractical to test all parameter choices. Another issue is that good choices for parameters such as crossover and mutation rates are unlikely to remain constant over the entire run. Rather than selecting static parameter values, another approach is to dynamically adapt the parameters during the run. Three broad methods of such adaptation exist (see Fig. 1) [2]. Deterministic methods of parameter control vary parameter settings during the GP run, without using any feedback from the search process. Under a feedback adaptive process, the parameter values are adapted based on feedback from the algorithm. For example, if the intent was to increase the level of diversity generation once the population has converged to a threshold level (perhaps measured using the entropy of the population structures), once such convergence is detected, the mutation rate could be increased by x%. Another possibility is to evolve good choices for

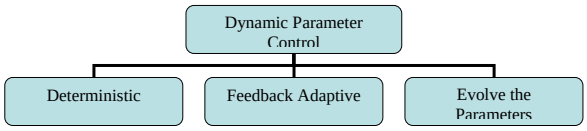


Fig. 1. Taxonomy of adaptive parameter control

its parameters dynamically during the run, which is called self adaptation. Our adaptive GP belongs to the second type.

There are comprehensive discussions about Evolutionary Algorithms' parameter adaptation in [19] and [18]. Most discussions of operator adaptation for evolutionary algorithms are found in the Genetic Algorithm literature. Studies in [9] and [26] provide the earlier discussions of this topic. [9] adapted the probability of operator according to a method favoring operators produced fitter children. In [26], the mutation probability is changed by a deterministic method. Relevant GA literature includes adaptation of parameters based on fitness feedback, such as [15], [14] and[11]. [10] based the feedback on population diversity. This application achieved the twin goals of population diversity and convergence capacity by adapting the probability of crossover and mutation. [16], [13] and [12] used self adaptive to evolve the parameters.

In the recent applications, the operator adaptation in GA has been more refined and complicated. They have not only combined more sophisticated population statistics into the feedback control variables,but also combined with other optimization methods. [21] explicitly adapted the mutation rate for each gene locus by the information of gene-based allele distribution. [22] further included the gene-based fitness into the control information. [20] explicitly adapted the operator probability according to mutation and crossover matrices which include fitness ranking, loci standard deviation of allele distribution and hamming distance of chromosome. [24] inferences of probability of crossover and mutation by a fuzzy-based system, which fuzzifies the relative sizes of the clusters containing the best and worst chromosomes. The above applications all get positive results based on their tested problems. However, there are also some studies try to compare the adaptive methods across different difficulty problems.[23] get the conclusion that in a variety of circumstances self-adaptation fails to allow the GA to perform better on some test suite than fixed mutation. At an earlier stage, it was concluded in [17] that successful adaptation has to combine the characters of real world problems,to detect and overcome the pitfall from them. Since different real world problems have different difficulties to be solved they need different adaptation method. In this application we leave the proposed adaptation method in the market option pricing equivalent difficulty environment.

It is important to note that findings in the GA literature will not necessarily carry over to GP. For example, in the GA, crossover and mutation operate on an underlying genotype, normally a fixed length string, whereas in GP they operate

directly on the phenotype, a popular type is syntax tree. Tree diversity measure is much harder to define and time consuming to calculate. Hence, although the concepts of crossover, mutation and selection in GA and GP are similar, their application produces different effects. There has also been a different history in the use of crossover and mutation between GA and GP. Typically, although the mutation rate in GA is small, mutation is considered to play a vital role in diversity generation. In contrast, early applications of GP emphasised the use of sub-tree crossover, rather than sub-tree mutation. However, in spite of the above, the GA literature on dynamic adaptation of parameter settings does provide a good foundation for thinking about these issues in GP. [25] proposed a metric to reflect tree structure difference and used fitness sharing to dynamically control GP's population diversity however the probability of operator kept fixed. The study of [27]adapted the probabilities of operator in a graph represented GP based on three methods related to operator successful rates, parents' fitness and operator successful history at individual level. Our adaptive method is based on feedback from the fitness function. This is supplemented by considering how long it has been since a new best solution was last uncovered. We term this period *the generation gap*. Compared with other adaptations this method does not involve extra calculation and memory cost though it returned improved results.

3 Applying GP for Options Pricing

In applications of GP to options pricing, the objective is to uncover an underlying option pricing model, using market data. The utility of the model is tested by comparing the quality of its predictions against real market option prices.

In this study, data is drawn from market option prices on the FTSE 100 futures index on the 17th March 2006. There are 187 different end-of-day settlement implied volatilities quoted for various strike prices and maturities. The option *moneyness* (defined as the the underlying asset price divided by the strike price) in our 187 data points varies from 0.77 to 1.43, the time-to-maturity varies from 35 days to 5754 days and the option price varies from 1.5E-12 to 4295. Market call option prices are calculated by substituting the implied volatilities into the Black-Scholes formula. The remaining factors such as the underlying asset price, the strike price and time-to-maturity are observed. The object of the GP application is to predict the 187 option prices given the explanatory variables. Among these 187 data points, 165 shorter dated options are used for in sample fitting and the remaining 22 longer dated options, which have time to maturity ranging 3199 to 5754 days and *moneyness* levels ranging from from 0.77 to 1.43 are used as the out of sample test. Thus information on shorter dated options is used to predict longer dated options using market option data. In selecting variables for inclusion as terminals, we used domain knowledge [7] to include option moneyness and implied volatility during the life of the option (table 1). The implied volatilities σ_{Mean}, σ_{Max} and σ_{Min} in table 1 are calculated using the shorter dated options with maturities ranging from 35 days to 553 days. We choose options with time to maturity from 35 days to 553 days as they are

shorter term options, which are most actively traded in the market. The non-terminal set is as listed in table 2. The predictive target is the entire range of option prices and in particular the out-of-sample longer dated option prices.

Table 1. Terminal set

Variables	*Expression*	*Definition*
X1	S_0	Asset price
X2	S_0/K	Asset price / Strike price
X3	T	Time to maturity
X4	r	Risk free rate
X5	σ_{Mean}	Mean implied volatility from option with T from 35 to 553 days
X6	σ_{Max}	Maximum implied volatility from option with T from 35 to 553 days
X7	σ_{Min}	Minimum implied volatility from option with T from 35 to 553 days
X8	$\sigma_{Mean} * \sqrt{T}$	Implied asset price volatility during option's life by σ_{Mean}
X9	$\sigma_{Max} * \sqrt{T}$	Implied asset price volatility during option's life by σ_{Max}
X10	$\sigma_{Min} * \sqrt{T}$	Implied asset price volatility during option's life by σ_{Min}

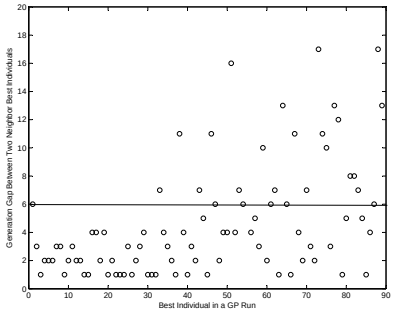
Table 2. Non-terminal set

Expression Sign	*Definiton*
+	Addition
-	Subtraction
*	Multiplication
x/y	Protected division, if y=0 then x/y= x; else x/y=x/y
$\log(x)$	Protected natural logarithm, if x=0 then $\log(x)$=0; else $\log(x)$=$\log(\sqrt{x^2})$
$\sqrt{x}$	Protected square root, if $x \leq 0$ then $\sqrt{x}$=0; else $\sqrt{x}$=$\sqrt{x}$
$N(x)$	Accumulated normal distribution
e^x	Exponential function

4 Experimental design

In previous applications of GP to options pricing, the probabilities of crossover and mutation were typically kept constant. For example, in [6] mutation is applied at a probability of 0.0033 and population of 500,with a mutation probability of 0.001 and population of 2000 being applied by [5] and a mutation probability of 0.001 and a population size of 50 being applied by [8]. [4] investigated the utility of various mutation rates between 0.1 to 0.5 (each of which was constant in a single run) with a population from 100 to 50,000.

As already noted, in this study we employ ten explanatory variables, see table 1. There are also eight functions in the non-terminal set. From the Black-Scholes formula we know the tree level should be around 11. This implies that we are searching for a global solution (the model) in a space of round 18^{11}, clearly a challenging task. Based on initial experiments, it was found (as expected)



(a) Constant Probability

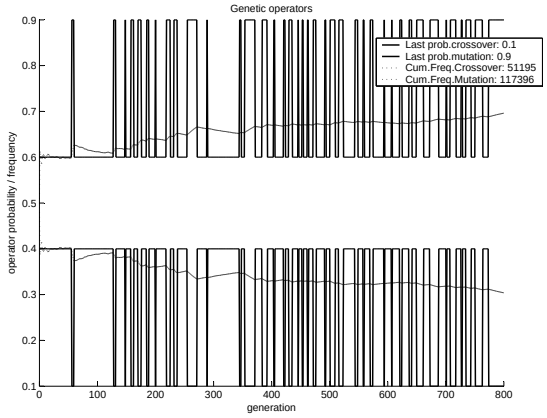
Fig. 2. The Generation Gaps Between Neighbor Best Individuals

that the best individual changes frequently in early generations, with the rate of change slowing down later in the run. Generation gaps between new best individuals are plotted in figure 2(a). The points above the line in the graph indicate the generation gap between subsequent best individuals are more than six generations. Based on these results we adapted the operator probabilities according to this generation gap.

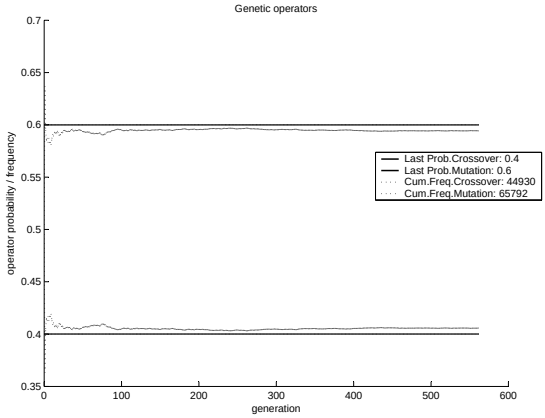
In the dynamic GP the mutation probability is increased at a constant rate whenever the best individual is unchanged over several generations and the mutation probability is changed to the normal rate when a new best individual is uncovered. This allows the search process to adapt to escape local optima, whilst permitting local improvement around just discovered new solutions. In this study, the width of the window is set at six generations. We are assuming six generations are plenty for GP to exploit the just explored area. In other words, mutation is fixed for the six generations following the uncovering of a new best solution. After six generations without finding a new best individual, the mutation probability increased until either it reaches 0.9 or alternatively, a new best solution is uncovered. Figures 3(a) and 3(b) illustrate the adaption of crossover and mutation rates during a sample GP run.

A total of twenty GP runs were undertaken, ten of which were fixed parameter GP runs and ten of which were dynamic parameter GP runs. The fixed parameters for crossover and mutation were 0.4 and 0.6 respectively, set after some initial trial and error experiments. In the adaptive experiments, the parameters for crossover and mutation are initially set to 0.4 and 0.6. If six generations have elapsed and the best individual has not changed this means the population is perhaps too concentrated and new area need to be explored hence the mutation rate is increased by 0.02 per generation, with crossover decreasing by 0.02 each time until limits of 0.9 and 0.1 are reached. Once a new best individual appears the mutation and crossover probabilities are restored to their initial values of 0.6 and 0.4.

For all the above experimental runs, ramped half-half initialization is employed. A roulette parental selection strategy, with a replacement strategy of half elitism (which means half of the new population will be filled by the best from both parent and children and the remaining places will be left to the best children), is also employed. The population size is fixed at 300. The GP run is terminated either when there has been no performance improvement for 40 generations, or when a maximum number of generations is hit (800 generations).



(a) Dynamic Probability



(b) Constant Probability

Fig. 3. Probability of Crossover and Mutation in a GP Run

5 Results

The results from the twenty GP runs are provided in tables 3 and 4. The in sample average absolute error from the constant parameter GP is 12.7% higher than the dynamic parameter GP counterpart, and the average percentage error is about 4.5% higher. The out sample average absolute error from the constant parameter GP is 231.4% higher than the dynamic parameter GP counterpart, and the average percentage error is 240.5% higher.

It can be seen from graphs 3(b) and 3(a), the ratio of the cumulated mutation frequency and crossover frequency in constant GP, where it is near 6:4, is much lower than it in dynamic GP, where it is near 7:3. The extra mutations in the second case are distributed in different searching periods. By this way GP effectively escape the local optima and explore more searching space, hence improve the performance.

Table 3. The results from the constant probability setting with mutation 0.6 and crossover 0.4. A.E., the absolute error calculated as the absolute value of the difference between option price returned by GP and the market option price P.E., the percentage error calculated as dividing the absolute error(A.E.) by the market price

Test Index	1	2	3	4	5	6	7	8	9	10	Average of 1-10
In sample											
A.E.	54.8	78.8	82.1	99.2	99.8	99.9	100	109	116	116	95.56
P.E. (%)	29.3	31.5	32.9	37.1	38.2	41.4	13.6	42.1	37.2	46	34.9
Out sample											
A.E.	151	915	333	317000	257	37300	476	8110	5890	574	37100.6
P.E. (%)	4.8	29.3	10.4	9070	7.9	1000	13.6	223	159	19	1053.7

Table 4. The results from the adaptive GP, if there is no improvement in 6 generations the mutation probability will increase by 0.02 each time until a max of 0.9 and drop back to 0.6 when a new best individual appears. A.E., P.E. as in the above table

Test Index	1	2	3	4	5	6	7	8	9	10	Average of 1-10
In sample											
A.E.	42.7	64.9	71.2	71.3	72.2	89.8	101	102	116	117	84.81
P.E. (%)	26.7	31.5	31.2	29.9	26.9	32.8	34.7	45.1	35.2	40.1	33.41
Out sample											
A.E.	465	143	1570	51400	432	505	1060	1000	5780	49600	11195.5
P.E. (%)	13.6	4.5	46.2	1440	14.4	17.4	31.6	28.6	158	1340	309.4

6 Conclusions

This paper illustrates the application of a novel dynamic form of GP, where the probability of crossover and mutation is adapted during the GP run, to the

important real-world problem of options pricing. The tests are carried out using market option price data and the results illustrate that the new method yields better results than are obtained from GP with fixed crossover and mutation rates.

It is noted that this dynamic GP method improves the performance without extra calculation costs. This method has potential for implementation across a wide range of dynamic problem environments and it is intended to test the utility of the methodology on a variety of non-financial dynamic problems. In future application, window size could be determined dynamically during the GP run. Future work also includes correcting the biases in the Black-Scholes options pricing model by applying GP to recover market option prices across a range of possible explanatory variables and thereby examining the interrelationships among option prices that goes beyond the Black-Scholes pricing formula.

References

1. Brabazon A.,O'Neill M. (2006). *Biologically Inspired Algorithms for Financial Modelling*, Springer, Berlin. 311–327.
2. Eiben A.E.,Smith J.E. (2003). *Introduction to evolutionary computing*, Springer, Berlin.
3. Koza, J.R. (1992). *Genetic programming—on the programming of computers by means of natural selection*, MIT Press.
4. Chidambaran N. K. (2003). Genetic Programming with Monte Carlo Simulation for Option Pricing, *Proceedings of the 2003 Winter Simulation Conference*, Vol.1, pp. 285-292
5. Noe T. H., Wang J. (2002). The Self-Evolving Logic Of Financial Claim Prices in *Genetic algorithms and genetic programming in computational finance*, Kluwer Academic Publishers, pp. 249–279.
6. Chen S. H., Yeh C. H. and Lee W. C. (1998). Option Pricing with Genetic Programming, in *proceedings of the Third Annual Genetic Programming Conference*, Morgan Kaufmann Publishers, San Francisco, CA, pp. 32–37.
7. Yin Z., Brabazon A. and O'Sulivan C. (2006). Genetic Programming and Option Pricing, in proceedings of the 2006 Annual Irish Accounting & Financial Association Conference.
8. Keber C. (2002). Evolutionary Computation in Option Pricing: Determining Implied Volatilities Based on American Put Options, in *Evolutionary Computation In Economics and Finance*, Physica-Verlag Heidelberg, pp. 399–415.
9. Davis L. (1989). Adapting Operator Probabilities in Genetic Algorithms, in *proceedings of the Third International Conference on Genetic Algorithms*, Morgan Kaufmann, San Mateo, CA, pp. 61–69.
10. Srinivas M., Patnaik L. M. (1994). Adaptive Probabilities of Crossover and Mutation in Genetic Algorithms, *IEEE Transactions on Systems, Man and Cybernetics*, Vol. 24, No. 4, April 1994, pp. 656–667.
11. Ho C. W., Lee K. H. and Leung K. S. (1999). A Genetic Algorithm Based on Mutation and Crossover with Adaptive Probabilities. In *Proceedings of the 1999 Congress on Evolutionary Computation, 1999. CEC 99.*, Vol. 1, pp. 768–775.
12. Boomsma W. (2004). A Comparison of Adaptive Operator Scheduling Methods on the Traveling Salesman Problem, *Evolutionary Computation in Combinatorial Optimization*, Springer Berlin, pp. 31–40.

13. Zhang J., Chung H. S. H., Hu B. J. (2004). Adaptive Probabilities of Crossover and Mutation in Genetic Algorithms Based on Clustering Technique in *Proceedings of the Congress on Evolutionary Computation, 2004. CEC2004.*, 19–23 June 2004, Vol. 2, pp. 2280–2287.
14. Julstrom B. A. (1997). Adaptive Operator Probabilities in A Genetic Algorithm that Applied Three Operators. *Proceddings of the 1997 ACM Symposium on Applied Computing*, ACM Press, pp. 233–238.
15. Julstrom B. A. (1995). What have you done for me lately? Adapting operator probabilities in a steady-state genetic algorithm, in *proceedings of the Sixth International Conference on Genetic Algorithms*, San Francisco, CA, Morgan Kaufmann, pp. 81–87.
16. Spears W. M. (1995). Adaptive Crossover in Evolutionary Algorithms, in *Proceedings of the 4th Annual Conference on Evolutionary Programming*, Cambridge, MA, MIT Press, pp. 367–384.
17. Tuson A., Ross P. (1998). Adapting Operator Settings in Genetic Algorithms. in *Evolutionary Computation* (1998), Vol. 6, pp. 161–184.
18. Eiben A. E., Hinterding P., Michalewicz Z. (1999). Parameter Control in Evolutionary Algorithms in *IEEE Transaction on Evolutionary Computation*, Vol. 3, No. 2, July 1999, pp. 124–141.
19. Angeline P. J. (1995) Adaptive and self-adaptive evolutionary computation in *Computational Intelligence: A Dynamic System Perspective*, New York: IEEE Press, 1995, pp. 152–161.
20. Law N. L., Szeto K. Y. (2007) Adaptive genetic algorithm with mutation and crossover matrices in *Twentieth International Joint Conference on Artificial Intelligence*, Hyderabad, India, Jan. 2007, pp. 2330–2333.
21. Yang S. X. (2003) Adaptive mutation using statistics mechanism for genetic algorithms in *Research and Development in Intelligent Systems*, London: Springer-Verlag, 2003, pp. 19–32
22. Yang S. X. and Uyar S. Adaptive Mutation with Fitness and Allele Distribution correlation for Genetic Algorithms in *Proceedings of the 21st ACM Symposium on Applied Computing,* ACM Press 2003, pp. 940–944.
23. Rand W. and Riolo R. (2005) The problem with a Self-Adaptative mutation rate in some environments: A Case Study Using the Shaky Ladder Hyperplane-Defined Functions in *Proceedings of the 2005 Conference on Genetic and Evolutionary Computation*, New York: ACM Press, 2005, pp. 1493–1500.
24. Zhang J., Chung H. S. and Zhong J. (2005) Adaptative Crossover and mutation in Genetic Algorithms Based on Clustering Technique in *Proceedings of the 2005 Conference on Genetic and Evolutionary Computation,* New York: ACM Press, 2005, pp. 1577–1578.
25. Ekart A. and Nemeth S. Z. (2002) Maintaining the Diversity of Genetic Programs in *Proceedings of the 5th European Conference on Genetic Programming*, Lecture Notes In Computer Science, vol. 2278. Springer-Verlag, London, pp. 162–171.
26. Fogarty T. (1989) Varying the probability of mutation in the genetic algorithm in *Proc. 3rd Int. Conf. Genetic Algorithms*, Morgan Kaufmann Publishers, San Francisco, CA, 1989, pp. 104–109.
27. Niehaus J. and Banzhaf W. (2001) Adaption of Operator Probabilities in Genetic Programming in *Proceedings of the 4th European Conference on Genetic Programming*, Lecture Notes In Computer Science, vol. 2038. Springer-Verlag, London, pp. 325–336.

Computer Aspects of Numerical Algorithms (CANA'07)

Numerical algorithms are widely used by scientists engaged in various areas. There is a special need of highly efficient and easy-to-use scalable tools for solving large scale problems. The workshop is devoted to numerical algorithms with the particular attention to the latest scientific trends in this area and to problems related to implementation of libraries of efficient numerical algorithms. The goal of the workshop is meeting of researchers from various institutes and exchanging of their experience, and integrations of scientific centers.

Topics*:*

- Parallel, distributed and grid numerical algorithms
- Data structures for numerical algorithms
- Libraries for numerical computations
- Numerical algorithms testing and benchmarking
- Analysis of rounding errors of numerical algorithms
- Optimalization of numerical algorithms
- Languages, tools and environments for programming numerical algorithms
- Paradigms of programming numerical algorithms
- Contemporary computer architectures
- Applications of numerical algorithms in science and technology

International Programme Committee

- Pierluigi Amodio, Universita di Bari, Italy
- Krzysztof Banas, AGH University of Science and Technology, Poland
- Luigi Brugnano, Universita degli Studi, Firenze, Italy
- Vasile Berinde, North University of Baia Mare, Romania
- Tim Davis, University of Florida, USA
- Ivan Dimov, The University of Reading, UK
- Graeme Fairweather, Colorado School of Mines, USA
- Salvatore Filippone, Universita' di Roma "Tor Vergata", Italy
- Domingo Giménez Cánovas, University of Murcia, Spain
- Ian Gladwell, Southern Methodist University, USA
- Andreas Griewank, Humboldt-Universität zu Berlin, Germany
- Stanislaw Grzegórski, Lublin University of Technology, Poland
- Andreas Karageorghis, University of Cyprus, Cyprus
- Erricos Kontoghiorghes, University of Cyprus, Cyprus and Birkbeck College, UK
- Jerzy Kozicki, Maria Curie-Sklodowska University, Poland
- Stanislaw Kozielski, Silesian University of Technology, Poland
- Anna Kucaba-Pietal, Rzeszow University of Technology, Poland
- Vyacheslav Maksimov, IMM RAS and Ural State Technical University, Russia
- Svetozar Margenov, Institute for Parallel Processing, BAS, Bulgaria
- Dana Petcu, Western University of Timisoara, Romania
- C J Kenneth Tan, OptimaNumerics Ltd., UK
- Luciano Tarricone, University of Salento, Italy
- Roman Trobec, Jozef Stefan Institute, Slovenia
- T.E. Simos, University of Peloponnese, Greece
- Marek Stanuszek, Cracow University of Technology, Poland
- Marian Vajtersic, Salzburg University, Austria
- Jerzy Wasniewski, Technical University of Denmark, Denmark

Organizing Committee

Przemyslaw Stpiczynski, Beata Bylina, Jaroslaw Bylina, Maria Curie-Sklodowska University, Poland

Proceedings of the International Multiconference on
Computer Science and Information Technology, pp. 449 – 455
ISSN 1896-7094
© 2007 PIPS

Applications of Finite Element Method in Synovial Joint Numerical Calculations

Jarosław Sęp, Anna Kucaba-Piętal,

Rzeszow University of Technology,
The Faculty of Mechanical Engineering and Aeronautics
Powstańców Warszawy 8
35-959 Rzeszów, Poland

Abstract. The problem of meshing for numerical calculations of human rapid movement influence on the tribological features of a synovial joint was discussed. The calculations performed pointed out great effectiveness of the ADINA-F solver.

1. Introduction

Biomechanics aims at describing human movement under dynamic conditions. The hip joint is one of the elements of the human osteoarticular system that are the most exposed to overload-related degenerative changes. The use of computer methods to solve equations which model the phenomena taking place in the human body is getting more and more common and brings about better and better results. In the case of research into the hip joint biobearing, the methods provide the information which may be useful for medical sciences and endoprosthesis design.

The aim of the research whose results are presented in the paper was to explain how human effort movements affect the tribological properties of the biobearing. In a previous work we presented highly accurate finite element method calculations of synovial joint tribology using the ADINA-F package. To dynamically describe the synovial fluid behavior, the nonstationary Navier Stokes equations were used. The calculations were done making use of the Galerkine weighted residual method, the Newton-Raphson method and the Gauss elimination method. To perform the calculations tetrahedral mesh elements were applied.

In this paper we focus on problems with optimal meshing in the numerical hip flow simulation. Numerical calculations of tribological values of the biobearing during effort movements were carried out. They were modeled dynamically, i.e. the results

were obtained by solving nonstationary equations of synovial fluid flow in the hip joint by the finite elements method making use of an ADINA-F software package. The problem was solved for a simplified model of a hip joint, ignoring the properties of articular cartilage – its permeability and elasticity.

2. Problem formulation

The assumed geometry of the articular fissure was based on a normal joint measurements. The situations considered were those encountered during getting up or jumping, which was modeled by moving the articular surfaces to each other.

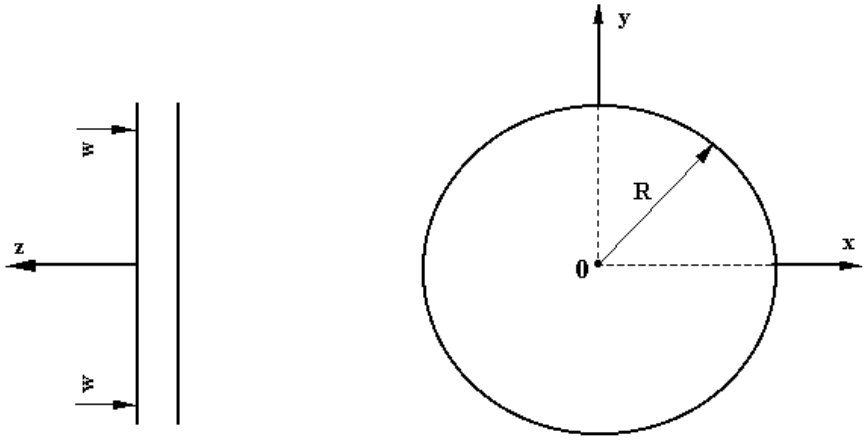


Fig1. Geometry of the flow.

The solution of the flow problem was obtained for a very much simplified geometrical model of the joint

- the flow area in the articular fissure was modeled as an area between parallel, circular plates,
- it was assumed that the surface of the plates is hard and impermeable (for the time being, elasticity and porosity of cartilage were neglected),
- initial (h_p) and final (h_k) heights of the lubrication fissure were assumed, supposing at the same time that the plate will displace from the initial to the final position in time t,
- mobile plate moves only along axis z

- as our purpose was to obtain qualitative results, the Newtonian model of synovial fluid was tentatively adopted.

The geometry of the flow is presented on Fig.1.Synovia flow in a channel was described by the Navier – Stokes system of equations, which, together with the equation of flow continuity for the analyzed problem is as follows:

$$\rho\left(\frac{\partial u}{\partial t}+u\frac{\partial u}{\partial x}+v\frac{\partial u}{\partial y}+w\frac{\partial u}{\partial z}\right)=-\frac{\partial p}{\partial x}+\eta\left(\frac{\partial^2 u}{\partial x^2}+\frac{\partial^2 u}{\partial y^2}+\frac{\partial^2 u}{\partial z^2}\right) \tag{2.1}$$

$$\rho\left(\frac{\partial v}{\partial t}+u\frac{\partial v}{\partial x}+v\frac{\partial v}{\partial y}+w\frac{\partial v}{\partial z}\right)=-\frac{\partial p}{\partial y}+\eta\left(\frac{\partial^2 v}{\partial x^2}+\frac{\partial^2 v}{\partial y^2}+\frac{\partial^2 v}{\partial z^2}\right) \tag{2.2}$$

$$\rho\left(\frac{\partial w}{\partial t}+u\frac{\partial w}{\partial x}+v\frac{\partial w}{\partial y}+w\frac{\partial w}{\partial z}\right)=-\frac{\partial p}{\partial z}+\eta\left(\frac{\partial^2 w}{\partial x^2}+\frac{\partial^2 w}{\partial y^2}+\frac{\partial^2 w}{\partial z^2}\right) \tag{2.3}$$

$$\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y}+\frac{\partial w}{\partial z}=0 \tag{2.4}$$

The boundary conditions are:

u=0, v=0, w=0 on the surface of the stationary plate, (2.5)

u=0, v=0, on the surface of the movable plate, (2.6)

p=0 on the edges of the stationary and movable plates, (2.7)

t-time of plate relocation from the initial to final position

(2.8)

where: **V=(u, v, w)** -synovial fluid velocity , **p** – pressure in the fluid layer [Pa]

The following value of parameters – resulting from the physiology of the joint – were used for the calculations:

h_p – initial height of the articular fissure, hp = 200 μm,

h_k – final height of the articular fissure, hk = 40 μm

t – time of the plate displacement from the initial to final position; six values of the parameter were used for the calculations: t_1 = 0,0005s, t_2 = 0,001s, t_3 = 0,003s, t_4 = 0,005s, t_5 = 0,0075s, t_6 = 0,01s,
R – radius of the plate, R=0.01m
η– dynamic viscosity of the synovial fluid, η=0,116 Pa·s,
ρ – synovial fluid density; ρ = 1000 kg/m^3.

3. Methodology of the calculations

The finite elements method and ADINA 8.1 software packages were applied to solve the (1)-(4) equations with (5)-(8) boundary conditions. The calculations were done making use of the Galerkine weighted residual method, the Newton-Raphson method and the Gauss elimination method. Tetrahedral-tetranodal elements were chosen to calculate the flow in the bearing.

The number of elements which is required for a given accuracy depends on the flow geometry. The mesh in ADINA can be generated either automatically or by the user. For the calculations the mesh thickened in the place of maximum pressure of the bioflow was created. Generated and tested and were four variants, each with a different number of elements.

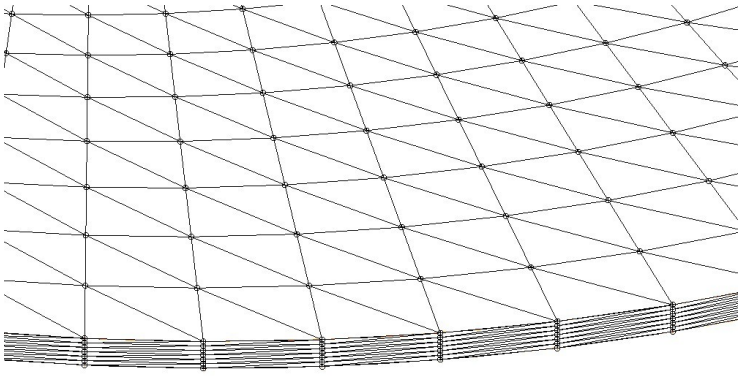


Fig2. Mesh plot.

The calculated values of physical quantities were accepted as a criterion for comparing the four variants with the analytical solutions They were:

- maximum pressure in the synovial film,

- maximum flow velocity values for synovia u_{max}, v_{max}

The results of the calculations are presented in the Table 1.

Table 1 Effect of Mesh density on calculations

Notation	Number of nodes	Number of elements	pmax [Pa]	u_{max} [m/s]	v_{max} [m/s]
S1	4448	19440	2,197·105	1,468	1.468
S2	8995	42240	2,099·105	1,580	1.580
S3	19803	100080	2,034·105	1,534	1.534
S4	28126	142560	2,034·105	1,534	1.534

An analysis of the results pointed out that the variant marked as S3 (19803 nods and 100080 elements) ensured the satisfactory accuracy of the calculation results.

4. Results

Numerical calculations of tribological values of the biobearing during effort movements were carried out. They were modeled dynamically, by six different times of the approaching plates. The resuls are presented in the Table 2. The obtained values of the calculations of the joint load are from the range of real values [1, 2] and show qualitative agreement with the calculations based on the micropolar model of synovial fluid [3].

Table 2. Calculations Results

approach time of the plates , t [s]	load capacity W [kN]	$p_{śr}$ · [MPa]	p_{max} [MPa]	u_{max} [m/]	v_{max} [m/]	w_{max} [m/s]
0,0005	28,5	90,43	181 ,29	61,33	61,33	0,325
0,0010	14,1	44,92	89,93	30,72	30,72	0,166
0,0030	4,68	14,85	29,80	10,23	10,23	0,055
0,0050	2,81	8,90	17,95	6,14	6,14	0,033
0,0075	1,87	5,93	11,89	4,09	4,09	0,022
0,0100	1,41	4,45	8,92	3,07	3,07	0,016

The illustrative example of pressure distribution is shown in Fig.3.

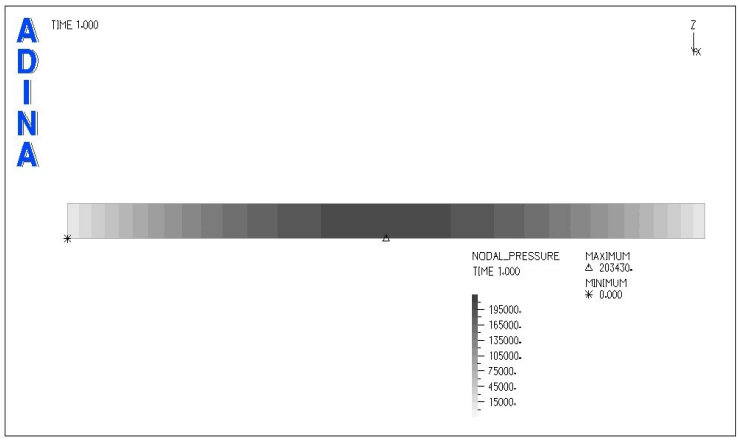


Fig. 3. Pressure distribution in the synovial film for t=0.001s.

Results shown that the approach time of the plates considerably affects the pressure in the joint and the flow rate of synovial fluid [3].

5. Conclusions

In this paper the problem of human rapid movement influence on the tribological features of a synovial joint was discussed. The flow in the joint was calculated numerically using the MES solver in an ADINA-F package.To dynamically describe the synovial fluid behaviour, the nonstationary Navier Stokes equations were used. The results show that rapid human movement results in increasing the tribological quantities of the synovial joint.

The obtained results show that rapid human movement results in increasing the tribological quantities of the synovial joint.

The calculations performed pointed out great effectiveness of the ADINA-F solver in such calculations. The use of the nonstandard mesh generator in ADINA enabled choosing the optimum mesh and the time step. The number of elements which is required for a given accuracy depends on the flow geometry.

References:

1. Dowson D: *Biomechanics of Diarthrodias,* vol. II. Springer-Verlag 1999.

2. Sęp J.: T*he oil film properties in journal bearings with non-typical pin geometry.* The Printing House of The Rzeszów University of Technology, Rzeszów 2006.

3. `www.medizin.fu-berlin.de/biomechanik/Reship1e.htm`;

4. Kucaba-Piętal A.: *Microflows modeling based on micropolar fluid theory,* The Printing House of The Rzeszów University of Technology, Rzeszów 2004.

5. Sęp J., Kucaba-Pietal A. Cwanek J,: *Premilinary calculation of the effect of articular cartilage fissure geometry on the flow in the hip joint,* Russian Journal of Biomechanics, tom 53 (4), 213-218, 2004.

6. Sęp J., Kucaba-Pietal A. Cwanek J,: *The Effect of Effort Movements on the Flow of Synovial Fluid,* in the Hip Joint Proceedings Mechanics in medicyn, Oficyna wydawnicza Politechniki Rzeszowskiej 2004.

7. Sęp J., Kucaba-Pietal A. Cwanek J, W. Liubimow: *Analiza topografii stawu biodrowego,* Acta of Bioengineering and Biomechanics. 3, 2, 471-476, 2001.

Proceedings of the International Multiconference on
Computer Science and Information Technology, pp. 457 – 465
ISSN 1896-7094
© 2007 PIPS

Molecular Dynamics computer simulation of nanoflows

Anna Kucaba-Piętal[1,3], Janusz Bytnar[3], Zbigniew Walenta[2,3]

[1]Rzeszow University of Technology,
The Faculty of Mechanical
Engineering and Aerodynamics,
Powstanców Warszawy 8,
35-959 Rzeszów, Poland
[2] Institute of Fundamenal
Technological Research PAN,
Department of Mechanics and
Physics of Fluids,
Swietokrzyska 21,
00-049 Warsaw, Poland
[3] Computer Science Institute,
PWSZ, Czarnieckiego 16
37-500 Jaroslaw Poland

Abstract. In this paper we present the results of utilizing scientific computing methodologies to address an engineering problem from nano technologies. In nano and micro-scale, the calculation could only be done with some particle based representation method. One of them is Molecular Dynamics (MD) method. In the paper we describe the construction of the Molecular Dynamics Method and we present some results of the MD simulation of the water nanoflows [13, 14].

Keywords: microflows, micropolar fluid, molecular dynamics simulation, direct Monte Carlo methods, microchannels.

1. Introduction

Numerical modeling and simulation is widely used in the practice of scientific research, engineering design and industrial applications throughout the world. Physical prototype models are now being improved and sometimes replaced with computer models, which have made significant economical and social impact. In many numerical simulations, the fundamental problems usually occur at the contact of the continuous model, consisting of the initial system of differential equations and boundary conditions, and the discrete model.

Computer simulation methods have become a very powerful tool to attack the many-body problem in statistical physics, physical chemistry and biophysics. Although the theoretical description of complex systems in the framework of statisti-

cal physics is well developed and the experimental techniques for detailed microscopic information are quite sophisticated, it is often only possible to study specific aspects of those systems in great detail via the simulation. On the other hand, simulations need specific input parameters, which characterize the system in question, and which come either from theoretical considerations or are provided by experimental data [1].

The traditional simulation methods for many-body systems can be divided into two classes of stochastic and deterministic simulations, which are largely covered by the Monte Carlo (MC) methods and the molecular dynamics (MD) methods, respectively. There are several methods utilizing stochastic approach to the problem, called in general "Monte Carlo methods". Some of them probe the configuration space by trial moves of particles. Within the so-called Metropolis algorithm, the energy change from step n to $n+1$ is used as a trigger to accept or reject the new configuration. Paths towards lower energy are always accepted, those to higher energy are accepted with a probability governed by Boltzmann statistics. In that way, properties of the system can be calculated by averaging over all Monte Carlo moves (where one move means that every degree of freedom is probed once on average).

By contrast, MD methods are governed by the system's Hamiltonian and consequently Hamilton's equations of motion are integrated to move particles to new positions and to get new velocities at these new positions. This is an advantage of MD simulations with respect to MC, since not only is the configuration space probed but the whole phase space which gives additional information about the dynamics of the system. Both methods are complementary in nature but they lead to the same averages of static quantities, provided that the system under consideration is ergodic and the same statistical ensemble is used[1].

The deterministic particle based method, molecular dynamics (MD) simulation, although theoretically valid for the whole range of conditions, is employed mainly for liquids, as the long flight paths between collision for gas molecules makes forward integration of the equations of motion prohibitively expensive. In liquids, the molecules are densely packed, leading to a more efficient application of MD simulation[2]. MD requires description of the molecules and the forces acting between them; a well known example is the Lennard-Jones potential, which describes spherically symmetric particles, repelling one another at close distances, but otherwise attracting.

Beginning in theoretical physics, the method of MD gained popularity in materials science and since the 1970s also in biochemistry and biophysics. In chemistry, MD serves as an important tool in protein structure determination and refinement using experimental tools such as X-ray crystallography and NMR. It has also been applied with limited success as a method of refining protein structure predictions. In physics, MD is used to examine the dynamics of atomic-level phenomena that cannot be observed directly, such as thin film growth and ion-subplantation. It is also used to examine the physical properties of nanotechnological devices that have not or cannot yet be created.

The motivation for this research stems from a real-world nanochannels problem. For flows of liquids in nanochannels the continuum approximation is not valid and

the Molecular Dynamic Simulation is the only appropriate solution some MD flow simulations results will be presented to illustrate usefulness of the method.

2. Molecular Dynamics Method

Molecular Dynamics it's an atomistic simulation method where:

- each atom is treated as a point mass,
- simple force rules describe the interactions between atoms
- Newton's equations are integrated to advance the atomic positions and velocities, and
- thermodynamic statistics are extracted from the motion of the atoms.

As pointed out in the introduction, Molecular Dynamics requires a description of the molecules and the forces that act between them. The MD simulation itself amounts to numerically integrating the equations of motion for systems of a few hundred to few million particles over many thousand (or more) timesteps. The paths the particles follow during the computation represent actual molecular trajectories.

Molecular dynamics is a computer simulation technique describing the time evolution of a set of interacting atoms. In molecular dynamics we follow the laws of classical mechanics,

$$F_i = m_i a_i \qquad (2.1)$$

for each atom i in system constituted by N atoms. Here $\mathbf{m_i}$ is the atom mass,

$$a_i = d^2 r_i / dt^2 \qquad (2.2)$$

its acceleration, and Fi the force acting upon it, due to the interactions with other atoms [3].

Molecular Dynamics is a deterministic technique: given an initial set of positions and velocities, the subsequent time evolution is in principle completely determined. In more pictorial terms, atoms “move” inside the computer, interacting with each other, wandering around (if the system is fluid), oscillating in waves, perhaps evaporating away from the system if there is a free surface, and so on, in a way similar to what atoms in a real substance would do.

If we denote the force exerted by atom α of molecule i on atom β of molecule j by $f_{i\alpha j\beta}$ then the total force acting on molecule i is

$$F_i = \sum_j \sum_\beta \sum_\alpha f_{i\alpha j\beta} \qquad (2.3)$$

and the torque is given by

$$N_i = \sum_{\alpha} \left(r_{i\alpha} - R_i \right) \times f_{i\alpha} \tag{2.4}$$

where

$$R_i = 1 / M_i \sum_{\alpha} m_{i\alpha} r_{i\alpha} \tag{2.5}$$

is the centre of mass of molecule i.

The motion is governed by the Newton-Euler equations

$$M_i \ddot{R}_i = F_i \tag{2.6}$$

$$I_i \cdot \dot{\omega}_i - \omega_i \times I_i \cdot \omega_i = N_i \tag{2.7}$$

where ωi is the anlegular velocity of the molecule,

$$I_i = \sum_{\alpha} m_{i\alpha} \left(p_{i\alpha}^2 1 - p_{i\alpha} p_{i\alpha} \right) \tag{2.8}$$

is the inertia tensor and

$$p_{i\alpha} = r_{i\alpha} - R_i \tag{2.9}$$

is the atomic site co-ordinate relative to the centre of mass.

The orientation of the molecules can be represented by quaternions as has now become common practice. They are preferred over Euler angles for two reasons. Firstly they lead to equations of motion which are free of singularities which means that no special-case code is required. This leads to much improved numerical stability of the simulation. Secondly, molecular symmetry operations and combinations of rotations are elegantly expressed in terms of a simple quaternion algebra [4].

3. Molecular Dynamics Algorithm

The engine of a molecular dynamics program is its integration algorithm, required to integrate the equations of motion of the interacting particles and follow their trajectories. Integration algorithms are based on finite difference methods, with discretized time and the time step equal to $\Delta \mathbf{t}$. Knowing the positions and some of their time derivatives at time t (the exact details depend on the type of algorithm), the

integration scheme gives the same quantities at a later time (**t + Δt**). With such procedure, the evolution of the system can be followed for long times [3].

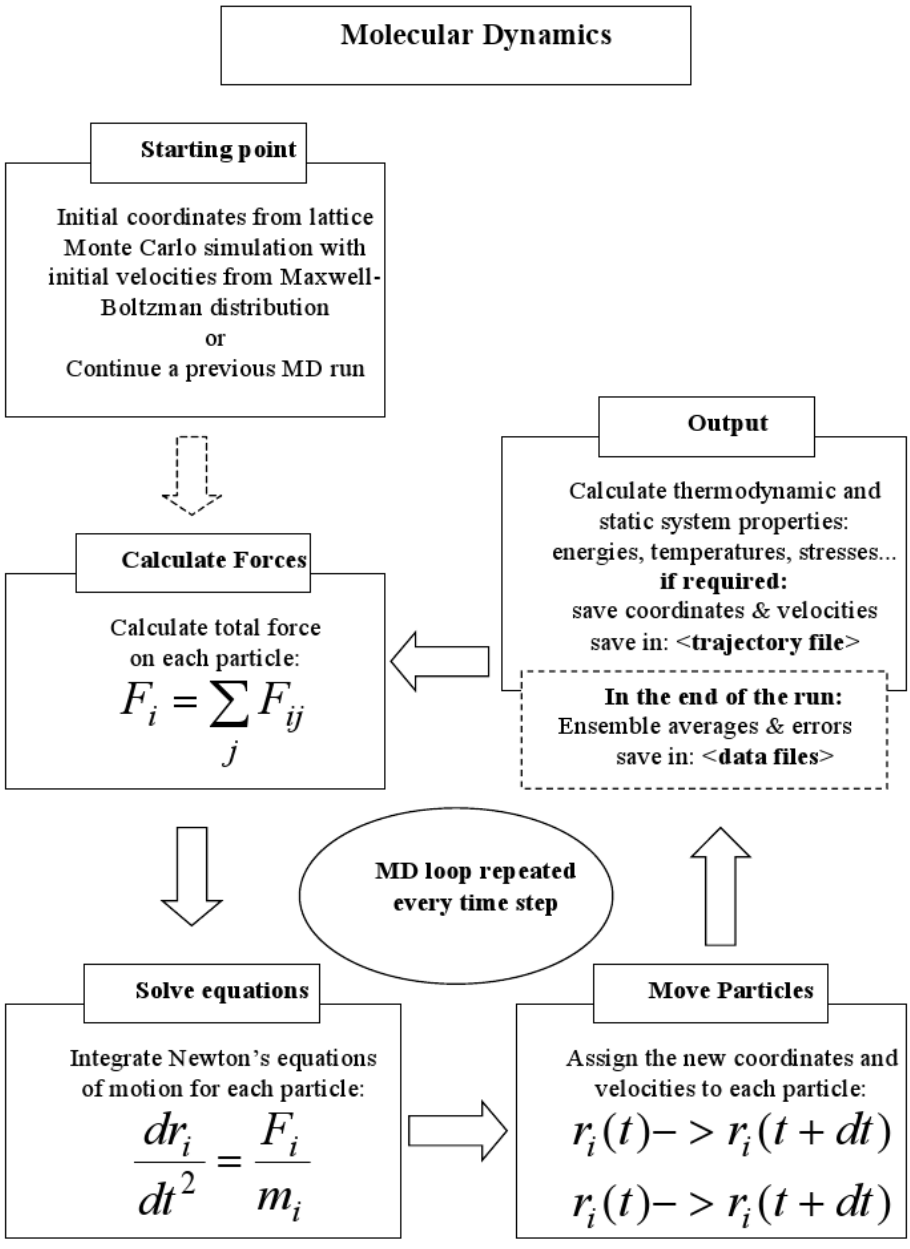


Fig. 1. Simplified algorithm of Molecular Dynamics.

For atomic system the accuracy is of the same order as the commonly used Verlet algorithm.

The basic idea of the algorithm is to write two third-order Taylor expansions for the positions $r(t)$, one forward and one backward in time. Denoting by v the velocities, a the accelerations, and b the third derivatives of r with respect to t ,one has:

$$r(t + \Delta t) = r(t) + v(t)\Delta t + 1/2a(t)\Delta t^2 + (1/6)b(t)\,\Delta t^3 + O(\Delta t^4) \quad (3.1)$$

$$r(t - \Delta t) = r(t) - v(t)\Delta t - 1/2a(t)\Delta t^2 - (1/6)b(t)\,\Delta t^3 + O(\Delta t^4) \quad (3.2)$$

Summation of the two expressions gives:

$$r(t + \Delta t) = 2r(t) - r(t - \Delta t) + a(t)\Delta t^2 + O(\Delta t^4) \quad (3.3)$$

This is the basic form of the Verlet algorithm. Since we are integrating Newton's equations, $a(t)$ is just the force divided by the mass, and the force is in turn a function of the positions $r(t)$:

$$a(t) = -(1/m)\nabla v(r(t)) \quad (3.4)$$

The velocities are not needed to compute the trajectories, but they are useful for estimating the kinetic energy (total energy) [5]. They may be obtained from the formula:

$$v(t) = \frac{r(t + \Delta t) - r(t - \Delta t)}{2\Delta t} \quad (3.5)$$

This algorithm is at the same time simple to implement, accurate and stable, which explains its large popularity among molecular dynamics simulators [3].

When studying the structural properties of highly entropic materials, the usual criteria for the choice of the time step, the stability of the energy and the pressure, are not always appropriate as they may lead to a time step which is an order of magnitude too large.

4. Molecular Dynamics Simulation Results Of Plane Poiseuillea Nanoflows

The results of the simulations of plane Poiseuille flow of water through narrow channels [13, 14, 15] are presented and compared with the analytical solutions for micropolar fluid. In the simulations performed by use the Molecular Dynamic method, physical properties of the materials and their electrostatic interactions were

considered. Simulation of the flow was carried on for two different channel width : W=5 and W=10 diameters of water molecule.

The simulations of water plane Poiseuille flow were performed with the program based on MOLDY [4]. The velocity calculations were carried out using the bin method to the data collected from 200 000 time steps Δt=0.005 picosecond long, after the considered system has reached the equilibrium.

To simulate the water molecules the four-site molecular model of Jorgensen [6] was used. The molecular parameters of quartz and copper from [7] and [8] were used.

The MD cells snapshots of water flows in quartz channels is presented at Fig.2.

In the figure 3 the results of MD simulation (dots) and analytic solution (solid red lines). are shown. The velocity profiles for narrower channels are significantly more distant from parabolic shape and analytical solutions.

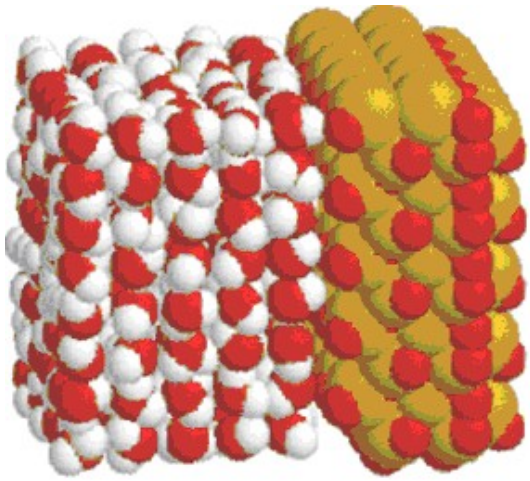


Fig. 2. The MD elementary cell for water/quartz channel used in simulation[13].

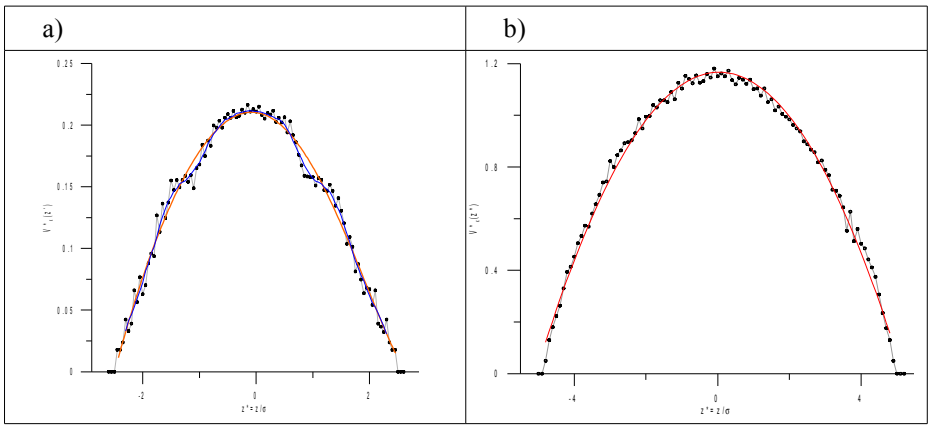


Fig. 3. Velocity distributions for Poiseuille flow of water in nanochannels, a)– narrow channel, b)– wide channel. MD simulation (dots) and analytic solution (solid red lines)[13].

The presented results for channel width effect are consistent with those of the other investigators [9, 10, 11, 12] who considered also flows of other liquids through channels. The similar size effect can be observed for gas flows too [16].

5. Conclusions:

The presented paper describes very briefly the main concepts of the Molecular Dynamics Simulation method, in application to flows of liquids. On the basis of the presented results it is possible to conclude, that this method is well suited to nanoflows and, in consequence, it has broad area of applicability – nano-physics, nanotechnology, engineering and others.

References:

1. J. Grotendorst, D. Marx, A. Muramatsu: *Quantum Simulations of Complex Many-Body Systems: From Theory to Algorithms, John von Neumann Institute for Computing,* Jülich, NIC Series, Vol. 10, ISBN 3-00-009057-6, pp. 211-254, 2002.
2. A. Z. Szeri, V. Radel: *Flow modeling of thin films from macroscale to nanoscale,* NATO Advanced Study Institute: Fundamentals of and Bridging the Gap between Macro- and Micro/nanoscale Tribology, Keszthely, Hungary, 2000.
3. F. Ercolessi: *A molecular dynamics primer,* Spring College in Computational Physics, ICTP, Trieste, Italy, 1997.
4. K. Refson: *Moldy User's Manual,* Chapter II, `ftp://ftp.earth.ox.ac.uk/pub`
5. M. P. Allan, D. J. Tildesley: *Computer simulations of Liquids,* Oxford Universit Press, 1987.
6. W. L. Jorgensen: J. Chem. Phys. **77**, 4156-63, 1982.
7. J. Adler: `https//reu. Magnet.fsu.edu/program/2003/paper/adler`
8. B. W. Van Beest, G. Kramer and R. A. Van Santen: Physical Rev. Letters **64,** 16, 955-1975, 1990.
9. J. Delhommelle, D. J. Evans: Molecular Physics, **100**, 17, 2857-65, 2002.
10. D. C. Rapaport: Europhysics Letters, **26**, 6, 401-406, 1994.
11. K. P. Travis, D. J. Evans: Phys. Rev. E., **55**, 2, 1566-1572, 1997.
12. A. Kucaba-Pietal, Z. Walenta, Z.Peradzynski: TASK Quaterly, **5**,2,179-189,2001.

13. A. Kucaba-Piętal.: *Modelowanie mikroprzepływów na gruncie teorii płynów mikropolarnych,* Oficyna Wydawnicza Politechniki Rzeszowskiej, 2004.

14. Z. Walenta, A. Kucaba-Piętal, Z. Peradzyński: *Water flows in copper and quartz nanochannels,* Mechanics XXI Century, Springer Netherlands, 2005.

15. A. Kucaba-Piętal, Z. Peradzyński, Z. Walenta: *Wall and size effect on water flows in nanochannels,* ICTAM Proceedings 2005.

16. Cieplak M., Koplik J., Banavar J. R.: Physica A, 274, 281-293, 1999.

Proceedings of the International Multiconference on
Computer Science and Information Technology pp. 23 – 477
ISSN 1896-7094
© 2007 PIPS

Linking of direct and iterative methods in Markovian models solving

Beata Bylina and Jarosław Bylina

Department of Computer Science
Institute of Mathematics
Marie Curie-Sklodowska University
Pl. M. Curie-Skłodowskiej 1, 20-031 Lublin, Poland
beatas@hektor.umcs.lublin.pl
jmbylina@hektor.umcs.lublin.pl

Abstract. An article identifies and assesses an effectiveness of two different methods applied to solve linear equations systems which result while modeling of computer networks and systems with Markov chains. The paper considers both the hybrid of direct methods as well as classic one of iterative methods. Two varieties of Gauss elimination will be considered as an example of direct methods: the LU factorization method and the WZ factorization method. Gauss-Seidel iterative method will be discussed. That issue points in preconditioning and matrix division into blocks where blocks will be solved applying direct methods. The paper presents an impact of liked methods on both time and accuracy of vector probability determining regarding particular networks and computer systems occurring.

1 Introduction

Computer networks are being continuously extended. Users demand bands, they want to send in real time both picture and sound. That is the reason for network work modeling to become important problem regarding both its work predicting and breakdown preventing. One of the computer networks modeling methods are Markovian models. Markovian models need two information; in what states can network exist and what are probabilities (or rates of transitions among states). Basing on mentioned above the equations system is being built to describe transition rates among particular states to determine the probability of particular networks states occurrence. After solving of equations systems linking states probabilities it is possible to determine the parameters of network work (ex.: capacity, packets loosing probability, delay, packets dropping rate).

In steady state (independent of time) while modeling with Markov chains— we obtain by linear equation system like follows;

$$\mathbf{Q}^T\mathbf{x} = \mathbf{0}, \qquad \mathbf{x} \geq \mathbf{0}, \qquad \mathbf{x}^T\mathbf{e} = 1, \tag{1}$$

where $\mathbf{Q}$ means transition rate matrix and $\mathbf{x}$ is a vector of states probabilities and vector $\mathbf{e} = (1, 1, ..., 1)^T$. Matrix $\mathbf{Q}$ is the square matrix $n \times n$, usually a big one, of

rank $n-1$, sparse, with dominant diagonal. It is also peculiar matrix demanding adequate methods to solve it. Markovian models solving demands overcoming both numerical and algorithmic problems. Equation system (1) solving generally demands applying of iterative methods, projection methods or decomposition methods but occasionally (for the need of accurate solution) direct methods are used. Information concerning direct methods applying in Markov chains can be found in the following elaborations: [7–10]. Iterative methods are described for example in [12, 13] and projection method are concerned in articles [2, 14]; and also decomposition methods were described in [11, 15]. The rich material concerning above mentioned methods can be found in [16].

Both direct methods and iterative ones have advantages and disadvantages. So one of the way to take advantage of both methods is linking them to solve big sparse problems. This kind of approach is described in Duff's paper [4] (among others).

In that article we will point at two ways of iterative and direct methods linking: preconditioning and block Gauss-Seidel method. Novelty of that elaboration concerns applying the WZ factorization method instead of LU factorization method —regarding both incomplete factorization in preconditioning techniques as well as during solving of block Gauss-Seidel method.

Second section of article includes discussion of direct and iterative methods features applied in solving of linear equation system modeling computer network work using Markov chains. Third section presents preconditioning idea and incomplete factorizations applying as preconditioning techniques. Fourth section describes block Gauss-Seidel method. Fifth section presents the results of numerical experiment conducted for matrices describing any computer network abstractive model. In last section there are conclusions regarding conducted experiments.

2 The scope and the constrains in the methods of equations system solving

In literature we can find four approaches (mentioned above) concerning linear equations system solving (1)—in article we describe two methods: direct methods and iterative methods.

2.1 Direct methods

Direct methods (also known as accurate ones) are the ones which would lead to accurate equation system solving after complete steps number if all calculations were done without rounding. All direct methods are the modifications of Gauss elimination method which enables determining of LU factorization, where matrix $\mathbf{L}$ is lower triangular and $\mathbf{U}$ is upper triangular matrix.

The features of direct methods:

– they give solution in finite steps number by applying the decomposition;

– applying direct methods in equations solving – elimination of one matrix's non-zero element in reduction phase is followed often by creating some non-zero elements where earlier there were zeros. The fill-in effect makes not only matrix organization saving more complicated but also the size of fill-in can be so big that accessible memory ends fast;
– direct methods enable determining of demanded operations number before calculations finishing (for Gauss elimination: $n^3/3 + n^2/2 - 5n/6$ multiplications and $n^2/2 + n/2$ divisions);
– for some problems solving with the direct methods determine more precise answer in shorter time than iterative methods.

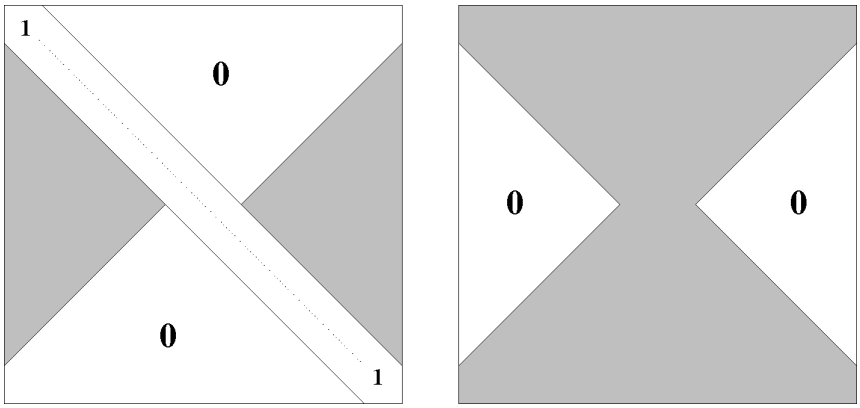


Fig. 1. The form of the output matrices in the WZ factorization (left: **W**; right: **Z**)

WZ factorization is another version of Gauss elimination (see Figure 1) which was designed and described in article [6] especially for SIMD computers (*Single Instruction Stream—Multiple Data Stream*) in 1979. For the reason that effective working computers were not constructed (of that architecture), factorization WZ was adapted to existing computer architectures [3, 5]. Paper [17] presents WZ factorization which can be faster on computers with parallel architecture from LU factorization for matrices with dominant diagonal—what is the feature of transition rate matrix **Q**.

Regarding above mentioned WZ factorization features, this direct method was chosen for equation system to be solved (1). That method can be applied successfully at determining of small quantities where there is a need to implement calculations with greater accuracy.

2.2 Iterative methods

Iterative methods begin from some approximation and create indirect results sequence which tends—as it is expected—to solve the problem. It is difficult to

know in advance how many iterations is demanded to gain assumed accuracy so it is impossible to determine the number of operations before calculations finishing. Gauss-Seidel method is an example of iterative method and it is is described further.

Matrix $\mathbf{Q}^T = \mathbf{L} + \mathbf{D} + \mathbf{U}$, where $\mathbf{L}$ is strictly lower triangular, $\mathbf{D}$ is diagonal matrix and $\mathbf{U}$ is strictly upper triangular matrix. Assuming that approximation $\mathbf{x}^{(i)} = [x_1^{(i)}, \ldots, x_n^{(i)}]^T$ is given, in Gauss-Seidel method the next approximation $\mathbf{x}^{(i+1)} = [x_1^{(i+1)}, \ldots, x_n^{(i+1)}]^T$ is determined to comply with equation:

$$a_{k1}x_1^{(i+1)} + \ldots + a_{kk}x_k^{(i+1)} + a_{k,k+1}x_{k+1}^{(i)} + \ldots + a_{kn}^{(i)} = b_k \tag{2}$$

Transforming the formula we have got the following result:

$$(\mathbf{L} + \mathbf{D})\mathbf{x}^{(i+1)} + \mathbf{U}\mathbf{x}^{(i)} = 0 \tag{3}$$

what gives the following iterative formula known as Gauss-Seidel method:

$$\mathbf{x}^{(i+1)} = -(\mathbf{L} + \mathbf{D})^{-1}\mathbf{U}\mathbf{x}^{(i)} \tag{4}$$

Equation (4) shows that in every iteration there are applied newly computed vector's values.

Advantages of iterative methods:

- easy in implementation because there is no matrices modification in calculation process;
- they enable memory saving;
- calculation can be done very precisely (the more precisely they are done the longer time of algorithm making is).

Problems concerning iterative methods:

- problem with convergence for ill-conditioned matricex;
- sometimes great number of operations to get convergence.

3 Preconditioning

Iterative methods convergence rate depends upon matrix $\mathbf{Q}$ attributes. Matrix $\mathbf{Q}$ is an ill-conditioned matrix—what can affect iterative methods applied to that matrix—they are slowly convergent or sometimes even divergent. One of the ways to prevent above mentioned is transforming of equation system into an equivalent system—having the same solution but better attributes to solve it with iterative methods. So preconditioning is an example of above mentioned applying.

System (1) is transformed into the following formula system:

$$\mathbf{M}^{-1}\mathbf{Q}^T\mathbf{x} = \mathbf{0}, \qquad \mathbf{x} \geq \mathbf{0}, \qquad \mathbf{x}^T\mathbf{e} = 1 \tag{5}$$

where matrix $\mathbf{M}$ approximates matrix $\mathbf{Q}^T$ in some way. So, instead of system (1) solving, we find solution of system (5). Applying any preconditioning is tightly connected with high calculations cost. That cost concerns the construction and application of preconditioning. (what means finding matrix $\mathbf{M}$ approximating matrix $\mathbf{Q}^T$ and inverse matrix $\mathbf{M}^{-1}$). The cost of preconditioning construction can be amortized by iterations quantity or by using the same preconditioning for different equation systems. Different techniques are applied to determine matrix $\mathbf{M}$. It is very difficult to find good preconditioning to solve sparse linear equation system. It happens very often that theoretical results can not be confirmed by practice. There are not enough tools and means to prove above mentioned (besides numerical experiments). Preconditioners are usually built basing on original matrix $\mathbf{Q}^T$ coefficients. In article [1] the authors consider preconditioning Krylov subspace methods for solving large singular linear systems arising from Markovian modeling. In next section preconditioning construction will be described basing upon incomplete factorizations applied for classical iterative methods.

3.1 Incomplete LU factorization

Preconditioning example basing on incomplete LU factorization—marked with ILU (incomplete LU) consisting in lower triangular matrix $\mathbf{L}$ calculation and also upper triangular matrix $\mathbf{U}$ which product gives approximating matrix $\mathbf{M}$. ILU has different variants but the simplest form of incomplete LU factorization is ILU(0) in which calculations are like in full LU factorization (taht is, Gauss elimination) but there are remembered only those elements l_{ij} and u_{ij} of matrices $\mathbf{L}$ and $\mathbf{U}$ for which adequate q_{ij} element of the given matrix $\mathbf{Q}^T$ was non-zero. That is why output matrices have the number of non-zero elements accurately the same what $\mathbf{Q}^T$ matrix has. In that way the most important problem associating sparse matrix factorization—fill-in (non-zero elements in places where in original matrix were only zeros—what is followed by matrices exchange from sparse for the dense one and widening the room to save it)—is eliminated.

Matrix $\mathbf{Q}^T$ can be written as $\mathbf{Q}^T = \mathbf{LU}+\mathbf{R}_{LU}$ where $\mathbf{R}_{LU}$ matrix is expected to be a small one (in a sense). Let $\mathbf{M} = \mathbf{LU}$, that is $\mathbf{M}^{-1} = \mathbf{U}^{-1}\mathbf{L}^{-1}$ and then equation (5) looks like

$$\mathbf{U}^{-1}\mathbf{L}^{-1}\mathbf{Q}^T\mathbf{x} = \mathbf{0}, \qquad \mathbf{x} \geq \mathbf{0}, \qquad \mathbf{x}^T\mathbf{e} = 1. \tag{6}$$

Let $\mathbf{S}_{LU} = \mathbf{U}^{-1}\mathbf{L}^{-1}\mathbf{Q}^T$. Then, equation (6) has the following form:

$$\mathbf{S}_{LU}\mathbf{x} = \mathbf{0}, \qquad \mathbf{x} \geq \mathbf{0}, \qquad \mathbf{x}^T\mathbf{e} = 1.$$

3.2 WZ incomplete factorization

Just like incomplete LU factorization above, we can define IWZ (incomplete WZ factorization)—more precisely: IWZ(0).

In IWZ(0) there are remembered only those elements w_{ij} and z_{ij} of matrices $\mathbf{W}$ and $\mathbf{Z}$ for which adequate q_{ij} element of the given matrix $\mathbf{Q}^T$ was non-zero.

Hence, the output matrices have the number of non-zero elements accurately the same what $\mathbf{Q}^T$ matrix has and the fill-in – is eliminated.

Matrix $\mathbf{Q}^T$ can be written as $\mathbf{Q}^T = \mathbf{WZ} + \mathbf{R}_{WZ}$ where $\mathbf{R}_{WZ}$ matrix is expected to be a small one (in a sense). Let $\mathbf{M} = \mathbf{WZ}$, that is $\mathbf{M}^{-1} = \mathbf{Z}^{-1}\mathbf{W}^{-1}$ and then equation (5) looks like

$$\mathbf{Z}^{-1}\mathbf{W}^{-1}\mathbf{Q}^T\mathbf{x} = \mathbf{0}, \qquad \mathbf{x} \geq \mathbf{0}, \qquad \mathbf{x}^T\mathbf{e} = 1. \tag{7}$$

Let $\mathbf{S}_{WZ} = \mathbf{Z}^{-1}\mathbf{W}^{-1}\mathbf{Q}^T$. Then, equation (7) has the following form:

$$\mathbf{S}_{WZ}\mathbf{x} = \mathbf{0}, \qquad \mathbf{x} \geq \mathbf{0}, \qquad \mathbf{x}^T\mathbf{e} = 1.$$

4 Block Gauss-Seidel method

It is possible to divide the transition rate matrix into blocks and develop iterative methods basing on that division. Generally iterative block methods demand more calculations per iteration what is recompensed by faster convergence rate.

Homogeneous equations system is divided into K^2 blocks in the following way:

$$\mathbf{Q}^T\mathbf{x} = \begin{bmatrix} \mathbf{Q}_{11} & \mathbf{Q}_{12} & \dots & \mathbf{Q}_{1K} \\ \mathbf{Q}_{21} & \mathbf{Q}_{22} & \dots & \mathbf{Q}_{2K} \\ \dots & \dots & \dots & \dots \\ \mathbf{Q}_{K1} & \mathbf{Q}_{K2} & \dots & \mathbf{Q}_{KK} \end{bmatrix} [\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_K]^T = \mathbf{0}$$

We introduce block splitting:

$$\mathbf{Q}^T = \mathbf{D}_K - (\mathbf{L}_K + \mathbf{U}_K),$$

where $\mathbf{D}_K$ is block-diagonal matrix, $\mathbf{L}_K$ - is strictly block lower triangular matrix, $\mathbf{U}_K$ is strictly block upper triangular matrix with form:

$$\mathbf{D}_K = \begin{bmatrix} \mathbf{D}_{11} & 0 & \dots & 0 \\ 0 & \mathbf{D}_{22} & \dots & 0 \\ \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & \mathbf{D}_{KK} \end{bmatrix},$$

$$\mathbf{L}_K = \begin{bmatrix} 0 & 0 & \dots & 0 \\ \mathbf{L}_{21} & 0 & \dots & 0 \\ \dots & \dots & \dots & \dots \\ \mathbf{L}_{K1} & \mathbf{L}_{K2} & \dots & 0 \end{bmatrix},$$

$$\mathbf{U}_K = \begin{bmatrix} 0 & \mathbf{U}_{12} & \dots & \mathbf{U}_{1K} \\ 0 & 0 & \dots & \mathbf{U}_{2K} \\ \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & 0 \end{bmatrix}.$$

Block Gauss-Seidel method is given by:

$$(\mathbf{D}_K - \mathbf{L}_K)x^{(i+1)} = \mathbf{U}_K x^{(i)}.$$

Describing above mentioned equation in scalar-like form we get:

$$\mathbf{D}_{jj}\mathbf{x}_j^{(i+1)} = (\sum_{l=1}^{i-1}\mathbf{L}_{jl}\mathbf{x}_l^{(i+1)} + \sum_{l=j+1}^{K}\mathbf{U}_{jl}\mathbf{x}_l^{(i)}) \quad \text{for } j = 1, 2, \ldots, K,$$

where $\mathbf{x}_j^{(i)}$ is the jth (n/K)-element sub-vector of the vector $\mathbf{x}^{(i)}$.

In the result of above in every step we must solve K equation systems of n/K size each in the following form:

$$\mathbf{D}_{jj}\mathbf{x}_i^{(i+1)} = \mathbf{z}_j^{(i+1)} \quad \text{for } j = 1, 2, \ldots, K, \tag{8}$$

where

$$\mathbf{z}_j^{(i+1)} = (\sum_{l=1}^{i-1}\mathbf{L}_{jl}\mathbf{x}_l^{(i+1)} + \sum_{l=j+1}^{K}\mathbf{U}_{jl}\mathbf{x}_l^{(i)}) \quad \text{for } j = 1, 2, \ldots, K.$$

We can apply different direct and iterative methods to solve equation (8). For every block we make factorization only once before iteration, in loop we change only right equation sides. If $\mathbf{D}_{ii}$ matrix has special structure, for example is a diagonal, upper triangular, lower triangular or tri-diagonal matrix—then LU factorization is very simple to get and iterative-block method becomes very attractive. If $\mathbf{D}_{ii}$ matrices do not have any of mentioned structures, then it can appear that block solving demands iterative methods. Then, we have inside iterative method with cardinal iterative method. It demands selecting of proper method for inside method to be convergent and initial vector to be chosen for the given method. There is small number of iterations demanded to obtain convergence for small number of blocks (sub-matrices are big).

5 Numerical experiment

Algorithms implementation was done in C language. Data structures to save matrices $\mathbf{Q}^T$, $\mathbf{W}$, $\mathbf{Z}$, $\mathbf{L}$, $\mathbf{U}$ were full two-dimensional arrays situated in RAM. Numerical experiment was conducted on PC computer with 1GB RAM, Pentium IV 2.80 GHz processor. Algorithms were tested in Linux environment and compilator `gcc` with `-O3` option was used in compilation. Tests were conducted for sparse matrices generated by authors. Generated matrices described abstractive queuing models and they had all the attributes of transition rate matrices $\mathbf{Q}^T$. Table 1 shows information about test matrices for which IWZ and ILU incomplete factorizations were applied —those matrices were not symmetric at all. Where n means columns number and nz means non-zeros number.

Tested matrices differed with average non-zero elements number in column. Two matrices had more than 10 elements in column (group I) and the other two had about four elements in column (group II). It essentially affects calculations accuracy and iterative methods convergence.

Table 1. Test matrices attributes

group	ID	n	nz	average non-zeros per column (nz/n)
I	1	1000	12678	12.68
I	2	4000	42041	10.51
II	3	1500	5873	3.92
II	4	4000	16946	4.24

Probabilities vector was determined for every tested matrix applying usual Gauss-Seidel iterative method (GS) and block Gauss-Seidel method where blocks were solved using LU factorization (GSLU) and WZ factorization (GSWZ) where K^2 is the number of blocks, the transition matrices were divided into, and number of iterations demanded for obtaining an accuracy $e-*$. Table 2 shows time needed for algorithms (GS, GSWZ, GSLU) as well as and impact of above algorithms on result accuracy. The symbol i$(e-*)$ means the number of iteration steps needed to get the accuracy $e-*$.

Table 2. Comparing of algorithms calculation time (GS, GSWZ, GSLU) and an impact of algorithms on result accuracy

ID(n)	time and accuracy	GS	GSWZ		GSLU	
			$K=2$	$K=20$	$K=2$	$K=20$
1(1000)	time:	6.59	0.29	11.57	0.3	0.18
	i$(e-05)$	4	6	4	6	6
	i$(e-10)$	8	12	8	12	13
	i$(e-15)$	12	18	12	18	19
2(4000)	time:	376.95	4.92	709.93	5.87	2.13
	i$(e-05)$	4	6	4	6	6
	i$(e-10)$	8	12	8	12	13
	i$(e-15)$	12	19	12	19	20
3(1500)	time:	21.68	0.58	40.13	0.63	0.33
	i$(e-05)$	7	11	7	11	11
	i$(e-10)$	17	>20	17	>20	>20
	i$(e-15)$	>20	>20	>20	>20	>20
4(4000)	time:	347.75	4.71	647.8	5.6	1.86
	i$(e-05)$	6	10	6	10	10
	i$(e-10)$	15	>20	15	>20	>20
	i$(e-15)$	>20	>20	>20	>20	>20

Applying block GS (in variants GSLU and GSWZ) comparing to classic GS gave the following observation:

- For all the matrices—dividing into small number of blocks influences faster method's convergence. But dividing into great number of blocks makes accuracy to be obtained at the same number of steps like for usual GS.
- Time of algorithm calculating depends only upon blocks number independently on matrices groups. The least blocks number is the longer time of calculations—what is normal for great matrices to be solved with accurate methods and cost depends on size. Dividing into great number of blocks shortens calculations time. The more blocks are the solving time becomes similar to usual Gauss-Seidel method time solving.
- Calculations accuracy does not depend on factorization: WZ or LU. But factorization influences time of calculation. Dividing into great number of blocks, we get 1.5 time faster algorithm GSWZ than GSLU.

Incomplete factorizations IWZ and ILU were applied for matrix $\mathbf{Q}^T$ as preconditioning. Iterative Gauss-Seidel method was applied for obtained equation system (6) and (7) or for block Gauss-Seidel. Preconditioning can be used for all iterative, projective and other methods.

Table 3. Comparing of time of algorithms (GS, IWZ+GS, ILU+GS, GSWZ, GSLU) and its accuracy

ID(n)	algorithm	time	$i = 5$	$i = 10$	$i = 20$
1(1000)	GS	0,18	e-04	e-08	e-16
	IWZ+GS	0,17	e-08	e-12	e-18
	ILU+GS	0,17	e-06	e-11	e-16
	GSWZ(2)	6,59	e-06	e-13	e-15
	GSLU(2)	11,57	e-06	e-13	e-15
2(4000)	GS	2,14	e-04	e-08	e-15
	IWZ+GS	2,54	e-09	e-14	e-19
	ILU+GS	2,33	e-07	e-12	e-17
	GSWZ(2)	376,95	e-06	e-12	e-15
	GSLU(2)	709,93	e-06	e-12	e-15
3(1500)	GS	0,33	e-03	e-04	e-08
	IWZ+GS	0,38	e-05	e-09	e-16
	ILU+GS	0,34	e-04	e-08	e-15
	GSWZ(2)	21,68	e-04	e-07	e-12
	GSLU(2)	40,13	e-04	e-07	e-12
4(4000)	GS	1,86	e-03	e-05	e-08
	IWZ+GS	2,38	e-05	e-09	e-17
	ILU+GS	2,15	e-04	e-08	e-16
	GSWZ(2)	347,75	e-04	e-07	e-13
	GSLU(2)	647,18	e-04	e-07	e-13

Table 3 shows the comparison of the methods: traditional Gauss-Seidel algorithm (denoted: GS), preconditioned GS—both with IWZ (denoted: IWZ+GS)

and with ILU (denoted: ILU+GS), and block GS, where blocks were solved with WZ (denoted: GSWZ(K)) and with LU (denoted: GSLU(K)). Here, the number of blocks is K^2. There was chosen $K = 2$, because of the best accuracy. The time was compared, as well as convergency after 5, 10 and 20 iterations.

The results observed:

- The convergence of matrices from group II (more sparse matrices) is weak so they need more accuracy followed by iteration number reduction or applying preconditioning or blocking.
- Applying both preconditioning and blocking improves convergence of GS iterative methods.
- Preconditioning accelerates iterative methods convergence—especially using incomplete IWZ factorization for usual GS regarding both calculations time and accuracy.

6 Conclusions

In above article the authors compared two techniques which can affect calculations accuracy: preconditioning and blocking in special linear equation systems solving. Matrix coefficients of those systems describe transition rates from one state to another state described by Markov processes. Numerical experiment results show that it is better to apply usual Gauss-Seidel preconditioning for these systems than blocking. It also should be noted that if we have IWZ and ILU to chose it is better to chose IWZ—having shorter calculations time and more accuracy. Moreover it was observed that convergence of matrices from group II is weak—so they need special techniques applying to improve accuracy what is followed by reducing of iterations number and using preconditioning or blocking: teats result in applying preconditioning -as the bets method. Matrix should be divided into less blocks number to improve accuracy what affects calculation time (longer one). In next articles we plan to join block methods and preconditioning as well as conducting of calculations time.

Acknowledgements

This work was partially supported by Marie Curie-Sklodowska University in Lublin within the project *Niekompletny rozkład WZ jako uwarunkowanie wstepne (ang. preconditioning) stosowane do znajdowania wektora prawdopodobieństw stanów sieci całkowicie optycznych modelowanych łańcuchami Markowa*.

This work was also partially supported within the project *Metody i modele dla kontroli zatłoczenia i oceny efektywności mechanizmów jakości usług w Internecie nastepnej generacji* (N517 025 31/2997).

References

1. Benzi M., Ucar B.: *Block Triangular preconditioners for M-matrices and Markov chains,* [To appear in Electronic Transactions on Numerical Analysis, 26 (2007).]

2. J. Bylina: *Distributed solving of Markov chains for computer network models,* Annales UMCS Informatica 1 (2003), pp. 15–20.
3. M. M. Chawla, R. R. Khazal: *A new WZ factorization for parallel solution of tridiagonal systems,* Int. J. Comput. Math. 80 (2003), no. 1, 123–131.
4. I. S. Duff: *Combining direct and iterative methods for the solution of large systems in different application areas,* Technical Report RAL-TR-2004-033.
5. D. J. Evans, M. Barulli: *BSP linear solver for dense matrices,* Parallel Computing 24 (1998), pp. 777–795.
6. D. J. Evans, M. Hatzopoulos: *The parallel solution of linear system,* Int. J. Comp. Math. 7 (1979), pp. 227–238.
7. R. E. Funderlic, C. D. Meyer: *Sensitivity of the stationary distrbution vector for an ergodic Markov chain,* Linear Algebra Appl., 76, 1986, pp. 1–17.
8. R. E. Funderlic, R. J. Plemmons: *Updating LU factorizations for computing stationary distributions,* SIAM J. Alg. Disc. Meth., 7, 1986, pp. 30–42.
9. G. H. Golub, C. D. Meyer: *Using the QR factorization and group inversion to compute, differentiate and estimate the sensitivity of stationary distributions for Markov chains,* SIAM J. Alg. Disc. Meth., 7, 1986, pp. 273–281.
10. W. J. Harrod, R. J. Plemmons: *Comparisons of some direct methods for computing stationary distributions of Markov chains* SIAM J. Sci. Comput., 5, 1984, pp. 453–469.
11. M. Haviv: *Aggregation/disagregation methods for computing the stationary distribution of a Markov chain,* SIAM J. Numer. Anal., 24, 1987, pp. 952–966.
12. A. Jennings, W. J. Stewart: *Simultaneous iteration for partial eigensolution of real matrices,* J. IMA, 15, 1975, pp. 351–361.
13. A. Jennings, W. J. Stewart: *A simultaneous iteration algorithm for real matrices,* ACM Trans. of Math. Software, 7, 1981, pp. 184–198.
14. Y. Saad, M. H. Schultz: *GMRES: A generalized minimal residual algorithm for solving non-symmetric linear systems,* SIAM Journal of Scientific and Statistical Computing, 7, 1986, pp. 856–869.
15. P. J. Schweitzer, K. W. Kindle: *An iterative aggregation-disaggregation algorithm for solving linear systems,* Applied Math. and Comp., 1986, 18, pp. 313–353.
16. W. Stewart: *Introduction to the Numerical Solution of Markov Chains,* Princeton University Press, Chichester, West Sussex 1994.
17. P. Yalamov, D. J. Evans: *The WZ matrix factorization method,* Parallel Computing 21 (1995), pp. 1111–1120.

Proceedings of the International Multiconference on
Computer Science and Information Technology, pp. 479 – 481
ISSN 1896-7094
© 2007 PIPS

Airfoil shape optimization by coupling computational fluid dynamics with evolutionary multiobjective optimization

Daniela Zaharie[1,2], Silviu Panica[1,2], Marius Stoia-Djeska[3],
Mircea Dragan[1,2], Dana Petcu[1,2]

[1] Department of Computer Science, West University of Timisoara, Romania
{silviu,dzaharie,dragan,petcu}@info.uvt.ro
[2] e-Austria Institute, Timisoara, Romania
[3] Politehnica University of Bucharest,
marius. stoia@rosa.ro

Abstract. The problem of coupling in an efficient way computational fluid dynamics and evolutionary multiobjective optimization codes in order to solve problems of optimal design is discussed. Both the problem of providing an easy to use framework and that of the computational cost are addressed. Moreover, a user interface was designed to allow the execution of different instances, with respect to the parameters of the evolutionary algorithm, of the combined code on several machines from a Grid infrastructure.

1 Introduction

The optimal design problems in aerodynamics have two main characteristics: the existence of multiple criteria to be optimized and the high computational cost of objective functions evaluation. For instance, in the case of airfoil shape optimization the aim could be to identify profiles that minimize the drag of the airplane but in the same time ensure a sufficient value for the lift and minimizes the difference between the actual pressure on the wing surface and a target pressure. Dealing with the tradeoff between these criteria means to solve a multiobjective optimization problem and to obtain a set of reciprocally nondominated solutions. For multiobjective optimization problems, the traditional gradient-based methods are not appropriate, thus derivative free stochastic methods, like evolutionary algorithms, became popular [3-5]. When applying evolutionary algorithms for optimal shape design there are at least two difficulties that arise: (i) the high cost of evaluating the population elements; (ii) the stochastic character of the evolutionary algorithms and their sensitivity to the parameters choice. A useful tool for designers would allow the easy choice of an evolutionary multiobjective optimization algorithm (EMOA) without requiring advanced knowledge in the field of evolutionary algorithms. On the other hand, it would offer support for parallel and/or distributed computing in order to make the computational fluid dynamics (CFD) computations efficient.

2 The problem

We will illustrate the main ideas of our approach for the optimization problem of airfoil shape. In this case, the aim is to find wing profiles, which optimize the aerodynamic forces, i.e. lift and drag. The profile is described by two Bezier curves, one for the upper surface and one for the lower surface, as in [3-5]. To define these curves a few control points are used and the coordinates of these control points are to be adjusted by the optimization procedure. For each profile, i.e. set of control points, the flow around the wing is simulated and the corresponding aerodynamic forces are estimated by a CFD code.

3 The CFD component

The CFD solver that we use in the context of our problem is based on a cell-centered, second order Godunov type finite volume discretization of the partial differential equation describing the flow. An implicit marching in time procedure allows us to obtain the final steady-state solution starting from an arbitrary initial state. To avoid the solution of large linear algebraic systems at each time step we use a dual-time approach combined with a fourth-order Runge-Kutta scheme. The boundary conditions are fixed in time. The convergence criterion is the reduction of the residuals under a predefined threshold. The CFD solver is implemented and available as executable code.

4 The EMOA component

The evolutionary optimization is based on a population of profile descriptions. Each element of the population is thus a set of coordinates of control points describing the upper and the lower curves defining the profile. Since these coordinates are real values a real coded EMOA, e.g. NSGA-II [2], is appropriate. Starting from a random initial population of profiles, new profiles are generated by specific mutation and crossover operations and those which are acceptable with respect to all optimization criteria are selected to survive in the next generation. The main problem with such an evolutionary approach is the high cost of the iterative process of evolution since at each generation each element of the population has to be evaluated with respect to all criteria. This means to generate/update the computational mesh corresponding to the current profile and to apply the CFD code. It is obvious that in such a case the most computational expensive operation is that of profiles evaluation which implies that an adequate parallelization strategy for EMOA would be based on the master-slave model [1]. The EMOA component is implemented in Java and uses MPIJava as communication library for parallel processing. Moreover it allows specifying the use of external codes in the evaluation of population elements.

5 A coupling framework

Having the CFD and EMOA codes independently implemented, the problem that arises consists in coupling them in order to obtain a functional evolutionary design of optimal shapes. The variant we propose is based on a small module ensuring the interface between the EMOA code and the CFD code. The main role of this module is to prepare the input data for the CFD code starting from the population element to be evaluated. On the other hand, we also designed a user interface for the coupling framework allowing the execution of different instances, with respect to the parameters of the EMOA, of the combined code on several machines from a Grid infrastructure. In this way, the user can deal with the stochastic character and parameters sensitivity of EMOAs and to improve the effectiveness of the approach.

Acknowledgment. This work is supported by the projects RO-CEEX 95/ 03.10.2005 (GridMOSI - UVT) and RO-CEEX 65/31.07.2006 (SIAPOM - IeAT).

References

1. E. Alba, M. Tomassini, *Parallelism and evolutionary algorithms,* IEEE Trans. Evol. Comput. 5 (6), pp. 443-462, 2002.
2. Deb K. et al. *A Fast Elitist Non-dominated Sorting Genetic Algorithm for Multi-objective Optimization: NSGA-II,* in M. Schonauer et al (Eds.), Proceedings of PPSN 2000, LNCS 1917, pp. 849-858, 2000.
3. R. A. E. Makinen, P. Neittaanmaki, J. Periaux and J. Toivanen, *A Genetic Algorithm for Multiobjective Design Optimization in Aerodynamics and Electromagnetics,* In K. D. Papailiou et al., editor, Proceedings of the ECCOMAS 98 Conference, vol. 2, pp. 418-422, 1998.
4. N. Marco, J-A. Desideri, S. Lanteri, *Multi-objective optimization in CFD by genetic algorithms,* Rapport de Recherche INRIA, no. 3686, 1999.
5. H-K. Ng, D. Lim, Y-S. Ong, B-S. Lee, L. Freund, S. Parvez, B. Sendhoff, *A multi-cluster Grid enabled evolution framework for aerodynamic airfoil design optimization,* in L. Wang, K. Chen and Y. S. Ong (Eds.): ICNC 2005, LNCS 3611, pp. 1112-1121, 2005.

Proceedings of the International Multiconference on
Computer Science and Information Technology pp. 483–492
ISSN 1896-7094
© 2007 PIPS

Comparative Analysis of High Performance Solvers for 3D Elliptic Problems

Ivan Lirkov and Yavor Vutov

Institute for Parallel Processing, Bulgarian Academy of Sciences,
Acad. G. Bonchev, Bl. 25A, 1113 Sofia, Bulgaria,
ivan@parallel.bas.bg yavor@parallel.bas.bg
http://parallel.bas.bg/~ivan/ http://parallel.bas.bg/~yavor/

Abstract. The presented comparative analysis concerns two iterative solvers for 3D linear boundary value problems of elliptic type. After applying the Finite Difference Method (FDM) or the Finite Element Method (FEM) discretization a system of linear algebraic equations has to be solved, where the stiffness matrix is large, sparse and symmetric positive definite. It is well known that the preconditioned conjugate gradient method is the best tool for efficient solution of large-scale symmetric systems with sparse positive definite matrices. Here, the performance of two preconditioners is studied, namely the Modified Incomplete Cholesky factorization MIC(0) and the Circulant Block-Factorization (CBF) preconditioning. Portable parallel codes are developed based on Message Passing Interface (MPI) standards. The comparative analysis is mostly based on the execution times to run the parallel codes. The number of iterations for both preconditioners are also discussed. The performed numerical tests on parallel computer systems demonstrate the level of efficiency of the developed algorithms. The obtained parallel speed-up and efficiency well illustrate the scope of efficient applications.

1 Introduction

We are concerned with the numerical solution of linear boundary value problems of elliptic type. After discretization, such problems are reduced to find the solution of linear systems of the form $A\mathbf{u} = \mathbf{b}$. We consider here symmetric and positive definite problems. We assume also, that A is a large scale matrix. In practice, large problems of this class are often solved by iterative methods, such as the conjugate gradient (CG) method. At each step of these iterative methods only the product of A with a given vector $\mathbf{v}$ is needed. Such methods are therefore ideally suited to exploit the sparsity of the matrix A.

Typically, the rate of convergence of these methods depends on the condition number $\kappa(A)$ of the coefficient matrix A: the smaller $\kappa(A)$ is, the faster convergence. Unfortunately, for elliptic problems of second order, usually $\kappa(A) = \mathcal{O}(n^2)$, where n is the number of mesh points in each coordinate direction, and hence grows rapidly with n. To accelerate the iteration convergence a preconditioner M is combined with the CG algorithm. The theory of the Preconditioned

CG (PCG) method says that M is considered as a good preconditioner if it reduces significantly the condition number $\kappa(M^{-1}A)$, and at the same time, if the preconditioner allows efficient computation of the product $M^{-1}\mathbf{v}$ for a given vector $\mathbf{v}$. A third important aspect should be added to the above two, namely, the requirement for efficient implementation of the PCG algorithm on recent parallel computer systems, see e.g. [5].

2 The 3D Elliptic Problem

Let us consider the following 3D elliptic problem:

$$\begin{gathered} -\frac{\partial}{\partial x_1}\left(k_1\frac{\partial u}{\partial x_1}\right)-\frac{\partial}{\partial x_2}\left(k_2\frac{\partial u}{\partial x_2}\right)-\frac{\partial}{\partial x_3}\left(k_3\frac{\partial u}{\partial x_3}\right)=f(x_1,x_2,x_3), \\ \forall (x_1,x_2,x_3)\in \Omega, \\ 0<\sigma_{\min}\le k_1(x_1,x_2,x_3),\quad k_2(x_1,x_2,x_3),\quad k_3(x_1,x_2,x_3)\le \sigma_{\max}, \\ u(x_1,x_2,x_3)=0,\qquad \forall (x_1,x_2,x_3)\in \Gamma=\partial\Omega, \end{gathered} \tag{1}$$

on the unit cube $[0,1]^3$. Let the domain be discretized by a uniform grid with mesh size $h=\frac{1}{n}$.

2.1 Nonconforming Finite Element Method

Let $\mathcal{T}_h$ is a decomposition of Ω with $n\times n\times n$ cubes.

The weak formulation of the problem (1) reads as follows: for $f\in L^2(\Omega)$ find $u\in\mathcal{V}\equiv H_0^1(\Omega)=\{v\in H^1(\Omega): v=0 \text{ on } \Gamma\}$, satisfying

$\mathcal{A}(u,v)=(f,v)\quad \forall v\in H_0^1(\Omega)$, where $\mathcal{A}(u,v)=\int_\Omega \sum_{i=1}^3 k_i\frac{\partial u}{\partial x_i}\frac{\partial v}{\partial x_i}dx$.

The above variational problem is then discretized using the finite element method, i.e., the continuous space $\mathcal{V}$ is replaced by a finite dimensional subspace $\mathcal{V}_h$. Then the finite element formulation is:

find $u_h\in\mathcal{V}_h$, satisfying $\mathcal{A}(u_h,v_h)=(f,v_h)\quad \forall v_h\in\mathcal{V}_h$.

The resulting discrete problem to be solved is then a linear system of equations

$$A^{(FEM)}\mathbf{u}_h=\mathbf{f}_h, \tag{2}$$

where $\mathbf{u}_h$ stands for the vector of unknown degrees of freedom, $A^{(FEM)}$ and $\mathbf{f}_h$ are the corresponding global stiffness matrix and global right hand side.

Non-conforming finite elements based on *rotated* multilinear shape functions were introduced by Rannacher and Turek [14] as a class of simple elements for the Stokes problem. Some more details about non-conforming finite elements can be found, e.g., in [3,6,9]. The cube $[-1,1]^3$ is used as a reference element $\hat{e}$ to define the isoparametric rotated trilinear element $e\in\mathcal{T}_h$. Let $\Psi_e:\hat{e}\to e$ be the corresponding trilinear one-to-one transformation, and let the nodal basis functions be determined by the relation

$\{\phi_i\}_{i=1}^6=\{\hat{\phi}_i\circ\Psi_e^{-1}\}_{i=1}^6,\qquad \{\hat{\phi}_i\}\in \text{span}\{1,x_1,x_2,x_3,x_2^2-x_1^2,x_1^2-x_3^2\}$,

where '$\circ$' denotes the composition of functions. Shape functions $\{\hat{\phi}_i\}_{i=1}^6$ used in presented experiments are found by the point-wise interpolation condition $\hat{\phi}_i(b_e^j)=\delta_{ij}$, where b_e^j, j=1,6 are the centers of the faces of the cube $\hat{e}$.

2.2 Finite Difference Method

Let us consider the usual seven-point centered difference approximation for problem (1). This discretization leads to a system of linear algebraic equations

$$A^{(FDM)}\mathbf{u} = \mathbf{b}$$

where the vector of unknowns $\mathbf{u}$ has size $(n-1)^3$. If the grid points are ordered along the x_1 and x_2 directions first, the matrix $A^{(FDM)}$ admits a block-tridiagonal structure. The matrix $A^{(FDM)}$ can be written in the following form $A^{(FDM)} = tridiag(A_{i,i-1}, A_{i,i}, A_{i,i+1})$, $i = 1, 2, \ldots, n-1$, where the diagonal blocks $A_{i,i}$ are block-tridiagonal matrices which corresponds to one x_3-plane and the off-diagonal blocks are diagonal matrices.

3 MIC(0) Factorization Preconditioning

Let us first recall some well known facts about the modified incomplete factorization MIC(0). Let us decompose the real $N \times N$ matrix A in the form $A = D - L - L^T$, where D is the diagonal and $(-L)$ is the strictly lower triangular part of A. Then we consider the approximate factorization of A which has the following form:

$$M_{MIC(0)}(A) = (X - L)X^{-1}(X - L)^T,$$

where $X = \mathrm{diag}(x_1, \ldots, x_N)$ is a diagonal matrix determined such that A and $M_{MIC(0)}$ have equal row sums. We restrict ourselves to the case when $X > 0$, i.e., when $M_{MIC(0)}$ is positive definite. In this case, the MIC(0) factorization is called *stable*. Concerning the stability of the MIC(0) factorization, we have the following theorem [8].

Theorem 1. *Let $A = (a_{ij})$ be a symmetric real $N \times N$ matrix and let $A = D - L - L^T$ be the splitting of A. Let us assume that (in an element-wise sense)*

$$L \geq 0,\ A\mathbf{e} \geq 0,\ A\mathbf{e} + L^T\mathbf{e} > 0, \qquad \mathbf{e} = (1, \cdots, 1)^T \in \mathbb{R}^N,$$

i.e., that A is a weakly diagonally dominant matrix with non-positive off-diagonal entries and that $A+L^T = D-L$ is strictly diagonally dominant. Then the relation

$$x_i = a_{ii} - \sum_{k=1}^{i-1} \frac{a_{ik}}{x_k} \sum_{j=k+1}^{N} a_{kj} > 0 \tag{3}$$

holds and the diagonal matrix $X = \mathrm{diag}(x_1, \cdots, x_N)$ defines a stable MIC(0) factorization of A.

Remark 1. The numerical tests presented in this work are performed using the perturbed version of MIC(0) algorithm, where the incomplete factorization is

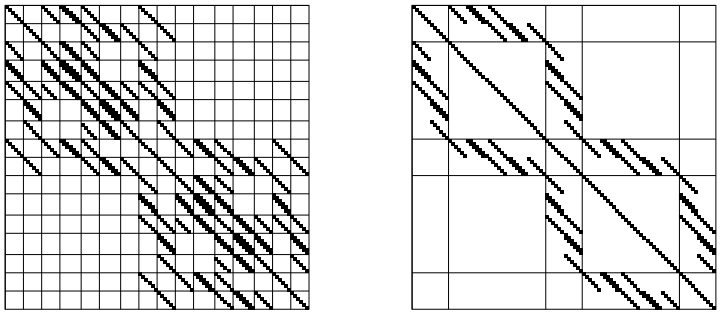


Fig. 1. Sparsity structure of the matrix A on the left and matrix B on the right, for the division of Ω into 2x2x6 hexahedrons. Non-zero elements are drawn with small squares.

applied to the matrix $\tilde{A} = A + \tilde{D}$. The diagonal perturbation $\tilde{D} = \tilde{D}(\xi) = \mathrm{diag}(\tilde{d}_1, \ldots \tilde{d}_N)$ is defined as follows:

$$\tilde{d}_i = \begin{cases} \xi a_{ii} & \text{if} \quad a_{ii} \geq 2w_i, \\ \xi^{1/2} a_{ii} & \text{otherwise,} \end{cases}$$

where $0 < \xi < 1$ is a constant and $w_i = -\sum_{j>i} a_{ij}$.

The idea of the parallel preconditioner is to apply the MIC(0) factorization on an auxiliary matrix B. The matrix B has a special block structure, which allows a scalable parallel implementation.

Following the standard FEM assembling procedure we write A in the form $A = \sum_{e \in \omega_h} L_e^T A_e L_e$, where A_e is the element stiffness matrix, L_e stands for the restriction mapping of the global vector of unknowns to the local one corresponding to the current element e. Let us consider the following approximation B_e of A_e:

$$A_e = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} & a_{15} & a_{16} \\ a_{21} & a_{22} & a_{23} & a_{24} & a_{25} & a_{26} \\ a_{31} & a_{32} & a_{33} & a_{34} & a_{35} & a_{36} \\ a_{41} & a_{42} & a_{43} & a_{44} & a_{45} & a_{46} \\ a_{51} & a_{52} & a_{53} & a_{54} & a_{55} & a_{56} \\ a_{61} & a_{62} & a_{63} & a_{64} & a_{65} & a_{66} \end{bmatrix}, \quad B_e = \begin{bmatrix} b_{11} & a_{12} & a_{13} & a_{14} & a_{15} & a_{16} \\ a_{21} & b_{22} & a_{23} & a_{24} & a_{25} & a_{26} \\ a_{31} & a_{32} & b_{33} & 0 & 0 & 0 \\ a_{41} & a_{42} & 0 & b_{44} & 0 & 0 \\ a_{51} & a_{52} & 0 & 0 & b_{55} & 0 \\ a_{61} & a_{62} & 0 & 0 & 0 & b_{66} \end{bmatrix}.$$

The local numbering follows the pairs of the opposite nodes of the reference element. The diagonal entries of B_e are modified to hold the row-sum criteria. Assembling the locally defined matrices B_e we get the global matrix $B = \sum_{e \in \omega_h} L_e^T B_e L_e$.

The sparsity structure of the matrices A and B is illustrated by Fig. 1. Lexicographic node numbering is used. The important property of the matrix B is that its diagonal blocks are diagonal matrices. This allows a parallel implementation, see [1, 2].

4 Circulant Block-Factorization Preconditioning

Let us recall that a circulant matrix C has the form $(C_{k,j}) = \left(c_{(j-k) \bmod m}\right)$, where m is the size of C. Any circulant matrix can be factorized as $C = F\Lambda F^*$, where Λ is a diagonal matrix containing the eigenvalues of C, and F is the Fourier matrix $F = \frac{1}{\sqrt{m}} \left\{ e^{2\pi \frac{jk}{m} \mathbf{i}} \right\}$, $0 \leq j, k \leq m-1$. Here $\mathbf{i}$ stands for the imaginary unit.

We use now the general form of the CBF preconditioning matrix M for the matrix $A^{(FDM)}$ by

$$M_{CBF} = \text{tridiag}(C_{i,i-1}, C_{i,i}, C_{i,i+1}) \qquad i = 1, 2, \ldots n-1,$$

where $C_{i,j} = Block - Circulant(A_{i,j})$ is block-circulant approximation of the corresponding block $A_{i,j}$ [12, 13]. The approach of defining block-circulant approximations can be interpreted as simultaneous averaging of the matrix coefficients and changing of the Dirichlet boundary conditions to periodic ones.

The algorithm (sequential and parallel) of the CBF preconditioner is described in [10, 11].

5 Numerical Tests

The numerical tests presented in this section illustrate the convergence rate as well as the parallel performance of the developed algorithms for 3D elliptic problems. We consider further test problems with variable coefficients in the form

$$\frac{\partial}{\partial x_1}\left[\left(1 + \frac{\epsilon}{2}\sin\left(2\pi\left(x_1 + x_3\right)\right)\right)\frac{\partial u}{\partial x_1}\right] + \frac{\partial}{\partial x_2}\left[\left(1 + \frac{\epsilon}{2}\sin\left(2\pi\left(x_1 + x_2\right)\right)\right)\frac{\partial u}{\partial x_2}\right] + \frac{\partial}{\partial x_3}\left[\left(1 + \epsilon e^{x_1+x_2+x_3}\right)\frac{\partial u}{\partial x_3}\right] = f\left(x_1, x_2, x_3\right) \tag{4}$$

where $\epsilon \in [0, 1]$ is a parameter. It is well known that the circulant preconditioners are competitive with the incomplete LU factorization for moderately varying coefficients. This reflects the averaging of the coefficients, used in the block-circulant approximations.

The right hand side f is chosen in such a way that the problem (4) has solution

$$u\left(x_1, x_2, x_3\right) = \sin 2\pi x_1 \sin 2\pi x_2 \sin 2\pi x_3.$$

The computations are done with double precision. The iteration stopping criterion is

$$||\mathbf{r}^{N_{it}}||_{M^{-1}} / ||\mathbf{r}^0||_{M^{-1}} < 10^{-3},$$

where $\mathbf{r}^j$ stands for the residual at the jth iteration step of the preconditioned conjugate gradient method. The codes have been implemented in C/C++ and the parallelization has been facilitated using the MPI [15, 16] library. We report

Table 1. Number of iterations and execution times of the algorithms.

N		N_{it}		Error		Execution time T_1			
						CLX		SP5	
MIC(0)	CBF	MIC(0)	CBF	MIC(0)	CBF	MIC(0)	CBF	MIC(0)	CBF
$\epsilon = 0$									
101 376	32 768	13	5	3.29 E-3	3.02 E-3	0.55	0.21	0.23	0.16
338 688	110 592	15	5	1.52 E-3	1.37 E-3	1.29	1.06	0.80	1.10
798 720	262 144	17	5	8.89 E-4	7.78 E-4	5.50	1.62	2.16	1.31
2 681 856	884 736	21	4	4.64 E-4	3.50 E-4	14.79	7.17	9.45	7.79
6 340 608	2 097 152	21	4	3.54 E-4	1.98 E-4	55.29	11.36	23.38	11.06
21 344 256	7 077 888	25	4	2.30 E-4	8.83 E-5		57.94	90.69	71.20
$\epsilon = 0.1$									
101 376	32 768	19	19	5.09 E-3	3.08 E-3	0.79	0.60	0.33	0.43
338 688	110 592	22	21	2.71 E-3	1.40 E-3	1.85	3.69	1.15	3.71
798 720	262 144	29	24	1.62 E-3	7.96 E-4	9.17	6.51	3.61	4.69
2 681 856	884 736	38	32	1.06 E-3	3.58 E-4	26.21	45.02	16.75	47.23
6 340 608	2 097 152	42	37	5.56 E-4	2.02 E-4	108.07	79.48	45.69	72.94
21 344 256	7 077 888	59	48	6.27 E-4	9.04 E-5		547.16	209.27	655.31
$\epsilon = 1$									
101 376	32 768	14	23	1.19 E-2	3.53 E-3	0.59	0.72	0.25	0.34
338 688	110 592	19	30	5.53 E-3	1.60 E-3	1.61	5.14	1.00	4.27
798 720	262 144	28	36	3.13 E-3	9.10 E-4	8.86	9.18	3.49	5.01
2 681 856	884 736	31	47	1.67 E-3	4.09 E-4	21.51	65.25	13.74	59.92
6 340 608	2 097 152	41	58	1.15 E-3	2.31 E-4	105.56	122.69	44.63	93.93
21 344 256	7 077 888	64	80	8.41 E-4	1.03 E-4		882.02	226.71	1001.33

the results of the experiments executed on Linux clusters located in Bologna, Italy. In our experiments, times have been collected using the MPI provided timer and the best results from multiple runs are reported. We present the elapsed time T_p in seconds on p processors, the speed-up $S_p = T_1/T_p$, and the parallel efficiency $E_p = S_p/p$.

Table 1 contains the number of iterations and the execution times collected on two clusters: CLX and SP5. CLX is an IBM Linux Cluster 1350 made of 512 2-way IBM X335 nodes. Each computing node contains 2 Xeon Pentium IV processors running at 3 GHz and 2 GB of RAM. Nodes are interconnected via a Myrinet network with a maximum bandwidth of 256 Mb/s. We have used IBM Visual Age compiler with options "-align -tpp7 -O3 -xN". SP5 is an IBM SP Cluster 1600, made of 64 nodes p5–575 (see `http://www.ibm.com/servers/eserver/pseries/library/sp_books/`) interconnected with a pair of connections to the Federation HPS (High Performance Switch). Globally the machine has 512 IBM Power5 processors and 1.2 TB of memory. A p5–575 node contains 8 SMP processors Power5 at 1.9 GHz and have at least 16 GB of memory. The HPS interconnect is capable of a bandwidth of up to 2 GB/s unidirectional.

First two columns of Table 1 show the size of the discrete problem. Next two columns report the number of iterations, the maximal error of the obtained solution of (4) is shown in next columns. For Laplace equation (for $\epsilon = 0$) for

Table 2. Execution time for parameter $\epsilon = 1$

p	MIC(0)						CBF					
	n											
	32	48	64	96	128	192	33	49	65	97	129	193
IBM Linux Cluster 1350												
1	0.59	1.61	8.86	21.51	105.56		0.72	5.14	9.18	65.25	122.69	882.02
2	0.33	1.44	4.84	18.52	58.88		0.45	3.06	6.03	39.10	90.35	536.99
3								1.98		26.38		358.07
4	0.22	0.94	2.81	10.27	32.04	154.37	0.24	1.66	3.43	21.33	49.82	293.03
6								1.11		14.53		198.28
8	0.18	0.68	2.00	6.14	17.93	91.32	0.12	0.77	1.89	11.35	27.04	157.66
12								0.54		7.89		111.16
16	0.48	0.70	2.43	4.76	13.31	63.31	0.07	0.41	0.86	6.10	14.79	88.26
24								0.28		3.87		58.41
32	0.24	0.54	1.33	3.96	10.08	42.51	0.08		0.11	2.83	8.36	45.11
48		0.66	1.36	3.39	11.88	46.60		0.22		1.97		31.80
64			2.26	3.93	7.51	30.03			0.39		3.59	23.78
IBM SP Cluster 1600												
1	0.25	1.00	3.49	13.74	44.63	226.71	0.34	4.27	5.01	59.92	93.93	1001.33
2	0.16	0.52	1.82	7.16	23.94	121.03	0.18	2.11	2.27	28.71	38.97	438.48
3								1.44		19.11		273.96
4	0.10	0.36	0.98	4.03	13.06	63.31	0.09	1.07	1.12	14.27	18.97	196.92
6								0.71		9.41		131.47
8	0.07	2.70	0.72	2.30	7.77	42.05	0.05	0.56	0.62	7.20	9.83	119.34
12								0.60		5.75		80.12
16	1.27	2.56	0.49	1.31	4.15	21.12	0.38	0.35	0.47	4.40	6.66	61.20
24								0.69		2.99		41.58
32	0.10	0.20	0.40	0.99	2.85	13.65	0.33		0.30	2.36	3.55	31.52
48		2.50	3.97	1.34	2.26	11.31		0.16		1.62		21.63
64			2.05	0.86	1.97	10.27			0.19		2.09	16.64

similar mesh size used in FEM and FDM discretization the obtained accuracy has the same order but the FEM discretization leads to a system of linear equations with approximately three times more unknowns. This leads to the larger execution time of MIC(0) algorithm. For problems with strongly varying coefficients MIC(0) algorithm has a preference and regardless of larger size of the discrete problem the MIC(0) algorithm is faster.

Table 2 shows the parallel execution time for the problem with strongly varying coefficients (for parameter $\epsilon = 1$). The comparison shows that the MIC(0) algorithm is faster except for relatively large mesh size on large number of processors. This fact confirms our general expectations that the incomplete Cholesky factorization is robust preconditioner (in this case — for problems with strongly varying coefficients) and that the CBF algorithm is highly parallelizable.

Table 3 compare the parallel execution time per one PCG iteration. The comparison shows that the MIC(0) algorithm is faster except execution on more than 16 processors on CLX cluster. The speed-up obtained on both clusters

Table 3. Execution time per iteration of the algorithms.

n	T_1^{it}	T_2^{it}	T_3^{it}	T_4^{it}	T_6^{it}	T_8^{it}	T_{12}^{it}	T_{16}^{it}	T_{24}^{it}	T_{32}^{it}	T_{48}^{it}	T_{64}^{it}	T_{96}^{it}
MIC(0) algorithm, IBM Linux Cluster 1350													
32	0.039	0.022		0.015		0.012		0.032		0.016			
48	0.080	0.072		0.047		0.034		0.035		0.027	0.033		
64	0.305	0.167		0.097		0.069		0.084		0.046	0.047	0.078	
96	0.672	0.579		0.321		0.192		0.149		0.124	0.106	0.123	
128	2.513	1.402		0.763		0.427		0.317		0.240	0.283	0.179	
192				2.375		1.405		0.974		0.654	0.717	0.462	
MIC(0) algorithm, IBM SP Cluster 1600													
32	0.016	0.011		0.007		0.005		0.085		0.007			
48	0.049	0.026		0.018		0.135		0.128		0.010	0.125		
64	0.120	0.063		0.034		0.025		0.017		0.014	0.137	0.071	
96	0.429	0.224		0.126		0.072		0.041		0.031	0.042	0.027	
128	1.062	0.570		0.311		0.185		0.099		0.068	0.054	0.047	
192	3.487	1.862		0.974		0.647		0.325		0.210	0.174	0.158	
CBF algorithm, IBM Linux Cluster 1350													
33	0.023	0.018		0.009		0.004		0.003		0.003			
49	0.147	0.097	0.061	0.052	0.035	0.023	0.016	0.012	0.008		0.006		
65	0.211	0.160		0.091		0.048		0.021		0.012		0.009	
97	1.193	0.806	0.534	0.438	0.295	0.234	0.153	0.120	0.075	0.056	0.038		0.025
129	1.718	1.500		0.830		0.453		0.238		0.118		0.059	
193	9.750	6.514	4.233	3.529	2.388	1.913	1.244	0.985	0.654	0.517	0.364	0.243	0.174
CBF algorithm, IBM SP Cluster 1600													
33	0.013	0.007		0.003		0.002		0.001		0.001			
49	0.135	0.066	0.045	0.033	0.022	0.017	0.015	0.011	0.007	0.004	0.004		
65	0.130	0.058		0.029		0.015		0.011		0.007		0.004	
97	1.232	0.588	0.391	0.292	0.192	0.147	0.117	0.089	0.060	0.046	0.031		0.020
129	1.561	0.643		0.312		0.161		0.109		0.057		0.033	
193	12.149	5.377	3.356	2.381	1.608	1.462	0.980	0.748	0.507	0.383	0.262	0.201	0.141

is reported in Table 4. As it was expected, the parallel features of the CBF algorithm lead to higher speed-up. Moreover, a super-linear speed-up is observed on SP5 cluster. The main reasons for this fact can be related to splitting the entire problem into subproblems which helps memory management, in particular allows for better usage of cache memories of individual parallel processors.

6 Concluding Remarks and Future Works

In this paper we concerned with the numerical solution of 3D elliptic problems. After discretization, such problems reduce to the solution of linear systems. We use two preconditioners: the Modified Incomplete Cholesky factorization MIC(0) and the Circulant Block-Factorization (CBF) preconditioning. Presented numerical results show that the rate of convergence of the CBF preconditioner is the same as for ILU factorization. We reported on the parallel performance of the studied preconditioners applied to the PCG algorithm. The developed MPI

Table 4. Speed-up of the algorithms.

n	S_2	S_3	S_4	S_6	S_8	S_{12}	S_{16}	S_{24}	S_{32}	S_{48}	S_{64}	S_{96}
MIC(0) algorithm, IBM Linux Cluster 1350												
32	1.751		2.505		3.122		1.219		2.385			
48	1.111		1.702		2.350		2.292		2.902	2.432		
64	1.826		3.118		4.394		3.633		6.615	6.389	3.883	
96	1.160		2.091		3.489		4.500		5.389	6.325	5.446	
128	1.792		3.290		5.882		7.919		10.430	8.861	14.003	
192			1.000		1.690		2.438		3.627	3.312	5.131	
MIC(0) algorithm, IBM SP Cluster 1600												
32	1.450		2.143		2.885		0.190		2.239			
48	1.852		2.765		0.369		0.389		4.593	0.397		
64	1.886		3.475		4.718		6.945		8.485	0.876	1.677	
96	1.913		3.396		5.893		10.364		13.588	10.029	15.355	
128	1.862		3.410		5.719		10.725		15.526	19.526	22.204	
192	1.872		3.579		5.386		10.720		16.566	20.040	21.983	
CBF algorithm, IBM Linux Cluster 1350												
33	1.33		2.73		5.88		9.16		8.37			
49	1.52	2.40	2.82	4.26	6.28	9.11	12.00	17.52		22.97		
65	1.32		2.32		4.36		9.96		17.95		24.76	
97	1.48	2.23	2.73	4.04	5.10	7.79	9.93	16.01	21.32	31.25		47.48
129	1.15		2.07		3.80		7.22		14.56		29.35	
193	1.50	2.30	2.76	4.08	5.10	7.84	9.90	14.90	18.86	26.80	40.17	55.96
CBF algorithm, IBM SP Cluster 1600												
33	1.94		3.80		7.24		8.73		13.76			
49	2.03	3.02	4.09	6.12	7.97	9.27	12.75	18.11	35.74	32.65		
65	2.23		4.52		8.59		11.52		18.88		31.60	
97	2.09	3.15	4.22	6.43	8.41	10.52	13.84	20.53	26.88	39.38		63.06
129	2.43		5.01		9.71		14.36		27.41		47.82	
193	2.26	3.62	5.10	7.56	8.31	12.39	16.24	23.95	31.68	46.33	60.49	85.93

codes provide new effective tool for solving of large-scale problems in realistic time on a coarse-grain parallel computer systems.

The experimental data collected from two clusters is only preliminary. In the near future we plan, first, to complete the performance studies by running our code on a number of additional machines. Second, we will extend our work to non-uniformly shaped domains, non-uniform discretization as well as situations when the proposed approach is embedded in a solver for non-linear problems. The implementation of the possibility for overlapping of computations and communications (see, e.g. [2, 7]) will be analyzed at the next step in development of the parallel code.

Acknowledgments. This research was partially supported by grant I-1402/2004 from the Bulgarian NSF and by the project BIS-21++ funded by FP6 INCO grant 016639/2005. The parallel numerical tests were supported via the EC Project HPC-EUROPA RII3-CT-2003-506079.

References

1. P. Arbenz, S. Margenov, Parallel MIC(0) preconditioning of 3D nonconforming FEM systems, *Iterative Methods, Preconditioning and Numerical PDEs*, Proceedings of IMET Conference, Prague, 2004, 12-15.
2. P. Arbenz, S. Margenov, Y. Vutov, Parallel MIC(0) Preconditioning of 3D Elliptic Problems Discretized by Rannacher–Turek Finite Elements, *Comput. Math. Appl.* (to appear)
3. D. N. Arnold, F. Brezzi, Mixed and nonconforming finite element methods: implementation, postprocessing and error estimates, *RAIRO Model. Math. Anal. Numer.*, **19**, 1985, 7–32.
4. O. Axelsson, *Iterative solution methods*, Cambridge University Press, Cambridge, 1994.
5. A. O. H. Axelsson, M.G. Neytcheva, *Supercomputers and numerical linear algebra*, KUN, Nijmegen, 1997.
6. G. Bencheva, S. Margenov, Parallel incomplete factorization preconditioning of rotated linear FEM systems, *J. Comput. Appl. Mech.*, **4** (2), 2003, 105–117.
7. G. Bencheva, S. Margenov, J. Starý. Parallel PCG Solver for Nonconforming FE Problems: Overlapping of Communications and Computations. *Large-Scale Scientific Computing*, I. Lirkov, S. Margenov, and J. Waśniewski eds., *Lecture notes in computer sciences*, **3743**, Springer Verlag, 646–654, 2005.
8. R. Blaheta, Displacement decomposition—incomplete factorization preconditioning techniques for linear elasticity problems, *Numer. Linear Algebra Appl.*, **1**, 1994, 107–126.
9. D. Braess, *Finite elements. Theory, fast solvers, and applications in solid mechanics*, Cambridge University Press, 1997.
10. I. Lirkov, S. Margenov, Parallel complexity of conjugate gradient method with circulant block-factorization preconditioners for 3D elliptic problems, *Recent Advances in Numerical Methods and Applications*, O. P. Iliev, M. S. Kaschiev, Bl. Sendov, P. V. Vassilevski, eds., World Scientific, Singapore, 1999, 482–490.
11. I. Lirkov, S. Margenov, M. Paprzycki, Parallel performance of a 3D elliptic solver, *Numerical Analysis and Its Applications II*, L. Vulkov, J. Waśniewski, P. Yalamov eds., *Lecture Notes in Computer Sciences*, **1988**, Springer Verlag, 2001, 535–543.
12. I. Lirkov, Y. Vutov, Parallel Performance of a 3D Elliptic Solver, *Proceedings of the International Multiconference on Computer Science and Information Technology*, Volume 1, M. Ganzha, M. Paprzycki, J. Wachowicz, K. Wecel eds., (CD-ROM), ISSN 1896–7094, 2006, 579–590.
13. I. Lirkov, Y. Vutov, The Convergence Rate and Parallel Performance of a 3D Elliptic Solver, *System Science*, to appear.
14. R. Rannacher, S. Turek, Simple nonconforming quadrilateral Stokes Element, *Numer. Methods Partial Differential Equations*, **8** (2), 1992, 97–112.
15. M. Snir, St. Otto, St. Huss-Lederman, D. Walker, J. Dongara, *MPI: The Complete Reference*, Scientific and engineering computation series, The MIT Press, Cambridge, Massachusetts, 1997, Second printing.
16. D. Walker and J. Dongara, MPI: a standard Message Passing Interface, *Supercomputer*, **63**, 1996, 56–68.

Proceedings of the International Multiconference on
Computer Science and Information Technology pp. 493–502
ISSN 1896-7094
© 2007 PIPS

On the computer simulation of heat and mass transfer in vacuum freeze–drying

K. Georgiev, N. Kosturski, I. Lirkov, and S. Margenov

Institute for Parallel Processing, Bulgarian Academy of Sciences
Acad. G. Bonchev str., Bl. 25–A, 1113 Sofia, Bulgaria
georgiev@parallel.bas.bg

Abstract. The paper is devoted to studying the problem of freeze–drying which is a process of the dehydrating frozen materials by sublimation under high vacuum. The mathematical and the computer models describing this process are presented. The discretizations used and the numerical treatment of the corresponding ordinary and partial differential equations is discussed. The results of some test experiments and the corresponding analysis can be found.

Key words: freeze–drying, heat and mass transfer, ordinary and partial differential equations, Runge-Kutta methods, heat conduction equation, finite element and finite difference methods

1 Introduction

Freeze–drying is a process by which a solvent is removed from a frozen material by sublimation under high vacuum ([4, 8]). The process of dehydration under vacuum starts from a frozen state. The heat is supplied to the drying material by: (i) conduction, (ii) radiation or (iii) both conduction and radiation but at a low rate in order to avoid the local melting. The main advantages of such kind of drying are:

- higher food quality of the dry product due to the minimum loss of flavor and aroma;
- the chances of the microbial to growth are minimized due to the water absence;
- no thermal and oxidizing processes in the dried product, etc.

In general, two different containers (cameras) are needed for the process of freeze–drying. The first one (food container) is for the product which will be dried. The second container consists of two subareas. One of them is fill–in with some material for the sorbtion of the water molecules, which come from the food container, and another is vacuum. The following three main stages can be seen in the process of freeze–drying:

1. Activation (dehydration) of the sorbent and preparation of the product to be dried. During this phase:
   - the freezing (drying) container is filled with the product to be dried and vacuumed;
   - the adsorbent which is located in the second container is warmed up, vacuumed and cooled to a room temperature.
2. Self–freezing of the product to be dried.
3. Drying in the conditions of an uniform sublimation of water steam from the product and their deposition in the sorbent.

This paper is devoted to study numerically (i) the *drying front movement in the freeze dryer*, (ii) *the temperature fields in the absorbent camera* and (iii) *the transfer of the sublimated water molecules and their retention in the sorbent granules.*

In this study *Zeolites granules* are used to sublimate the water molecules. Zeolites are a special type of silica–containing material which have a porous structure that makes them valuable as adsorbents and catalysts (for more details see e. g. [10]). They are used in different environmental problems, e. g. for reducing methyl bromide emissions, cleaning water, etc. Let us mention, that in practice not only natural zeolites but also synthetic ones can be used in the process of freeze–drying. Both, natural and synthetic zeolites have a rigid, three–dimensional crystalline structure (similar to honeycomb) consisting of a network of interconnected tunnels and cages ([11]).

The mathematical model of the process of freeze–drying is described by a system of time–depending differential equations. The model has a hierarchical structure and a splitting procedure according to the technological processes involved, is applied.

The rest of the paper is organized as follows. In Section 2 the mathematical model is discussed. The discretization used and the numerical treatment of the mathematical model is presented in Section 3. The results of a test experiment are given in Section 4. Finally, some conclusions and plans for future work can be found in Section 5.

2 The mathematical model

As was mentioned above, the process of freeze–drying has a hierarchical structure. Therefore, a splitting procedure according to the technological subprocesses involved, is applied.

2.1 Modelling the drying front in the freeze camera

The process of freeze–drying starts with the *self–freezing* of the product which will be dried. This first step of the process is relatively very fast and it takes short time after the connection between the *freezing* and *adsorption* cameras is open. There are three possibilities of heating the frozen product after this initial

stage (see Fig. 1). One can use a *radiation heater* (see Fig. 1, the left picture). In this case (so called *radiation heating*) the heating is coming from above and the dried layer is starting from above. If for heating is used a plate from bellow, then two different cases can occur. In the first one (see Fig. 1, the picture in the middle), the dried layer is starting from bellow and in the second case (see Fig. 1, the right picture) the dried layer is starting from above. Let us denote with t the time after starting of the process of drying, with L – the hight of the frozen product and with $Y = Y(t)$ the thickness of the dried layer at time t.

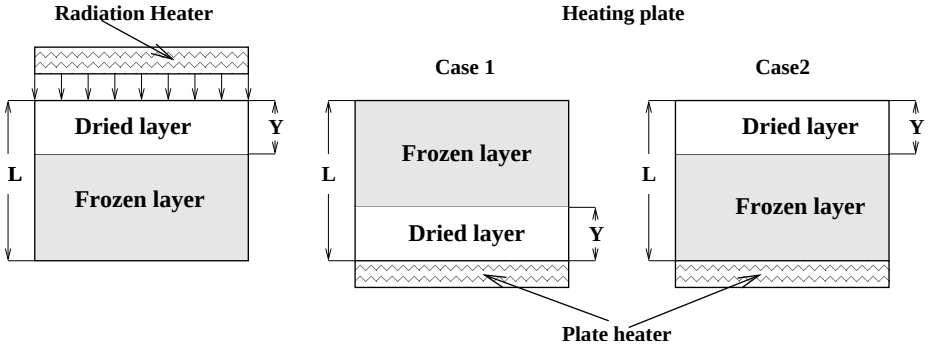


Fig. 1. Drying cases

Then, the drying front in the freezing camera can be described with one of the following three ordinary differential equations (ODE):

– radiation heating:

$$\frac{T_r - T_{sub}}{\frac{1}{h_r} + \frac{Y}{k_{cr}}} = \lambda \frac{P_{sub} - P_c}{\frac{1}{k_m} + \frac{Y}{\beta_{cr}}} = \rho\varepsilon \frac{dY}{dt}; \tag{1}$$

– heating plate, case 1:

$$\frac{T_p - T_{sub}}{\frac{1}{h_c} + \frac{Y}{k_{cr}}} = \rho\varepsilon \frac{dY}{dt}; \tag{2}$$

– heating plate, case 2:

$$\frac{T_p - T_{sub}}{\frac{1}{h_c} + \frac{L-Y}{k_{ice}}} = \lambda \frac{P_{sub} - P_c}{\frac{1}{k_m} + \frac{Y}{\beta_{cr}}} = \rho\varepsilon \frac{dY}{dt}. \tag{3}$$

The following notations are used in the last three equations:

– T_r – radiation temperature;
– T_p – plate temperature;
– T_{sub} – sublimation temperature;
– P_{sub} – water vapour pressure at the sublimation temperature;

- P_c – water vapour pressure at the condensation temperature;
- h_r – radiation heat transfer coefficient;
- h_c – contact heat transfer coefficient;
- k_{cr} – thermal conductivity of the dried layer;
- k_{ice} – thermal conductivity of the frozen layer;
- ρ – density coefficient;
- ε – moisture content;
- λ – latent of sublimation of the ice;
- β_{cr} – permeability of the dried layer;
- k_m – external mass transfer coefficient.

2.2 Modelling the process of heat and mass transfer

The process of heat and mass transfer in the adsorbent camera is described by the nonlinear partial differential equation (PDE) of parabolic type:

$$c\rho\frac{\partial T}{\partial t} = \mathcal{L}\,T + f(x,t,T), \qquad x \in \Omega,\ t > 0, \tag{4}$$

where

$$\mathcal{L}\,T = \sum_{i=1}^{d} \frac{\partial}{\partial x_i}\left(k(x,t)\frac{\partial T}{\partial x_i}\right). \tag{5}$$

The following notations are used in (4) and (5):

- $T(x,t)$ – unknown distribution of the temperature;
- d – dimension of the space ($d = 2$ in this study);
- $\Omega \in R^d$ – computational domain (see Fig. 3);
- $k = k(x,t) > 0$ – heat conductivity;
- $c = c(x,t) > 0$ – heat capacity;
- $\rho > 0$ – material density;
- $f(x,t,T)$ – right-hand side function, which is responsible for the non–linear process of transfer of water molecules in the adsorption container. This function couples the two systems of equation (one of (1), (2) or (3), and (4)).

Let us mention, that the adsorbent camera consists of several subparts, and each of them has their own conductivity, capacity and density coefficients.

Initial (6) and *boundary* (7) conditions are assigned to the parabolic equation (4):

$$T(x,0) = T_0(x), \qquad x \in \Omega, \tag{6}$$

$$T(x,t) = \mu(x,t), \qquad x \in \Gamma \equiv \partial\Omega, \quad t > 0, \tag{7}$$

where $T_0(x)$ is the initial distribution of the temperature in the computational domain and $\mu(x,t)$ is most often the surrounding of the freeze-drying device temperature.

3 Numerical treatment

3.1 Numerical treatment of the drying front equation

Let us write the three cases in describing the drying front in the freezing camera presented by (1), (2) or (3) in the form:

$$\frac{dY}{dt} = f(t, Y), \qquad t \in [0,\, T], \tag{8}$$

where the right–had–side function f is one of the left–side–functions in (1), (2) or (3) depending on the case of drying, dividing by $\rho\,\varepsilon\,\lambda$. An uniform discretization of the time interval $t_i = t_0 + i\tau$, where τ is the time–step, is used. For the numerical solution of (8) the Runge–Kutta method of order fourth is applied ([6]). Then

$$Y_{n+1} = Y_n + \sum_{r=1}^{4} p_j\, k_j, \tag{9}$$

where $k_j = \tau f\left(t_n + \alpha_j \tau,\, Y_n + \sum_{l=1}^{j-1} \beta_{j,l}\, k_l\right), \quad j = 1, 2, 3, 4$, and the parameters in the last formulas are taken according to the so–called *3/8-Rule* as follows:

$$p_1 = p_4 = \frac{1}{8},\ p_2 = p_3 = \frac{3}{8},\ \alpha_1 = 0,\ \alpha_2 = \frac{1}{3}\ \alpha_3 = \frac{2}{3}\ \alpha_4 = 1,$$

$$\beta_{2,1} = \frac{1}{3},\ \beta_{3,1} = -\frac{1}{3},\ \beta_{3,2} = \beta_{4,1} = \beta_{4,3} = 1,\ \beta_{4,2} = -1.$$

3.2 Numerical treatment of the heat and mass transfer front equation

For the numerical solution of the above discussed boundary value problem the Finite Element Method (FEM) in space is used ([2]). In fact, the Courant linear finite elements are chosen in this study. After the space discretization, the time derivatives are discretized via finite differences and well known Crank–Nicolson scheme is used ([7]). The computational domain is discretaised via triangle finite elements (see e.g. Fig. 4) using the computer mesh generator **Triangle** ([3, 9]). Options which are used for the mesh generating (generation of the main grid) are: *minimal angle of a triangle* (the most often used value is 30°) and *maximal area of a single triangle* (different for the different subdomains and depending of the geometry). During the single run this basic grid can be refined depending on a run parameter with a factor of $k = 1, 2, 3, \ldots$ and one can get respectively from each basic triangle $4, 16, 64, \ldots$ finner triangle elements, after bisection of the basic grid triangles sides.

Let us denote with K and M the stiffness and mass matrices coming from the Finite Element approach. In our case they can be written in the forms:

$$K = [K_{ij}]_{i,j=1}^{N} = \left[\int_{\Omega} k \, \nabla \Phi_i \cdot \nabla \Phi_j \, dx \right]_{i,j=1}^{N},$$

$$M = [M_{ij}]_{i,j=1}^{N} = \left[\int_{\Omega} c\rho \, \Phi_i \Phi_j \, dx \right]_{i,j=1}^{N}.$$

Then, the parabolic equation (4) can be written in matrix form as:

$$M \frac{dT}{dt} + KT = F, \tag{10}$$

where

$$F = \left[\int_{\Omega} f(x,t,T) \, \Phi_j \, dx \right]_{j=1}^{N}$$

If we denote with τ the time step, with T^{n+1} the solution (temperature) on the current time level and with T^n the solution on the previous time level, and do an approximation of the time derivative in (10) we will obtain the following system of linear algebraic equations for the nodal values of T^{n+1}:

$$\left(M + \frac{K\tau}{2} \right) T^{n+1} = \left(M - \frac{K\tau}{2} \right) T^n + \tau \frac{F^{n+1} + F^n}{2}. \tag{11}$$

The Preconditioned Conjugate Gradient (PCG) method with a Modified Incomplete Cholesky
(MIC(0)) Preconditioner ([1, 5]) is used to solve the system of algebraic equations (11).

3.3 Structure of the computer model

The computer code realizing the above presented algorithm consists of two main modules according to the splitting done. The structure of the code is as follows:

- Numerical solution of the drying front equations in drying freeze camera (Runge–Kutta methods for solution of nonlinear ODE);
- Numerical solution of the heat–mass transfer equation in adsorbent camera (Crank–Nikolson method and FEM for parabolic PDEs):
  - triangulation of the computational domain (computer mesh generator **Triangle**);
  - generation of the element stiffness and mass matrices and vectors;
  - assembling of the global stiffness and mass matrices and vectors;
  - solution of the systems of linear algebraic equations (PCG method with MIC(0) preconditioner);
  - visualization of the output results.

The program modules are developed using the algorithmic languages *Fortran* and C^{++} under *Linux/Unix* operating system.

4 Numerical test on a real–life experiment

Many numerical experiments with known exact solutions were run in order to fix the mesh and time parameters of the computer model. For the final test of the numerical algorithm discussed above, an operating experimental unit was used (see Fig. 2). In this experiment the case where for the heating is used a plate from bellow and the dried layer is started from bellow is used (see Fig. 1, case in the middle of the picture). The product which is subject to drying is put in the left container. In the case study this container is a *glass flask* and the drying product is *grated carrots* (see Fig. 2a)). The right container is the absorption camera (in this study it is again made from *glass*), where the zeolite granules are placed (see Fig.2b)).

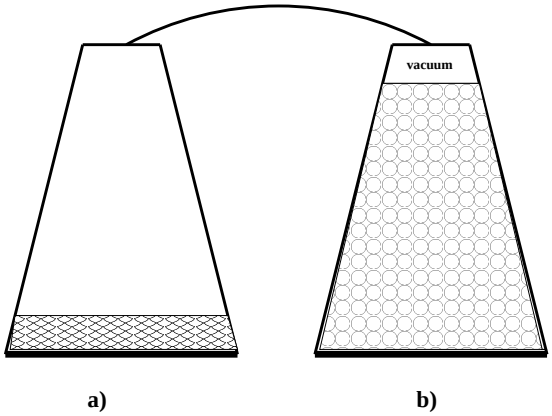


Fig. 2. *The experiment*

Two input parameters (data) for the solution of the heat–mass transfer equation in the adsorbent container are obtained as output results from solution of the drying front movement equation:

- quantity of the sublimated water molecules per a time unit, and
- the total time to finalize the process, i.e. the time interval for the parabolic PDE $[0, T]$.

The quantity of the sublimated water molecules for a time unit is multiplied by the adsorption heat coefficient in order to obtain the integral intensity of the heater.

We study the heat–mass transfer in the absorption camera via solving the above introduced parabolic boundary value problem. The computational domain (see Fig. 3) is discretaised via triangle finite elements (see Fig. 4) using the computer mesh generator **Triangle** ([3, 9]). Let us mention that there are big jumps in the values of the coefficients of the parabolic equation. The computational

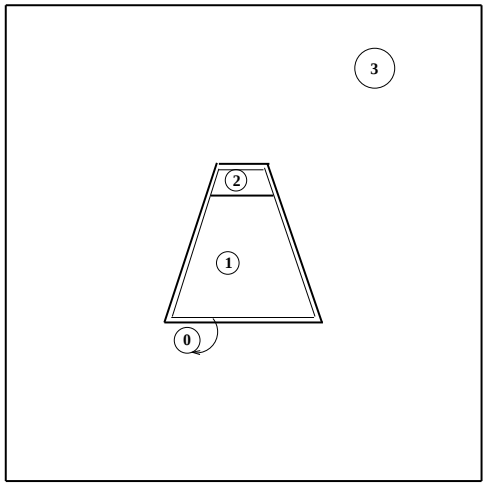


Fig. 3. *The computational domain in the case–study (0 – glass, 1 – zeolite granules, 2 – vacuum, 3 – air)*

domain consists of four strongly different according to their physical characteristics, subdomains: glass, zeolite granules, vacuum and air. In the beginning of the process the air around the container with zeolite granules is with a room temperature (e.g. about $20°C$). A number of computer experiments were done in order to obtain appropriate values of the mesh parameters and the time step. The scalability of the code were studied (see bellow some results showing the good scalability on several successive grids). Results from one of the provided computer test experiments, where a mesh with 6577 triangles and 3365 nodes and a time step $\tau = 10sec.$ were used are presented in this paper. The time to end the process of drying T is an input parameter. CPU time for running the code on Pentium IV, 1.5 GHz is $393.82sec.$ The temperature field obtained in the adsorption camera can be seen on Fig. 5, a), while on Fig. 5, b) one can see how the zeolite granules are filled in with water molecules coming from the drying container.

5 Conclusion

This paper is devoted to a numerical study of a freeze–drying process. Results from some of the performed numerical experiments are presented. In particular, the following subtasks are studied: (i) the drying front movement in the freeze dryer, (ii) the temperature fields in the absorbent camera and (iii) the transfer of the sublimated water molecules and their retention in the sorbent granules. Zeolites granules are used to sublimate the water molecules.

The mathematical model of the process of freeze–drying is described by a system of time–depending differential equations.

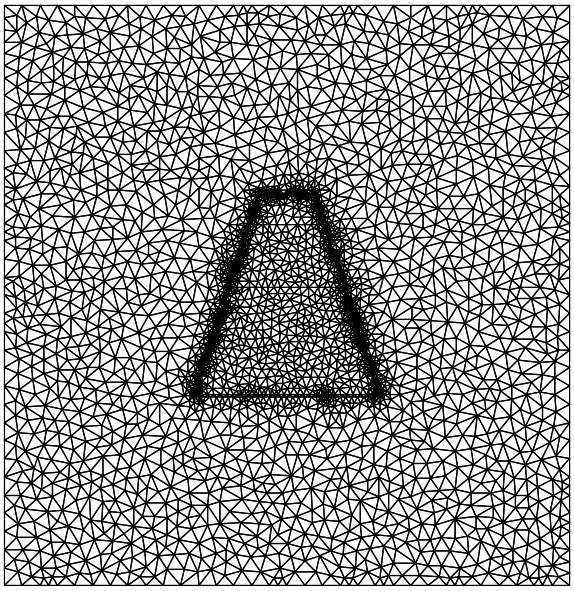


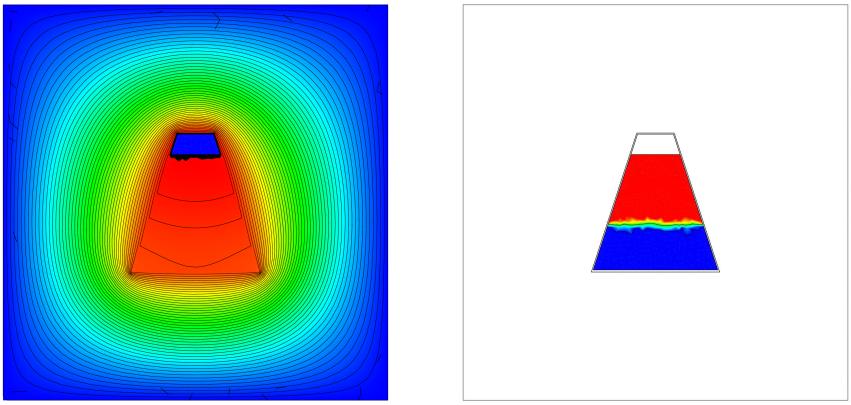
Fig. 4. *Finite element discretization of the computational domain*

For implementation of the presented mathematical algorithm a computer code is developed using the algorithmic languages *Fortran* and C^{++}. The scalability of the code was studied on three successive grids and the results are given in Table 1.

Table 1. Scalability of the code

Number of unknowns	CPU time	Ratio nodes	Ratio CPU time
$N_1 = 1174$	$T_1 = 137.14$	$N_2/N_1 = 1.85$	$T_2/T_1 = 1.91$
$N_2 = 2171$	$T_2 = 261.52$	$N_3/N_2 = 1.55$	$T_3/T_2 = 1.51$
$N_3 = 3365$	$T_3 = 393.82$	$N_3/N_1 = 2.866$	$T_3/T_1 = 2.872$

The output results of the numerical tests performed with the computer code realizing the presented algorithm are compared with measurements coming from a real–life experiment. These comparisons show that the solver could be successfuly used for simulation of the process of freezy–drying.



a) b)

Fig. 5. *The temperature field—a) and the adsorption of the water molecules—b), in the adsorption camera.*

Acknowledgment

The research reported in this paper was partly supported by the Bulgarian IST Center of Competence in 21st century – BIS-21++ funded by the European Commission in FP6 INCO via Grant 016639/2005 and by the Executive Agency of Bulgarian SME under Contract IF-02-37/01.11.2005.

References

1. O. Axelsson (1996), *Iterative Solution Methods*, Cambridge University Press.
2. S. Brenner, L. Scott (1994), *The mathematical theory of finite element methods*, Texts in applied mathematics, **15,** Springer-Verlag.
3. P. Frey, P.-L. George (2000), *Mesh Generation*, Hermes Science.
4. F. Jafar, M. Farid (2003), *Analysis of heat and mass transfer in freeze-drying,* Drying Technology **21,** No. 2, pp. 249–263.
5. I. Gustafsson (1996), *An incomplete factorization preconditioning method based on modification of element matrices,* BIT **36,** No. 1, pp. 86–100.
6. E. Hairer, S. P. Norsett, G. Wanner (2000, 2002), *Solving ordinary differential equations I, II,* Springer Series in Comp. Math.
7. D. Kincaid, W. Cheney (2002), *Numerical Analysis: Mathematics of Scientific Computing,* Thomson Learning.
8. A. Mujumdar (1999), *Handbook of Industrial Drying,* Marcel Dekker, Inc.
9. *TRIANGLE, a two dimensional quality mesh generator and delaunay triangulator,* Carnegie Mellon University, `http://www.cs.cmu.edu/~quake/triangle.html`
10. G. Tsitshvili, T. Andronikashvili, G. Kirov (1992), *Natural zeolites,* Prentice Hall.
11. `http://www.stcloudmining.com/what-is-zeolite.html`

Proceedings of the International Multiconference on
Computer Science and Information Technology pp. 503–515
ISSN 1896-7094
© 2007 PIPS

Image reconstruction from incomplete projection data by means of iterative algebraic algorithms

Nadiya Gubareni[1] and Mariusz Pleszczynski[2]

[1] Politechnika Częstochowska,
ul. Dąbrowskiego 69, 42-200 Częstochowa, Poland
gubareni@zim.pcz.pl,
[2] Politechnika Śląska,
ul. Kaszubska, 23, 44-101 Gliwice, Poland
dm101live@interia.pl

Abstract. In this paper we consider the problem of image reconstruction from incomplete projection data for some particular schemes of reconstruction. We present the numerical reconstruction algorithms for image reconstruction of high contrast objects from incomplete data. Numerical simulation results for a number of modeling objects with hight contrast are presented and discussed.

1 Introduction

In the recent years technique of computerized tomography found a wide application not only in medicine but also for solving many technical problems. In all these applications we deal with inverse problems which are those of image reconstruction from projections. In spite of the wide different physical methods to collect projection data there is a common mathematical nature of reconstruction problems: there is an unknown distribution of some physical parameter. A finite number of line integrals of this parameter can be estimated from physical measurements, and an estimate of the distribution of the original parameter is desired.

In many applications the projection data are often not available at each angle of view and may be very limited in number. In particular, such kind of problems arises in mineral industries and engineering geophysics connected with acid drainage, the stability of mine workers, mineral exploration and others [9, 10].

For these problems the projection operator can be represented algebraically and the problem of image reconstruction is reduced to solving a system of linear algebraic equations. For solving such systems there often used different kinds of algebraic iterative algorithms the most well-known from which are ART and MART algorithms [1, 3–8]. They are generally simple, flexible and permit to use *a priori* knowledge of the object before its reconstruction that is very important in many practical applications.

In this paper we consider the problems of image reconstruction from incomplete projection data for some particular reconstruction schemes which arise in

some problems of engineering geophysics and mineral industry. For solving these problems we use the iterative algebraic algorithms. Numerical simulation of solving problems for image reconstruction from incomplete projection data for some modeling objects, comparing evaluations of errors and rate of convergence of these algorithms are presented. It is shown then for some choice of parameters we can obtain a good enough quality of reconstruction with these algorithms for some large number of iterations.

2 Problem of incomplete projection data

Let $f(x, y)$ be a function which represents the spatial distribution of a physical parameter. If L is a line (ray) in the plane then the line integral

$$p_L = \int_L f(x, y) dL, \tag{1}$$

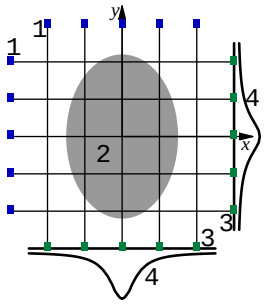
which is usually called a projection, is usually obtained from physical measurements.

From mathematical point of view the problem of reconstruction from projections is to find an unknown function $f(x, y)$ by means of a given set of projections p_L for all L. This problem was solved by J. Radon in 1917 in the form of the inversion formula which expresses f in terms of p. Unfortunately, this mathematical problem represents only an idealized abstraction of problems which occur in practical applications. In practice we are given only discrete projection data that estimate p for a finite number of rays. And we want to find an image function $f(x, y)$, i.e. a reconstructed estimate of the unknown object. Moreover, we are deal with limited precision of measured data and so the projection data are given with some errors. Therefore all these restrictions do not allow us to use the Radon inversion formula directly.

There exist two fundamentally different approaches for solving the image reconstruction problem. In one approach the problem is formulated for continuous functions f and p and the inversion formula is derived in this continuous model. This method is called the transform method approach [2]. The second approach is connected with discretization of the function f at the outset. The object f and measurements p then become vectors in the finite dimensional Euclidean space. And for solving this model we use the methods of linear algebra and optimization theory. This approach is called the fully discretized model [2].

If the number of projection data can be obtained large enough and for any directions then in these applications (in medicine, for example) it is more preferred to use the transform method approach. But in many practical applications the projection data cannot be available for each direction and its number is very limited. In this case we say that we have a problem of image reconstruction with incomplete projection data. In particular, such kind of problems arise in mineral industries and engineering geophysics connected with acid drainage, the stability of mine workers, mineral exploration and others [9, 10].

In dependence on the obtaining system of projections there are many image reconstruction schemes, the main of them are parallel and beam schemes in the two-dimensional space. Both of them are represented in Fig.1 and Fig.2.



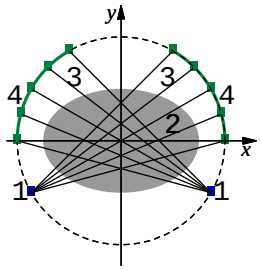
1 – Sources

2 – Object to be reconstructed

3 – Detectors

4 – Projections

Fig. 1. Parallel scheme of image reconstruction



1 – Sources

2 – Object to be reconstructed

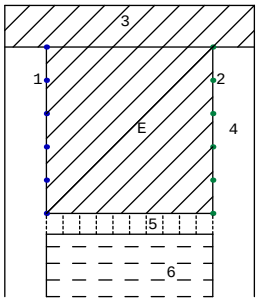
3 – Rays

4 – Detectors

Fig. 2. Beam scheme of image reconstruction

In some practical problems, in engineering for example, it is impossible to get projections from all directions because of the existing some important reasons (such as situation, size or impossibility of an access to a research object). This situation arises, for example, in the coal bed working. In such a coal bed during the preparing process for working in dependence on the scheme the access to longwalls may be very difficult or impossible at all. Sometimes it is impossible to access to one or two sides of longwalls, and sometimes it is impossible only to access to the basis but all the longwalls are accessible. Each this situation has its own scheme of obtaining information.

In this paper we present results for image reconstructions only for two different schemes, which are described below.

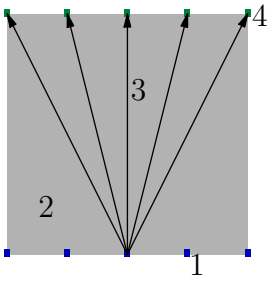


1 – Sources
2 – Detectors
3 – Unmined coal
4 – Heading
5 – Longwall with mechanized lining, belt conveyor flight, heading machine so on.
6 – Caving or filling
E – Researching coal bed

Fig. 3. The scheme of a coal bed working

2.1 System 1×1

Consider the scheme which is the worst from the view of obtaining projection data. In this scheme we have an access to a research coal bed from only two opposite sides. This situation often arise in engineering geophysics as well. In this case the sources of rays are situated only on one side and the detectors are situated on the opposite side of the research part of a coal bed. This scheme of obtaining information is shown in Fig.4 and we shall call it as the system 1×1.
As one may see this system is the most far from the common systems for image



1 – Sources of rays
2 – Research object
3 – Rays
4 – Detectors

Fig. 4. The system 1×1.

reconstruction problems (see Fig.1 and Fig.2), where projections can be obtained from any direction.

2.2 System $(1 \times 1,\ 1 \times 1)$

Consider the situation when we can have an access to all four sides of a coal bed. In this scheme the sources are situated onto two neighboring sides, and the detectors are situated on the opposite sides. So the projections can be obtained from two pair of the opposite sides. This situation is shown in Fig.5 and we shall call it as the system $(1 \times 1,\ 1 \times 1)$.

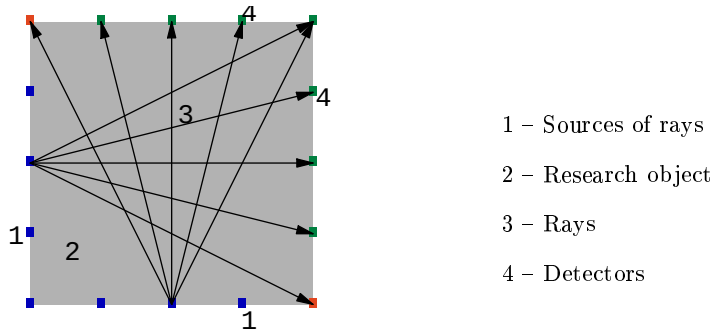


Fig. 5. The system $(1 \times 1,\ 1 \times 1)$

3 Reconstruction algebraic algorithms

At first, for constructing the full discrete model we include a reconstructed domain $D \subset R^2$ into a rectangle E and divide it into n small elements (pixels). The full discrete model of the problem of image reconstruction is based on the main principal that a research object has the constant distribution inside each pixel. So for any i-th pixel we can correspond an unknown x_i. Secondly, we assume that sources and detectors are points and the rays between them are lines. We denote by a_{ij} the length of the intersection of the i-th ray with j-th pixel. In this case the problem of image reconstruction is reduced to solving a system of linear algebraic equations:

$$\mathbf{A} \cdot \mathbf{x}^T = \mathbf{p}^T, \tag{2}$$

where:
$\mathbf{A} \in R^{m,n}$ is the projection matrix,
$\mathbf{x} \in R^n$ is the image vector,
$\mathbf{p} \in R^m$ is the measurement vector of projection data.

This system has a few characteristics: it is a rectangular as a rule and it has a very large dimension. For solving this system it is often used different kind of algebraic iterative algorithms which are based on the Kaczmarz method, the most well-known of which are the additive algorithm ART and the multiplicative algorithm MART (see [1], [3]-[8]). These algorithms are very flexible and allow to apply different *a priory* information about object before its reconstruction that is especially very important when we have an incomplete projection data. The basic idea of these algorithms is to run through all equations cyclically with modification of the present estimate $\mathbf{x}^{(k)}$ in such a way that the present equation with index i is fulfilled.

In this paper we use some modifications of these algorithms which we call ART-3 and MART-3.

Algorithm *ART*-3

1. $\mathbf{x}^{(0)} \in R^n$ is an arbitrary vector.
2. The k+1-th iteration is calculated in accordance with the following scheme:

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + S_k \frac{\mathbf{a}^{i(k)}}{\|\mathbf{a}^{i(k)}\|^2}, \tag{3}$$

where:

$$S_k = \begin{cases} 0, & \text{for } |p_i - (\mathbf{a}^{i(k)}, \mathbf{x}^{(k)})| \leq \varepsilon_i; \\ p_i - (\mathbf{a}^{i(k)}, \mathbf{x}^{(k)}), & \text{for } |p_i - (\mathbf{a}^{i(k)}, \mathbf{x}^{(k)})| \geq \varepsilon_i; \\ 2(p_i + \varepsilon_i - (\mathbf{a}^{i(k)}, \mathbf{x}^{(k)})), & \text{for } p_i + \varepsilon_i < (\mathbf{a}^{i(k)}, \mathbf{x}^{(k)}) < p_i + 2\varepsilon_i; \\ 2(-p_i + \varepsilon_i + (\mathbf{a}^{i(k)}, \mathbf{x}^{(k)})), & \text{for } p_i - 2\varepsilon_i < (\mathbf{a}^{i(k)}, \mathbf{x}^{(k)}) < p_i - \varepsilon_i, \end{cases} \tag{4}$$

where:
$\mathbf{a}^i$ is the i-th row of the matrix $\mathbf{A}$,
p_i is the i-th coordinate of the projection vector $\mathbf{p}$,
ε_i is a parameter which defines precision of a solution
and $i(k) = k \,(\mathrm{mod}\, m) + 1$.

This algorithm was investigated by G.T.Herman [6], and it was used successfully in medicine.

Algorithm *MART*-3

1. $\mathbf{x}^{(0)} \in R^n$ is an arbitrary vector and $\mathbf{x}^{(0)} > 0$.
2. The k+1-th iteration is calculated in accordance with the following scheme:

$$x_j^{(k+1)} = \left(\frac{p_i}{(\mathbf{a}^{i(k)}, \mathbf{x}^{(k)})} \right)^{\lambda_k^i a_{ij}} x_j^{(k)}, \tag{5}$$

where:
$\mathbf{a}^i$ is the i-th row of the matrix $\mathbf{A}$,
λ_k^i is a relaxation parameter,
p_i is the i-th coordinate of the projection vector $\mathbf{p}$,
and $i = k \,(\mathrm{mod}\, m) + 1$.

This algorithm was invented and reinvented in several fields. It was shown that it is convergent if $0 < \lambda_k^i a_{ij} \leq 1$ for all i, k, j, and its solution gives the solution of the linearly constrained entropy optimization problem (see [2],[4],[8]).

In this paper we consider that $\lambda_k^i = \lambda = const$ for all i, j, k and we use algorithms ART-3 and MART-3 with following constraining operators:

$$C_1[\mathbf{x}] = \begin{cases} \mathbf{x}, \text{ if } \mathbf{x} \in D; \\ 0, \text{ otherwise} \end{cases} \tag{6}$$

$$(C_2[\mathbf{x}])_i = \begin{cases} a, \text{ if } x_i < a; \\ x_i, \text{ if } a \leq x_i \leq b; \\ b, \text{ if } x_i > b \end{cases} \tag{7}$$

$$(C_3[\mathbf{x}])_j = \begin{cases} 0, \text{ if } p_i = 0 \text{ and } a_{ij} \neq 0; \\ x_j, \text{ otherwise} \end{cases} \tag{8}$$

4 Computer simulation and experimental results

In order to evaluate the goodness of the compute reconstruction of a high-construct image from a limited number of projections and incomplete data we tested different kind of geometric figures and reconstruction schemes.

An important factor in the simulation process of image reconstruction is the choice of modeling objects which describe the density distribution of research objects. In a coal bed, where we search the reservoirs of compressed gas or interlayers of a barren rock, the density distribution may be considered discrete and the density difference of these three environments (coal, compressed gas and barren rock) is significant. Therefore for illustration of the implementation of the algorithms working with incomplete data we chose the discrete functions with high contrast. In this paper we present only the results of computer reconstruction for two discrete figures and two reconstructions schemes 1×1 and $(1 \times 1, 1 \times 1)$.

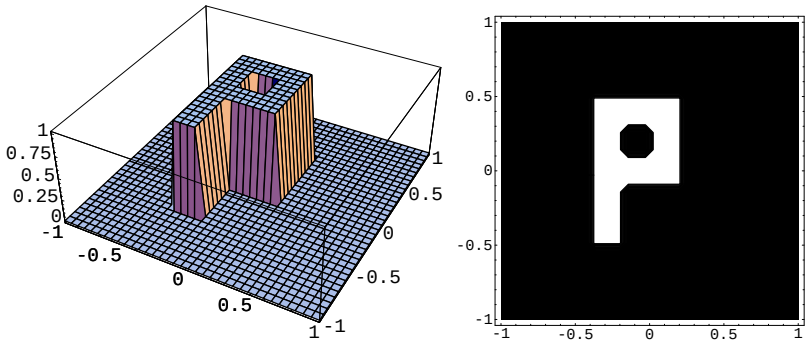


Fig. 6. The original function $f_1(x, y)$.

The first discrete function $f_1(x,y)$ is given by the following equation:

$$f_1(x,y) = \begin{cases} 1, & (x,y) \in D \subset E \subset \mathbb{R}^2, \\ 0, & \text{otherwise} \end{cases} \tag{9}$$

where E is a square $E = \{(x,y)\colon\ -1 \le x,y \le 1\}$, and D is a subset of E of the following form:

$$\begin{aligned} D = [-0.4,-0.2] \times [-0.5,0.5] \cup [-0.2,0.2] \times [0.3,0.5] \cup \\ \cup\, [-0.2,0.2] \times [-0.1,0.1] \cup [0,0.2] \times [0.1,0.3]. \end{aligned} \tag{10}$$

The plot of this function is presented in Fig.6, where the figure on the left side is a three-dimensional view of the plot of $f_1(x,y)$, and on the right side it is represented the two-dimensional view of the plot of $f_1(x,y)$ (the black color denotes the value 0 and the white color denotes the value 1).

The second figure is given by the following equation:

$$f_2(x,y) = \begin{cases} 1, & (x,y) \in D_1 \subset E \subset \mathbb{R}^2, \\ 2, & (x,y) \in D_2 \subset E \subset \mathbb{R}^2, \\ 3, & (x,y) \in D_3 \subset E \subset \mathbb{R}^2, \\ 4, & (x,y) \in D_4 \subset E \subset \mathbb{R}^2, \\ 0, & \text{otherwise} \end{cases} \tag{11}$$

where E is a square $E = \{(x,y)\colon\ -1 \le x,y \le 1\}$, and D_i are subsets of E of the following form: $D_1 = [-0.1,0.3] \times [0.3,0.4]$, $D_2 = [-0.3,-0.1] \times [-0.6,0.6]$, $D_3 = [-0.4,-0.3] \times [0,0.1]$, $D_4 = [-0.1,0.2] \times [-0.4,-0.3]$. The plot of this function is given in Fig.7.

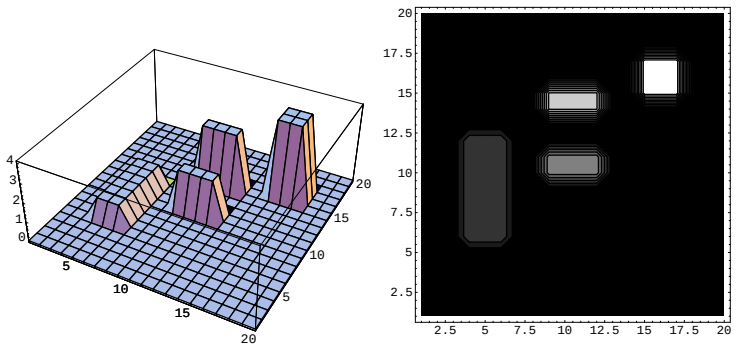


Fig. 7. The original function $f_2(x,y)$.

As was shown earlier (see, for example [4, 6, 7]), the image reconstruction of such objects from complete data gives a good enough results after 6-7 full iterations.

In this paper we present the numerical results of image reconstruction of chosen functions with two algebraic iterative algorithms ART-3 and MART-3 for two reconstruction schemes 1×1 and $(1 \times 1, 1 \times 1)$. We compare these results of reconstructions, and we investigate the influence of various parameters of these algorithms such as a pixel initialization, relaxation parameters, number of iterations and noise in the projection data on reconstruction quality. The convergence of these algorithms was studied in dependence on different these parameters. The convergence characteristic plots are given in view of plots for the maximum absolute error: $\Delta = \max_i |f_i - \bar{f}_i|$, the maximum relative error: $\delta\% = \frac{\max_i |f_i - \bar{f}_i|}{\max_i (f_i)} \cdot 100\%$, and the mean absolute error $\delta = \frac{1}{n} \sum_i |f_i - \bar{f}_i|$, where f_i is the value of a given modeling function in the center of the i-th pixel and $\bar{f}_i$ is the value of the reconstructed function in the i-th pixel.

The reconstruction result of $f_1(x, y)$ with algorithm ART-3 after 15 iterations in the reconstruction scheme $(1 \times 1, 1 \times 1)$ for $n = 20 \times 20$ pixels, $m = 644$ projections is presented in Fig.8. The plot of the reconstruction function is shown on the left side, and the plot of the mean absolute errors for this image reconstruction is shown on the right side.

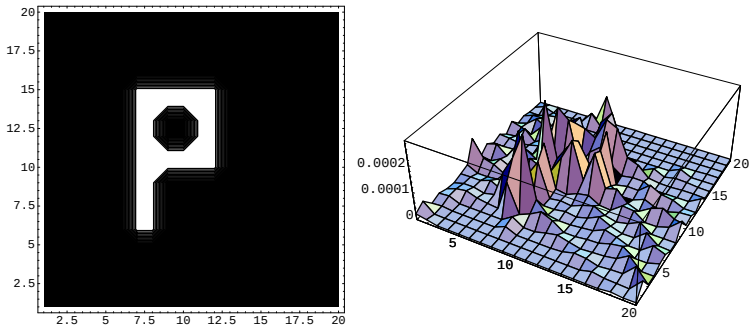


Fig. 8: The image reconstruction of $f_1(x, y)$ with ART-3 for $n = 20 \times 20$, $m = 644$, $iter = 15$ for the scheme $(1 \times 1, 1 \times 1)$.

For comparison this function $f_1(x, y)$ was reconstructed with multiplicative algorithm MART-3 for the same parameters and $\lambda = 6.9$ and the plots, which are presented in Fig. 9, illustrate the dependence of the maximum relative error and the mean absolute error on number of iterations with algorithms ART-3 and MART-3 in the system $(1 \times 1, 1 \times 1)$:

The same function $f_1(x, y)$ was also reconstructed in the system 1×1. The result of this reconstruction with algorithm ART-3 for $n = 20 \times 20$, $m = 788$ and 100 iterations is shown in Fig. 10.

The plots, which are presented in Fig.11, illustrate the dependence of the maximum relative error and the mean absolute error on number of iterations of image reconstruction of $f_1(x, y)$ with algorithm ART-3 in the system $(1 \times 1, 1 \times 1)$ Âăand in the system (1×1):

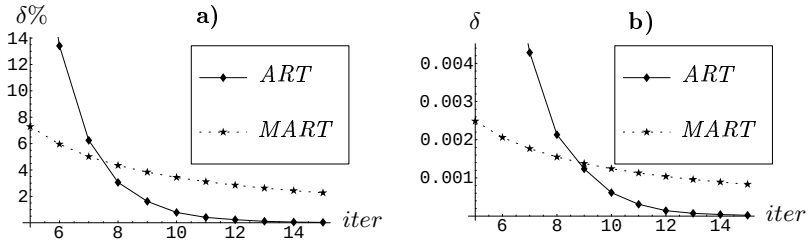


Fig. 9: Dependence of the maximum relative error (on the left side) and the mean absolute error (on the right side) on number of iterations for image reconstruction of $f_1(x, y)$ with algorithms ART-3 and MART-3 for $n = 400$, $m = 644$ in the system $(1 \times 1, 1 \times 1)$.

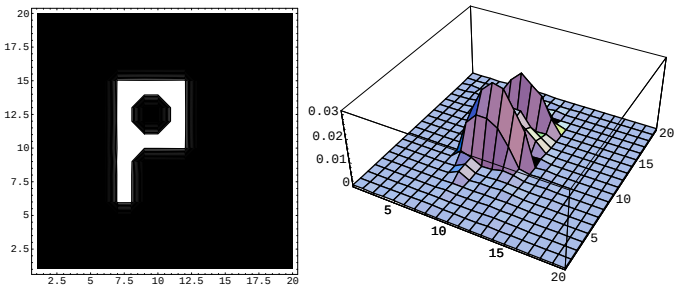


Fig. 10: The image reconstruction and the mean absolute error of $f_1(x, y)$ obtained with algorithm ART for $n = 20 \times 20$, $m = 788$, $iter = 100$ in the system 1×1.

The analogous reconstructions were obtained for the function $f_2(x, y)$. The results of image reconstructions for $f_2(x, y)$ with algorithm ART-3 for the same parameters are given in Fig. 12.

The results of reconstruction of the function $f_2(x, y)$ in the system 1×1 is shown in Fig. 13

The plots presented in Fig.14 illustrate the dependence of the maximum relative error and the mean absolute error on the number of iterations of image reconstruction of $f_2(x, y)$ with algorithm ART-3 in the system $(1 \times 1, 1 \times 1)$ and in the system $(1 \times 1, 1 \times 1)$:

All experimental results in the case of reconstruction of objects from limited projection data showed that the errors of reconstruction is constantly reduced with increasing the number of iterations. The following table shows the dependence of the maximum absolute error on the number of iterations for the algorithm ART-3 for two systems (1×1) and $(1 \times 1, 1 \times 1)$ and the same set of parameters which were chosen above:

$iter$	100	200	500	1000	1297	2000	system
Δ	0.0306	0.00201	1.209×10^{-6}	6.435×10^{-12}	4.218×10^{-15}	5.44×10^{-15}	1×1
$iter$	10	20	40	50	60	105	system
Δ	0.0077	9.837×10^{-6}	3.127×10^{-11}	3.985×10^{-14}	6.661×10^{-16}	8.881×10^{-16}	$1 \times 1\ 1 \times 1$

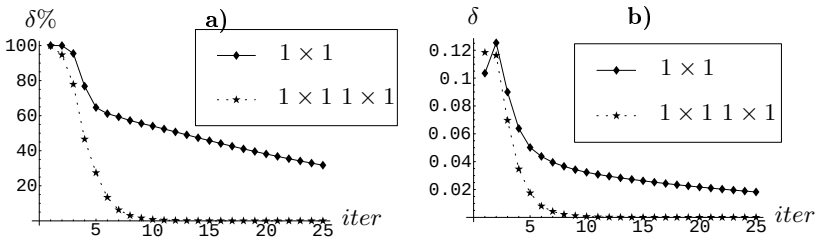


Fig. 11: Dependence of the maximum relative error (on the left side) and the mean absolute error (on the right side) of image reconstruction of $f_1(x, y)$ in the system $(1 \times 1,\ 1 \times 1)$ and in the system 1×1.

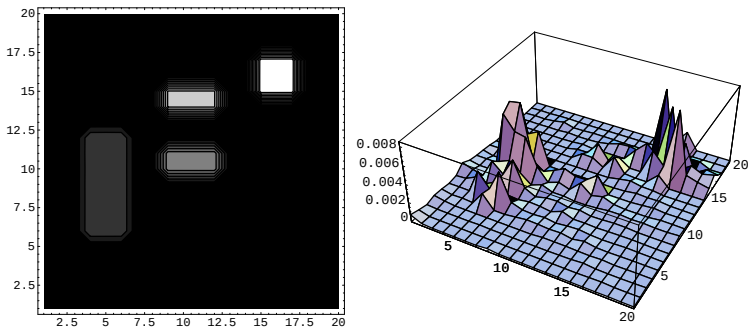


Fig. 12: The image reconstruction and the mean absolute error for $f_2(x, y)$ obtained with algorithm ART-3 for $n = 20 \times 20$, $m = 644$, $iter = 15$ in the system $(1 \times 1,\ 1 \times 1)$.

Note that the analogous table was obtained for algorithm MART-3, but in this case the velocity of convergence is considerably less.

For obtaining a good reconstruction in all experimental results it is very important, of course, the choice of the main parameters. From our investigation it follows that the best rate of convergence has the algorithm when the number of pixels (that is, the number of columns of an algebraic system of equations) is almost two times less than the number of all projection data (that is, the number of rows). The first parameter n, the number of pixels, was chosen in this paper in such a way that, in the first, the reconstructed function may be seen good enough on the picture, and, on the second, the obtained system of equations may be solved in the personal computer in a real time. The second main parameter m, the number of rays, was chosen in such a way that for some constant number of iteration $iter = 10$ (note, that for this number of iteration the quality of reconstruction was already obtained good enough) the reconstruction error was the least. For the system $(1 \times 1, 1 \times 1)$ the research for choosing m may be seen from the following table:

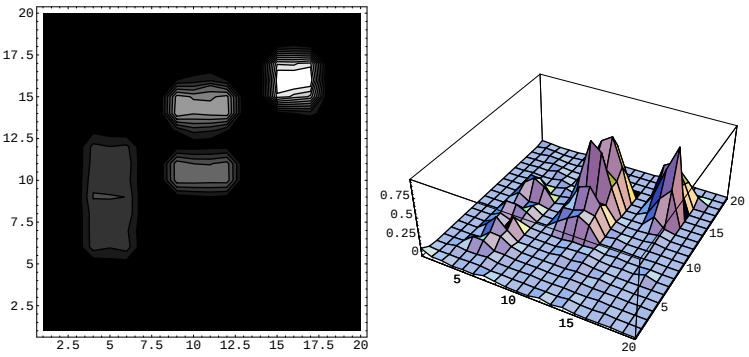


Fig. 13: The image reconstruction and the mean absolute error of $f_2(x, y)$ obtained with algorithm ART-3 for $n = 20 \times 20$, $m = 788$, $iter = 25$ in the system 1×1.

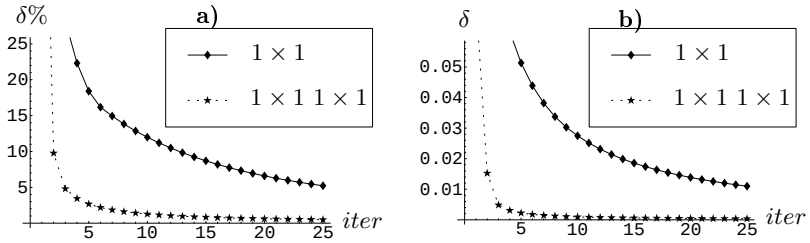


Fig. 14: Dependence of the maximum relative error (on the left side) and the mean absolute error (on the right side) of image reconstruction of $f_2(x, y)$ in the system $(1 \times 1,\ 1 \times 1)$ and in the system 1×1.

$m = 28$	$m = 124$	$m = 288$	$m = 388$	$m = 644$	
1.000	0.544	0.073	0.0215	0.0077	Δ
100.0	54.42	7.328	2.1538	0.7793	$\delta\%$
0.122	0.029	0.002	0.0007	0.0006	δ

All algorithms were implemented on IBM/PC (procesor AMD Duron XP, 1600 MHz) by means of C++ and MATHEMATICA 5.1. One iteration by means of Mathematica 5.1 was implemented approximately 0.5s for algorithm ART-3 and 2.5s for algorithm MART-3, and in C++ one iteration for both algorithms is implemented in a real time.

5 Conclusion

The aim of this paper was an elaboration and comparison of the iterative algebraic algorithms for reconstruction of high-contrast objects from incomplete projection data. We study the quality and convergence of these algorithms. The experimental results show that for each considered scheme of reconstruction

there exist the parameters which allow to obtain an enough good quality of reconstruction after the number of iteration which is considerably larger than for reconstruction with complete projection data. The convergent characteristics of algorithm ART-3 are considerably better by comparison with algorithm MART-3. And the configuration $(1 \times 1, 1 \times 1)$ is considerably better by comparison with the scheme 1×1. The number of iterations for achieving the stable reconstruction is approximately two times more for the second scheme by comparison with the first one. And this number is approximately 10 times more for the the scheme $(1 \times 1, 1 \times 1)$ by comparison with the case of the complete data. Moreover, if for the case of complete projection data there exists a number of iteration for which we have the best image reconstruction and after this number the error of reconstruction begins to increase, for the considered schemes of reconstruction with incomplete data the error of reconstruction reduces all time while it become stable and very small for enough large number (~ 1000).

References

1. Censor Y., De Piero A. R., Elfing T., Herman G. T., Iusem A. N.: *On iterative methods for linearly constrained entropy maximilization,* In: Numerical Analysis and Math. Modeling, A.Wakulicz (ed.), v. 24, 145–163, Banach Center Publications, Warsaw, Poland, 1990.
2. Censor Y., Zenios S. A., *Parallel Optimization. Theory, Algorithms, and Applications,* Oxford University Press, 1997.
3. Eggermont P. P. B., Herman G. T., Lent A.: *Iterative algorithms for large partitioned linear systems with applications to image reconstruction,* Linear Algebra and Its Appl. v. **40** (1981), 37–67.
4. Gubareni N., Computed Methods and Algorithms for Computer Tomography with limited number of projection data, Naukova Dumka, Kiev, 1997. (in Russian).
5. Herman G. T.: *Image Reconstruction from Projections,* Academic Press, New York (1980).
6. Herman G. T.: *A relaxation method for reconstructing objects from noisy x-rays,* Math. Programming v. **8** (1975) 1–19.
7. Herman G. T., Lent A., Rowland S.: *ART: Mathematics and application (a report on the mathematical foundations and on the applicability to real data of the Algebraic Reconstruction Techniques),* Journ. of Theoretica Biology v. **43** (1973), 1–32.
8. Lent A., *A convergent algorithm for maximum entropy image restoration, with a medical x-ray application.* In R.Shaw (ed), Image Analysis and Evaluation, 1977, p. 249–257. Society of Photographic Scientists and Engineers, Washington, USA.
9. Patella D.: *Introduction to ground surface self-potential tomography,* Geophysical Prospecting, vol. 45 (1997), 653–681.
10. Williams R. A., Atkinson K., Luke S. P., Barlow R. K., Dyer B. C., Smith J., Manning M.: *Applications for Tomographic Technology in Mining, Minerals and Food Engineering,* Particle and Particle Systems Characterization, vol. 12, N. 2 (2004) 105–111.

Proceedings of the International Multiconference on
Computer Science and Information Technology pp. 517–526
ISSN 1896-7094
© 2007 PIPS

Parallel PCG algorithms for voxel FEM elasticity systems

S. Margenov and Y. Vutov

Institute for Parallel Processing, Bulgarian Academy of Sciences
Acad. G. Bonchev, bl. 25A, 1113 Sofia, Bulgaria
margenov@parallel.bas.bg and *yavor@parallel.bas.bg*

Abstract. The presented comparative analysis concerns two parallel iterative solvers for large-scale linear systems related to μFEM simulation of human bones. The benchmark problems represent the strongly heterogeneous structure of real bone specimens. The voxel data are obtained by a high resolution computer tomography. Non-conforming Rannacher-Turek finite elements are used for discretization of the considered problem of linear elasticity. It is well known that the preconditioned conjugate gradient method is the best tool for efficient solution of large-scale symmetric systems with sparse, positive definite matrices. Here, the performance of two parallel preconditioners is studied. Both are based on displacement decomposition. The first one uses modified incomplete Cholesky factorization MIC(0) and the other – algebraic multigrid. The comparative analysis is mostly based on the computing times to run the codes. The number of iterations for both preconditioners are also discussed.

Keywords: FEM, PCG, DD, MIC(0), AMG, parallel algorithms.

1 Introduction

This study is motivated by the development and tuning of robust iterative solution methods, algorithms and software tools for μFE (micro finite element) simulation of human bones. A voxel representation of the bone structure based on micro computer tomography (CT) images is used to formulate the problem. The computational domain is a strongly heterogeneous composition of solid and fluid phases, see Fig. 2. The considered isotropic linear elasticity model is a current brick in the development of a toolkit for μFE simulation of the bone microstructure. The implementation of a poroealsticity model is the next step in this project.

Non-conforming Rannacher-Turek FEs are used for discretization of the problem. The obtained linear system is large, with a sparse, symmetric and positive definite matrix. This implies the use of iterative solvers based on the preconditioned conjugate gradient (PCG) method [7]. The elasticity stiffness matrix has a coupled block structure corresponding to a separable displacement ordering of

the unknowns. Here, the performance of the following two basic preconditioning codes, incorporated in a displacement decomposition framework, is studied. The first one is the modified incomplete factorization, MIC(0), and the second is the algebraic multigrid, AMG. The MIC(0) code is developed in IPP-BAS, Sofia, while the AMG one is the BoomerAMG module of the software system Hypre developed at LLNL, Livermore. The comparative analysis is focused on the number of iterations and the related computing times for real-life large-scale problems.

The presented results are based on some recent studies of the authors addressed to the case of scalar elliptic problems [18]. The earlier considered FEM elliptic solvers are used here in the construction of efficient preconditioners for the coupled elasticity system.

The paper is organized as follows. In Section 2 we describe the Finite Element Method (FEM) setting of the problem. Then, in Section 3 MIC(0) and BoomerAMG preconditioning algorithms are presented. Section 4 is devoted to numerical experiments. First, experiments with constant coefficients on three parallel computers are shown. The next set of experiments illustrates the behavior of preconditioners on heterogeneous (voxel) problems with strong coefficient jumps.

2 Non-conforming FEM formulation of the problem

We consider the weak formulation of the linear elasticity problem in the form [1]: find $\mathbf{u} \in [H^1_E(\Omega)]^3 = \{\mathbf{v} \in [H^1(\Omega)]^3 : \mathbf{v}_{\Gamma_D} = \mathbf{u}_S\}$ such that

$$\int_\Omega [2\mu\varepsilon(\mathbf{u}) : \varepsilon(\mathbf{v}) + \lambda \operatorname{div}\mathbf{u} \operatorname{div}\mathbf{v}]d\Omega = \int_\Omega \mathbf{f}^t\mathbf{v}d\Omega + \int_{\Gamma_N} \mathbf{g}^t\mathbf{v}d\Gamma, \qquad (1)$$

$\forall \mathbf{v} \in [H^1_0(\Omega)]^3 = \{\mathbf{v} = [H^1(\Omega)]^3 : \mathbf{v}_{\Gamma_D} = 0\}$, with the positive constants λ and μ of Lamé, the symmetric strains

$$\varepsilon(\mathbf{u}) := 0.5(\nabla\mathbf{u} + (\nabla\mathbf{u})^t),$$

the volume forces $\mathbf{f}$, and the boundary tractions $\mathbf{g}$, $\Gamma_N \cup \Gamma_D = \partial\Omega$, $|\Gamma_D| \neq \emptyset$. The Lamé coefficients are given by $\lambda = \dfrac{\nu E}{(1+\nu)(1-2\nu)}$, $\mu = \dfrac{E}{2(1+\nu)}$, where E stands for the modulus of elasticity, and $\nu \in (0, \frac{1}{2})$ is the Poisson ratio.

To obtain a stable saddle-point system one usually uses a mixed formulation for $\mathbf{u}$ and $\operatorname{div}\mathbf{u}$. By the choice of piece-wise constant finite elements for the dual variable, it can be eliminated at the macroelement level, and thereafter a symmetric positive definite FEM system in primal unknowns (displacement) is obtained. This approach is known as *reduced and selective integration* (RSI) technique, see [3]. For the discretization of (1) we use nonconforming rotated trilinear elements of Rannacher-Turek [8].

Let $\Omega^H = w_1^H \times w_2^H \times w_3^H$ be a regular coarser decomposition of the domain $\Omega \subset \mathbb{R}^3$ into hexahedrons, and let the finer decomposition $\Omega^h = w_1^h \times w_2^h \times w_3^h$

be obtained by a regular refinement of each macro element $E \in \Omega^H$ into eight similar hexahedrons. The cube $\hat{e} = [-1,1]^3$ is used as a reference element in the parametric definition of the rotated trilinear elements. For each $e \in \Omega^h$, let $\psi_e : \hat{e} \to e$ be the trilinear 1–1 transformation. Then the nodal basis functions are defined by the relations $\{\phi_i\}_{i=1}^6 = \{\hat{\phi}_i \circ \psi_e^{-1}\}_{i=1}^6$, where $\hat{\phi}_i \in span\{1, \xi_j, \xi_j^2 - \xi_{j+1}^2, j = 1,2,3\}$. One possible set of interpolation conditions are: $\hat{\phi}_i(b_{\Gamma^j}) = \delta_{ij}$, where δ_{ij} is the Kronecker symbol and b_{Γ^j} are the midpoints of the walls $\{\Gamma^j\}_{j=1}^6$ of $\hat{e}$. Then,

$$\begin{aligned}
\hat{\phi}_1(x,y,z) &= \left(1 - 3x + 2x^2 - y^2 - z^2\right)/6,\\
\hat{\phi}_2(x,y,z) &= \left(1 + 3x + 2x^2 - y^2 - z^2\right)/6,\\
\hat{\phi}_3(x,y,z) &= \left(1 - x^2 - 3y + 2y^2 - z^2\right)/6,\\
\hat{\phi}_4(x,y,z) &= \left(1 - x^2 + 3y + 2y^2 - z^2\right)/6,\\
\hat{\phi}_5(x,y,z) &= \left(1 - x^2 - y^2 - 3z + 2z^2\right)/6,\\
\hat{\phi}_6(x,y,z) &= \left(1 - x^2 - y^2 + 3z + 2z^2\right)/6.
\end{aligned}$$

The RSI FEM discretization reads as follows: find $\mathbf{u}^h \in V_E^h$ such that

$$\sum_{e \in \Omega^h} \int_e \left[2\mu\varepsilon^*(\mathbf{u}^h) : \varepsilon^*(\mathbf{v}^h) + \lambda \ \mathrm{div}\,\mathbf{u}^h \ \mathrm{div}\,\mathbf{v}^h\right] de = \int_\Omega \mathbf{f}^t\mathbf{v}^h d\Omega + \int_{\Gamma_N} \mathbf{g}^t\mathbf{v}^h d\Gamma, \tag{2}$$

$\forall \mathbf{v}^h \in V_0^h$, where $\varepsilon^*(\mathbf{u}) := \nabla\mathbf{u} - 0.5 I_L^{Q^H}[\nabla\mathbf{u} - (\nabla\mathbf{u})^t]$, V_0^h is the FEM space, satisfying (in nodalwise sense) homogeneous boundary conditions on Γ_D, the operator $I_L^{Q^H}$ denotes the L^2–orthogonal projection onto Q^H, the space of piecewise constant functions on the coarser decomposition Ω^H of Ω. Then the standard computational procedure leads to the coupled system of linear equations

$$\begin{bmatrix} K_{11} & K_{12} & K_{13} \\ K_{21} & K_{22} & K_{23} \\ K_{31} & K_{32} & K_{33} \end{bmatrix} \begin{bmatrix} \mathbf{u}_h^1 \\ \mathbf{u}_h^2 \\ \mathbf{u}_h^3 \end{bmatrix} = \begin{bmatrix} \mathbf{f}_h^1 \\ \mathbf{f}_h^2 \\ \mathbf{f}_h^3 \end{bmatrix}. \tag{3}$$

Here the stiffness matrix K is written in block form corresponding to a separate displacements components ordering of the vector of nodal unknowns. Since K is sparse, symmetric and positive definite, we use the PCG method to solve the system (3).

3 Preconditioning algorithms

The PCG is known to be the best algorithm for solution of large systems of linear equations with symmetric and positive definite sparse matrices [7]. Crucial for its performance is the preconditioning technique used. Here we focus on two preconditioners based on the isotropic variant of the displacement decomposition (DD)[4,5]. We write the DD auxiliary matrix in the form

$$C_{DD} = \begin{bmatrix} A & & \\ & A & \\ & & A \end{bmatrix} \tag{4}$$

where A is the stiffness matrix corresponding to the bilinear form

$$a(u^h, v^h) = \sum_{e \in \Omega^h} \int_e E \left(\sum_{i=1}^{3} \frac{\partial u^h}{\partial x_i} \frac{\partial v^h}{\partial x_i} \right) de. \tag{5}$$

Such approach is motivated by the second Korn's inequality, which holds for the RSI FEM discretization (2) under consideration. This means that the estimate

$$\kappa(C_{DD}^{-1} K) = O((1 - 2\nu)^{-1})$$

holds uniformly with respect to the mesh size parameter in the FEM discretization [6]. The first of the studied preconditioners is obtained by MIC(0) factorization of the blocks in (4). As an alternative, inner PCG iterations with BoomerAMG for A are used to approximate the DD block-diagonal matrix (4).

3.1 Parallel MIC(0) preconditioning

First, we give a brief introduction to the modified incomplete factorization [9], see also [11]. Our presentation at this point follows those from [4]. Let us rewrite the real $N \times N$ matrix $A = (a_{ij})$ in the form

$$A = D - L - L^T$$

where D is the diagonal and $(-L)$ is the strictly lower triangular part of A. Then we consider the approximate factorization of A which has the form:

$$C_{MIC(0)} = (X - L) X^{-1} (X - L)^T$$

with $X = diag(x_1, \cdots, x_N)$ being the diagonal matrix determined by the condition of equal rowsums. We are interested in the case when $X > 0$ and thus $C_{MIC(0)}$ is positive definite for the purpose of preconditioning. If this holds, we speak about *stable* MIC(0) factorization. Concerning the stability of MIC(0), the following theorem holds [10,4].

Theorem 1. *Let $A = (a_{ij})$ be a symmetric real $N \times N$ matrix and let $A = D - L - L^T$ be the splitting of A. Let us assume that*

$$\begin{aligned} L &\geq 0, \\ A\mathbf{e} &\geq 0, \\ A\mathbf{e} + L^T\mathbf{e} &> 0, \qquad \mathbf{e} = (1, \cdots, 1)^T \in R^N, \end{aligned}$$

i.e. that A is a weakly diagonally dominant with nonpositive offdiagonal entries and that $A + L^T = D - L$ is strictly diagonally dominant. Then the relation

$$x_i = a_{ii} - \sum_{k=1}^{i-1} \frac{a_{ik}}{x_k} \sum_{j=k+1}^{N} a_{kj} > 0$$

holds and the diagonal matrix $X = diag(x_1, \cdots, x_N)$ defines stable MIC(0) factorization of A.

Remark 1. The presented numerical tests are performed using the perturbed version of MIC(0) algorithm, where the incomplete factorization is applied to the matrix $\tilde{A} = A + \tilde{D}$. The diagonal perturbation $\tilde{D} = \tilde{D}(\xi) = diag(\tilde{d}_1, \dots \tilde{d}_N)$ is defined as follows:

$$\tilde{d}_i = \begin{cases} \xi a_{ii} & \text{if} \quad a_{ii} \geq 2w_i \\ \xi^{1/2} a_{ii} & \text{if} \quad a_{ii} < 2w_i \end{cases}$$

where $0 < \xi < 1$ is a properly chosen parameter, and $w_i = \sum_{j>i} -a_{ij}$.

The idea of our parallel algorithm is to apply the MIC(0) factorization on an auxiliary matrix B, which approximates A. The matrix B has a special block structure, which allows a scalable parallel implementation.

Following the standard FEM assembling procedure we write A in the form $A = \sum_{e \in \omega_h} L_e^T A_e L_e$, where A_e is the element stiffness matrix, L_e stands for the restriction mapping of the global vector of unknowns to the local one corresponding to the current element e. Let us consider the following approximation B_e of A_e:

$$A_e = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} & a_{15} & a_{16} \\ a_{21} & a_{22} & a_{23} & a_{24} & a_{25} & a_{26} \\ a_{31} & a_{32} & a_{33} & a_{34} & a_{35} & a_{36} \\ a_{41} & a_{42} & a_{43} & a_{44} & a_{45} & a_{46} \\ a_{51} & a_{52} & a_{53} & a_{54} & a_{55} & a_{56} \\ a_{61} & a_{62} & a_{63} & a_{64} & a_{65} & a_{66} \end{bmatrix}, \quad B_e = \begin{bmatrix} b_{11} & a_{12} & a_{13} & a_{14} & a_{15} & a_{16} \\ a_{21} & b_{22} & a_{23} & a_{24} & a_{25} & a_{26} \\ a_{31} & a_{32} & b_{33} & 0 & 0 & 0 \\ a_{41} & a_{42} & 0 & b_{44} & 0 & 0 \\ a_{51} & a_{52} & 0 & 0 & b_{55} & 0 \\ a_{61} & a_{62} & 0 & 0 & 0 & b_{66} \end{bmatrix}.$$

The local numbering follows the pairs of the opposite nodes of the reference element. The diagonal entries of B_e are modified to hold the rowsum criteria. Assembling the locally defined matrices B_e we get the global matrix $B = \sum_{e \in \omega_h} L_e^T B_e L_e$. The condition number estimate $\kappa(B^{-1}A) \leq 3$ holds uniformly with respect to mesh parameter and possible coefficient jumps (see for the related analysis in [15,16]). This particular choice of B_e gives an important property of the matrix B – its diagonal blocks (corresponding to (x, y) cross sections) are diagonal matrices. This allows a scalable parallel implementation, which for the case of scalar elliptic problems is studied in [17,18]. The sparsity structure of the matrices A and B is illustrated by Fig. 1. Lexicographic node numbering is used.

This paper concerns with scalability of the parallel DD MIC(0) preconditioner

$$C_{DDMIC(0)} = \begin{bmatrix} C_{MIC(0)}(B) & & \\ & C_{MIC(0)}(B) & \\ & & C_{MIC(0)}(B) \end{bmatrix}.$$

3.2 BoomerAMG

BoomerAMG contains sequential and parallel implementations of algebraic multigrid methods [12]. It can be used as a solver or as a preconditioner. Various different parallel coarsening techniques and relaxation schemes are available.

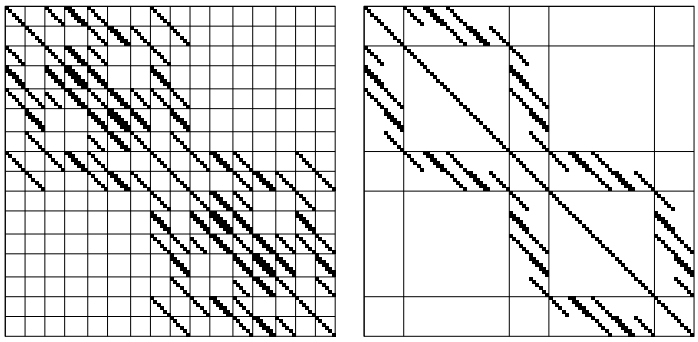


Fig. 1. Sparsity structure of the matrix A on the left and matrix B on the right, for the division of Ω into 2x2x6 hexahedrons. Non-zero elements are drawn with small squares.

See [13,14] for a detailed description of the coarsening algorithms, the interpolation and numerical results.

Version 2.0.0 of the Hypre library was used for the performed tests.

The following coarsening techniques are available:

- the Cleary-Luby-Jones-Plassman (CLJP) coarsening,
- various variants of the classical Ruge-Stüben (RS) coarsening algorithm, and
- the Falgout coarsening which is a combination of CLJP and the classical RS coarsening algorithm.

The following relaxation techniques are available:

- Jacobi relaxation,
- hybrid Gauss-Seidel / Jacobi relaxation scheme,
- symmetric hybrid Gauss-Seidel / Jacobi relaxation scheme, and
- Gauss-Seidel relaxation.

The Falgout coarsening was used in the presented tests. A V(1,1)-cycle with hybrid Gauss-Seidel smoothing is performed. The related AMG strength threshold is 0.5. The default coarsening with one inner iteration was set at the beginning. This variant proved to be very expensive, especially with respect to the consumed memory. Then we switched on the option of aggressive coarsening on the first two levels. As a result the memory consumption noticeably dropped. To get a reasonable convergence of the outer PCG iteration, the number of inner iterations were to be increased. Table 1 justifies our decision to use the aggressive coarsening in the further numerical tests. The following data are collected in this table: mesh size parameter n; number of processors p; number of AC levels (levels with aggressive coarsening where 0 stands for no aggressive coarsening); number of outer iterations It_{out}; related inner iterations It_{inn}; computation time

Table 1. BoomerAMG

n	p	AC levels	It_{out}	It_{inn}	$T[s]$	$M[MB]$
32	1	0	12	1	15.6	108
32	1	2	9	4	10.4	66
64	1	0	13	1	185.3	899
64	1	2	9	4	107.2	432
64	4	0	24	1	180.1	1225
64	4	2	10	4	58.5	664

T in seconds; required total memory M in megabytes. A more detailed description of the model test problem is given at the beginning of the next section. As one can see, despite the increased number of inner iterations, the computational times are strongly decreased and the parallel efficiency is improved if aggressive coarsening is used.

4 Comparative numerical tests

4.1 Scalability tests

Numerical tests with the considered two parallel algorithms and codes are present and analyzed in this section. The tests are run on three parallel platforms, referred to further as C1, C2 and C3. Platform C1 is an "IBM SP Cluster 1600" consisting of 64 p5-575 nodes interconnected with a pair of connections to the Federation HPS (High Performance Switch). Each p5-575 node contains 8 Power5 SMP processors at 1.9GHz and 16GB of RAM. The network bandwidth is 16Gb/s. Platform C2 is an IBM Linux Cluster 1350, made of 512 dual-core IBM X335 nodes. Each node contains 2 Xeon Pentium IV processors and 2GB of RAM. Nodes are interconnected with a 1Gb Myrinet network. Platform C3 is a "Cray XD1" cabinet, fully equipped with 72 2-way nodes, totaling in 144 AMD Opteron processors at 2.4GHz. Each node has 4GB of memory. The CPUs are interconnected with the Cray RaidArray network with a bandwidth of 5.6Gb/s.

The computational domain is the cube $[0,1]^3$, where homogeneous Dirichlet boundary conditions are assumed at the bottom. The force $||g|| = 1$ is acting on the top. The mesh is uniform. Here n stands for the number of subintervals in the fine grid of the RSI FEM discretization in each direction. The mechanical characteristics of the model problem are $E = 1$ and $\nu = 0.3$. The size of the resulting nonconforming FEM system is $N = 9n^2(n+1)$. The number of processors p is increased proportionally with the problem size N. The stopping criterion in all considered tests is $(C^{-1}r^{N_{it}}, r^{N_{it}})/(C^{-1}r^0, r^0) < 10^{-6}$, where r^i is the current residual and C stands for the used preconditioner. Table 2 presents the time T in seconds, the number of iterations It (the outer ones for the AMG code), varying the preconditioners, the problem sizes and the platforms.

In a good agreement with the theory, the number of iterations for MIC(0) increases as $O(\sqrt{n})$, while the AMG iterations stay about the same. For the

Table 2. Parallel Tests I

			C1				C2				C3			
			MIC(0)		AMG		MIC(0)		AMG		MIC(0)		AMG	
n	N	p	$T[s]$	It	$T[s]$	It	$T[s]$	It	$T[s]$	It	$T[s]$	It	$T[s]$	It
64	2 396 160	1	136.6	115	150.1	9	83.7	115	84.0	9	83.9	115	115.1	9
128	19 021 824	8	202.0	163	195.6	10	172.1	163	229.8	10	127.8	163	152.6	10
256	151 584 768	64	355.6	230	261.4	10	464.1	230	430.0	10	328.2	230	307.1	10

smallest problem (N=2 396 160) MIC(0) clearly outperforms the AMG code. For the medium size (N= 19 021 824) the times are rather similar. However, for the largest problem (N=151 584 768) the advantage of AMG is well expressed.

4.2 Voxel analysis tests

The bone microstructure is a typical example of strongly heterogeneous media. In the presented tests, the computational domain is a composition of solid and fluid phases. The CT image is extracted from the dataset [2]. The voxel size is 37μm. Each voxel corresponds to a macroelement from the RSI FEM discretization. The bone specimen is placed between two plates (see Fig. 2). The thickness of the plates is 1 voxel. The position of the bottom plate is fixed (homogeneous Dirichlet boundary conditions), and a force of $||g|| = 1$ is uniformly distributed on the top one. This setting simulates a vertically loaded bone specimen. The density of the solid phase is an important characteristics of the bone. The density of the considered three specimens is respectively 25%, 19%, 18% for the cases $32 \times 32 \times 32$, $64 \times 64 \times 64$, $128 \times 128 \times 128$.

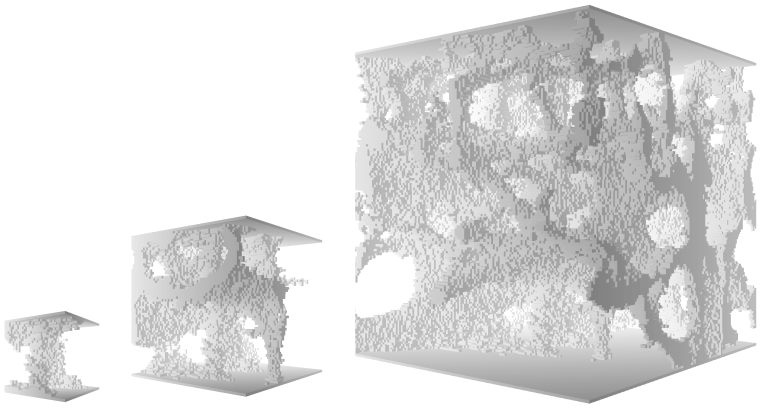


Fig. 2. Structure of the solid phase: $32 \times 32 \times 32$ - left, $64 \times 64 \times 64$ - middle, and $128 \times 128 \times 128$ - right.

The considered test problems are given by the following parameters: $E_p = 10$, $E_s = 1$, $E_f = \zeta \in \{0.1, 0.01, 0.001\}$, $\nu = 0.3$. Here, E_p is the elasticity modulus of the two plates, E_s stands for a scaled elasticity modulus of the solid phase, while E_f introduces varying coefficient jumps between solid and fluid phases.

The results presented in Table 3 are obtained on the platform C2. For the case of the biggest coefficient jumps (ζ=0.001) and the biggest problem (N=151 584 768), outer PCG iteration with AMG preconditioner fails to converge within the specified time limit of 7200 seconds. This test was repeated with an increased number of inner iterations. The corresponding values in the table are obtained with $It_{in} = 6$.

Table 3. Parallel Tests II

		$\zeta = 0.1$				$\zeta = 0.01$				$\zeta = 0.001$			
		MIC(0)		AMG		MIC(0)		AMG		MIC(0)		AMG	
n	p	$T[s]$	It	$T[s]$	It	$T[s]$	It	$T[s]$	It	$T[s]$	It	$T[s]$	It
64	1	239.3	330	374.9	27	348.3	505	757.9	57	588.6	823	1040.5	78
128	8	833.2	708	681.0	25	975.5	830	1501.3	60	2166.7	1850	2908.9	107
256	64	2393.8	1237	945.4	25	3495.7	1831	2114.4	57	6025.8	3150	5520.1	114

What we see is that the number of iterations for the MIC(0) code increase more drastically with the problem size, than the case without jumps. AMG preconditioner manages to sustain the number of iterations for different problem sizes and fixed ζ, except for the case $\zeta = 0.001$, where slight increase is observed. For the smallest problem (N=2 396 160) MIC(0) outperforms the AMG code again for all variations of ζ. For the medium sized problem (N=19 021 824) MIC(0) code is faster for the cases of strong coefficient jumps ($\zeta = 0.01$ and $\zeta = 0.001$). For the largest problem (N=151 584 768) AMG is faster, but its advantage decreases with the rise of the coefficient jump. The reason for this behavior is that the AMG iterations are much more expensive than the MIC(0) ones. So, the increased number of iterations has much heavier impact on the computational time for the AMG code.

The general conclusion is that the studied codes provide a stable toolkit for computer simulation of the bone microstructure. Both approaches have their advantages depending on the size of the FEM systems and the level of heterogeneity of the bone specimens. The achieved parallel scalability well corresponds to the connectivity of the considered problems.

Acknowledgments. The partial support through EC INCO Grant BIS-21++ 016639/2005 and Bulgarian NSF Grant VU-MI-202/2006 is gratefully acknowledged. The parallel numerical tests are supported via EC Project HPC-EUROPA RII3-CT-2003-506079.

References

1. S. Margenov, *Displacement doecomposition-MIC(0) preconditioning of linear elasticity non-conforming FEM problems*, 16th IMACS World Congress 2000, Lausanne, Proceedings, (2000), 107-4.
2. Gisela Beller, Markus Burkhart, Dieter Felsenberg, Wolfgang Gowin, Hans-Christian Hege, Bruno Koller, Steffen Prohaska, Peter I. Saparin and Jesper S. Thomsen: Vertebral Body Data Set ESA29-99-L3, `http://bone3d.zib.de/data/2005/ESA29-99-L3/`
3. D. Malkus, T. Hughes. Mixed finite element methods. Reduced and selective integration techniques: an uniform concepts. *Comp. Meth. Appl. Mech. Eng.,*15: 63-81, 1978.
4. R. Blaheta: *Displacement Decomposition—incomplete factorization preconditioning techniques for linear elasticity problems*, Numer. Lin. Alg. Appl., **1** (1994), 107–126
5. O. Axelsson, I. Gustafsson. Iterative methods for the Navier equation of elasticity. *Comp. Meth. Appl. Mech. Engin.,* 15: 241-258, 1978.
6. Axelsson, O., On iterative solvers in structural mechanics; separate displacement orderings and mixed variable methods, Math. Comput. Simulation 50 (1999), no. 1–4, 11–30.
7. O. Axelsson: *Iterative Solution Methods.* Cambridge University Press, Cambridge, 1994.
8. R. Rannacher, S. Turek: *Simple nonconforming quadrilateral Stokes Element*, Numer. Methods Partial Differential Equations 8(2), 1992, 97–112.
9. I. Gustafsson: *Modified incomplete Cholesky (MIC) factorization*, In: D. J. Evans (ed.): Preconditioning Methods; Theory and Applications, Gordon and Breach, 1983, 265–293.
10. I. Gustafsson: An incomplete factorization preconditioning method based on modification of element matrices. BIT 35: 1, 86–100, 1996.
11. I. Gustafsson: *Stability and rate of convergence of modified incomplete Cholesky factorization methods*, Chalmers University of Technology, Report Nr. 79.02R, 1979.
12. J. W. Ruge and K. Stüben. *Algebraic multigrid (AMG).* In S. F. McCormick, editor, Multigrid Methods, volume 3 of Frontiers in Applied Mathematics, SIAM, Philadelphia, PA, 1987, 73–130.
13. V. E. Henson and U. M. Yang. *BoomerAMG: a parallel algebraic multigrid solver and preconditioner.* Applied Numerical Mathematics, 41(5), 2002, 155–177. Also available as LLNL technical report UCRL-JC-141495.
14. U. M. Yang. *Parallel algebraic multigrid methods—high performance preconditioners.* In A. M. Bruaset and A. Tveito, editors, Numerical Solution of Partial Differential Equations on Parallel Computers, Springer-Verlag, 2005, 209–236. Also available as LLNL technical report UCRL-BOOK-208032.
15. P. Arbenz, S. Margenov: *Parallel MIC(0) preconditioning of 3D nonconforming FEM systems*, Iterative Methods, Preconditioning and Numerical PDEs, Proceedings of IMET Conference, Prague, 2004, 12–15.
16. P. Arbenz, S. Margenov, Y. Vutov: *Parallel MIC(0) Preconditioning of 3D Elliptic Problems Discretized by Rannacher-Turek Finite Elements*, Comput. Math. Appl. (to appear)
17. Y. Vutov: *Parallel Incomplete Factorization of 3D NC FEM Elliptic Systems*, Numerical Methods and Applications, Springer LNCS 4210, 2007, 114–121.
18. S. Margenov, Y. Vutov: *Preconditioning of Voxel FEM Elliptic Systems*, TASK, Quaterly 11, No 1–2, 2007, 117–128.

Proceedings of the International Multiconference on
Computer Science and Information Technology pp. 527–540
ISSN 1896-7094
© 2007 PIPS

Implementing the Conjugate Gradient Method on a grid computer

Tijmen Collignon and Martin B. van Gijzen

Delft University of Technology,
Faculty of Electrical Engineering, Mathematics and Computer Science,
Mekelweg 4, 2628 CD Delft, the Netherlands,
{t.p.collignon,m.b.vangijzen}@tudelft.nl,
Website: http://ta.twi.tudelft.nl/nw/users/collignon/

Abstract. We study two implementations of the Conjugate Gradient method for solving large sparse linear systems of equations on a heterogeneous computing grid, using GridSolve as grid middleware. We consider the standard CG algorithm of Hestenes and Stiefel, and as an alternative the Chronopoulos/Gear variant, a formulation that is potentially better suited for grid computing since it requires only one synchronisation point per iteration, instead of two for standard CG. The computational work is divided into tasks which are dynamically distributed over the available resources using a resource–aware data partitioning strategy. We present numerical experiments that show lower computing times and better speed–up for the Chronopoulos/Gear variant. We also identify bottlenecks and suggest improvements to GridSolve.

1 Introduction

The solution of sparse linear systems is the computational bottleneck for many large scale numerical simulations. In order to solve these systems, which may consist of millions of equations, the combined computing power of many processors is needed. Dedicated parallel hardware, however, is expensive.

A natural idea to provide cheap parallel computing power is to use the available non–dedicated hardware, and thus to make better use of the existing resources. This idea has given rise to the concepts of grid computing and of computational grids, see for example [1]. In grid computing a pool of computational tasks is dynamically distributed over a computational grid, which can be a local cluster of computers, but it can also be a group of computers at geographically different locations that are connected via the Internet. This approach has proven to be successful for embarrassingly parallel applications where the tasks do not require interprocessor communication, as exemplified by the well–known SETI@home project [2].

For the numerical solution of linear systems of equations, however, inter–task communication is unavoidable. For this application, developing efficient parallel numerical algorithms for dedicated homogeneous systems is a difficult problem, but becomes even more challenging when applied to heterogeneous systems. In

particular, the heterogeneity of the computational nodes and the variability in network performance offer new algorithmic problems.

In this paper we study different implementations of the Conjugate Gradient (CG) method [3] on a computational grid. We use the GridSolve library [4] as grid middleware. Load balancing is achieved using a simple resource–aware data partitioning strategy. The number of synchronisation points in the CG algorithm which is in its standard implementation equal to two, can be reduced to one by using the implementation that has been proposed by Chronopoulos/Gear [5]. Our numerical experiments show that by minimising the number of synchronisation points and by careful load balancing a significant speed–up can also be achieved for the solution of systems of equations, despite the fact that for this application the tasks are tightly coupled.

The remainder of the paper is organised as follows. In the next section we describe in detail our architecture–aware Conjugate Gradient algorithm and related issues, which includes a description of GridSolve, data management strategies, and specific implementation details. Section 3 contains some experimental results and in Sect. 4 we give concluding remarks and some suggestions for future work.

2 Heterogeneous sparse linear solvers in GridSolve

2.1 Motivation

This work is part of a larger project where we want to apply the Immersed Boundary Method [6] to simulate general fluid–structure interaction problems using grid computers. Examples of said problems are the swimming of fish or the airflow around wind turbine rotor blades. These simulations involve numerically solving the governing fluid equations on a structured grid, where the most expensive part usually consists of solving a large sparse linear system $Ax = b$ at each time step. Such a system typically emerges from a finite difference discretisation of the Poisson equation with varying coefficients and Neumann boundary conditions,

$$\begin{cases} -\nabla \cdot \left(\frac{1}{\rho(\mathbf{x})} \nabla p(\mathbf{x}) \right) = f(\mathbf{x}) & \mathbf{x} \in \Omega, \\ \frac{\partial}{\partial \mathbf{x}} p(\mathbf{x}) = g(\mathbf{x}) & \mathbf{x} \in \partial\Omega. \end{cases} \tag{1}$$

Here, p and ρ represent the pressure and density respectively. For the purpose of this paper we confine ourselves to a two–dimensional square domain where p is zero on the boundary. Nevertheless, this problem still retains much of the characteristics of the original problem. The equation is discretised on a $k \times k$ grid using second order finite differences, resulting in the discrete Poisson equation,

$$\begin{aligned} \rho_{i+\frac{1}{2},j} p_{i+1,j} + \rho_{i,j+\frac{1}{2}} p_{i,j+1} - \left(\rho_{i+\frac{1}{2},j} + \rho_{i,j+\frac{1}{2}} + \rho_{i-\frac{1}{2},j} + \rho_{i,j-\frac{1}{2}} \right) p_{ij} + \\ \rho_{i-\frac{1}{2},j} p_{i-1,j} + \rho_{i,j-\frac{1}{2}} p_{i,j-1} = h^2 f_{ij}, \quad \text{for } 0 \le i,j < k. \end{aligned} \tag{2}$$

Using the lexicographic ordering of the unknowns we can set

$$x_{i+jk} := p_{ij}, \quad \text{for } 0 \leq i, j < k, \tag{3}$$

with $n = k^2$ as the length of x. The right–hand side vector b is defined similarly. Note that we only number the internal points. The coefficient matrix A can be written as

$$A = \begin{bmatrix} Q^{(0)} & R^{(0)} & & & \\ R^{(0)} & Q^{(1)} & R^{(1)} & & \\ & \ddots & \ddots & \ddots & \\ & & R^{(k-3)} & Q^{(k-2)} & R^{(k-2)} \\ & & & R^{(k-2)} & Q^{(k-1)} \end{bmatrix}, \tag{4}$$

where the block matrices Q are R are defined by

$$q_{st}^{(j)} = \begin{cases} \mu_{sj} & \text{if } s = t \\ \rho_{s-\frac{1}{2},j} & \text{if } s - t = 1 \\ \rho_{s+\frac{1}{2},j} & \text{if } t - s = 1 \\ 0 & \text{otherwise} \end{cases}, \qquad \text{for } 0 \leq j < k, \tag{5}$$

with $\mu_{ij} = -\left(\rho_{i+\frac{1}{2},j} + \rho_{i,j+\frac{1}{2}} + \rho_{i-\frac{1}{2},j} + \rho_{i,j-\frac{1}{2}}\right)$ and

$$r_{st}^{(j)} = \begin{cases} \rho_{s,j+\frac{1}{2}} & \text{if } s = t \\ 0 & \text{otherwise} \end{cases}, \qquad \text{for } 0 \leq j < k - 1. \tag{6}$$

Summarizing, we need to solve the sparse linear system

$$Ax = b, \tag{7}$$

where A is an $n \times n$ block tridiagonal symmetric positive definite sparse matrix. The reason that we chose to solve this system using a preconditioned Conjugate Gradient method is threefold: (1) the CG method is simple to implement, (2) it is the obvious choice for this system, and (3) it is relatively straightforward to parallelize.

2.2 Brief overview of GridSolve

GridSolve (GS) is a distributed programming system which uses a client–server model for solving complex problems remotely on global networks [4, 7]. It is an instantiation of the GridRPC model, an emerging standard for a Remote Procedure Call (RPC) mechanism on Grid computers [8]. The GridRPC Application Programming Interface (API) is defined within the Global Grid Forum [9]. Other projects that use the GridRPC API are DIET [10], NetSolve [11], Ninf–G [12], and OmniRPC [13].

Software environments such as GridSolve are often called Network Enabled Servers (NES). These systems typically consist of six components: clients, agents, servers, databases, monitors, and schedulers. We will elaborate on the specific details of these components in the context of the current version (0.16.0) of GS (see Fig. 1). The GS servers (component 3) are software components that are started on each computational node which may consist of a single CPU or a cluster. The server monitors the workload of the node and keeps an updated list of the services (or *tasks*) that are installed on the server. For example, a task can be a single `dgemm` or a parallel MPI job. Services can be added or modified without restarting the server.

A single GridSolve agent (component 2) actively monitors the server properties such as CPU speed, memory size, computational services, and availability. These properties are stored in a database on the agent node and are periodically updated. When a GridSolve client program (component 1) written in either C, Fortran, or Matlab uses the GridRPC API to initiate a GS call to a remote problem, the GS middleware first contacts the agent. Based on the problem complexity, size of the input parameters, and the available computational resources, the agent then returns a list of servers sorted by minimum completion time. The client resorts the list after performing a quick network performance test. Input parameters are sent to the first server on the list and the task, which can be either blocking or non–blocking, is executed on the server. The result (if any) is then sent back to the client. If a task should fail it is transparently resubmitted to the next server on the list.

To determine the completion time of a particular task on server s, the total flop count of the problem is divided by the *effective speed* of the server. The latter is calculated using

$$\frac{s_{\text{flops}} \times s_{\text{ncpu}}}{\frac{s_{\text{workload}}}{100} + 1.0}, \tag{8}$$

where s_{flops} is the speed of server s in flops determined by multiplication of two dense matrices of fixed size, s_{ncpu} is the number of CPUs in the node, and $s_{\text{workload}} \in [0, 100]$ denotes the periodically updated workload.

The main advantages of GridSolve are that it is easy to use, install, and maintain. It allows for easy access to advanced remote computational resources. Furthermore, fault tolerance is supported through a simple but effective mechanism. Nevertheless, the current implementation has several obvious limitations. For example, the remote servers cannot communicate directly, which imposes a severe constraint on the type of applications that can be efficiently solved using the current implementation of GridSolve. It is therefore naturally suited for coarse–grained applications such as parametric studies and 'embarrassingly parallel' problems. In contrast, traditional parallel iterative solvers are inherently fine–grained and much research needs to be done before iterative solvers can be efficiently applied in Grid computing.

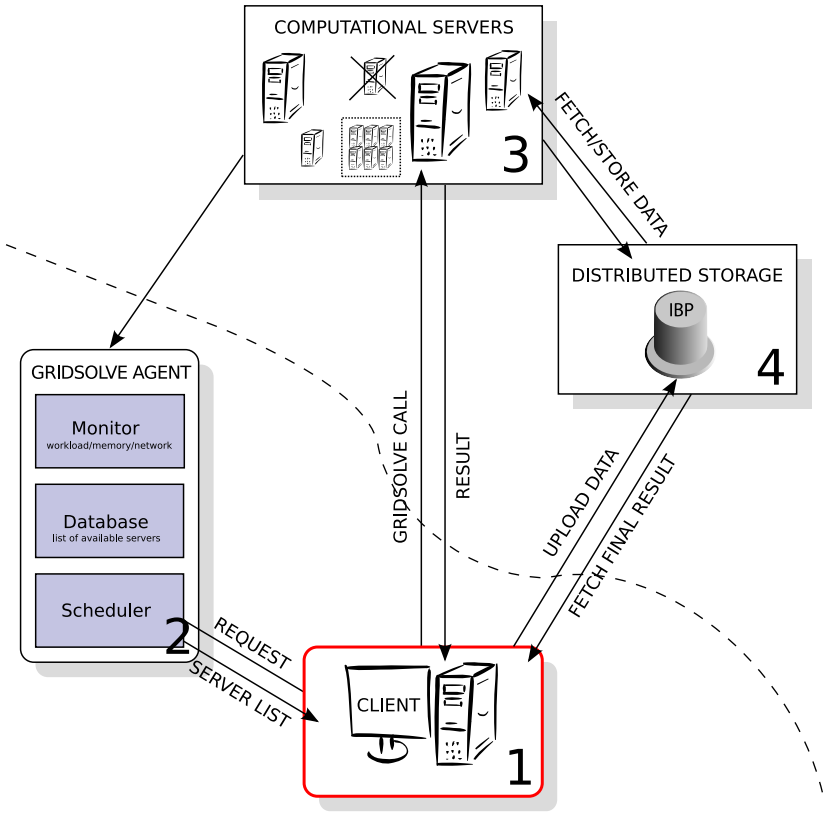


Fig. 1. Schematic overview of GridSolve. The dashed line symbolizes distance between client and servers.

2.3 Data management

In the current GridSolve model, separate tasks communicate data through the client, resulting in bridge communication. As a result, input and output data associated with a task is continuously being sent back and forth between the client and the server using a possible slow network connection. Also, any data that is read or generated locally during the execution of a task is lost after it finishes. Several strategies such as data persistence and data redistribution have been proposed to tackle these deficiencies for different implementations of the GridRPC API [14–18].

In GridSolve there is a partial solution to the first problem called the Distributed Storage Infrastructure (DSI). At the Logistical Computing and Internetworking (LoCI) Laboratory of the University of Tennessee the IBP (Internet Backplane Protocol) middleware has been developed based on this approach [19]. To avoid multiple transmissions of the same data between the client and the

server, the client can upload data to an IBP data depot which is in close proximity to the computational servers. Subsequently a data handle is sent to the server and the task can fetch and update the data on the IBP depot (see component (4) in Fig. 1). Using the DSI can be considered as programming for a shared memory model.

An approach similar to [15] in which the RPC model of NetSolve is extended to include communication between remote servers is currently being developed for GridSolve [20]. In the future we hope to use this extension and apply it to our problem.

2.4 Resource–aware load balancing

Algorithm 1 Resource–aware Preconditioned Conjugate Gradient Algorithm; p servers

1: Agent partitions work based on available computational resources.
2: Client sets initial values and uploads initial vectors such as b to the IBP data depot; Set $k = 0$.
3: **while** CG not converged and $k < k_{\max}$ **do**
4: Client assigns CG tasks to p servers and waits until tasks have completed.
5: Client repartitions work if significant change in workload and/or computational resources has occurred.
6: Set $k = k + 1$
7: **end while**
8: Client reads final answer from IBP depot.

We are interested in solving large sparse linear systems $Ax = b$ using GridSolve with architecture–aware dynamic load balancing. For this purpose it is insufficient to let the agent return a sorted list based on problem complexity and available resources, as is normally being done in GridSolve. Instead, we need to use a slightly different approach. Suppose that the client wishes to use p servers to solve a linear system. The scheduler in the GridSolve agent has been enhanced so that it creates a simple (non–homogeneous) partitioning of the computational work over p servers using information about currently available resources. It then returns the partitioning and a list of said servers to the client, after which the client initiates a series of non–blocking calls *explicitly specifying* the size and location of each task. Thus we ensure that the computational task is being performed on the intended server, in accordance with our partitioning. Unfortunately the fault–tolerance mechanism within the original GridSolve is now being circumvented, because the tasks cannot be resubmitted to another server should a task fail.

Algorithm 1 shows the general resource–aware CG algorithm. Note that the tasks use DSI file handles to manipulate the vectors on the IBP depot. The specific structure of the CG tasks will be discussed in the next section. After

each timestep of the iteration the client may decide to repartition the work and assign the work to different computational servers in the network accordingly.

2.5 Partitioning algorithm and CG schemes

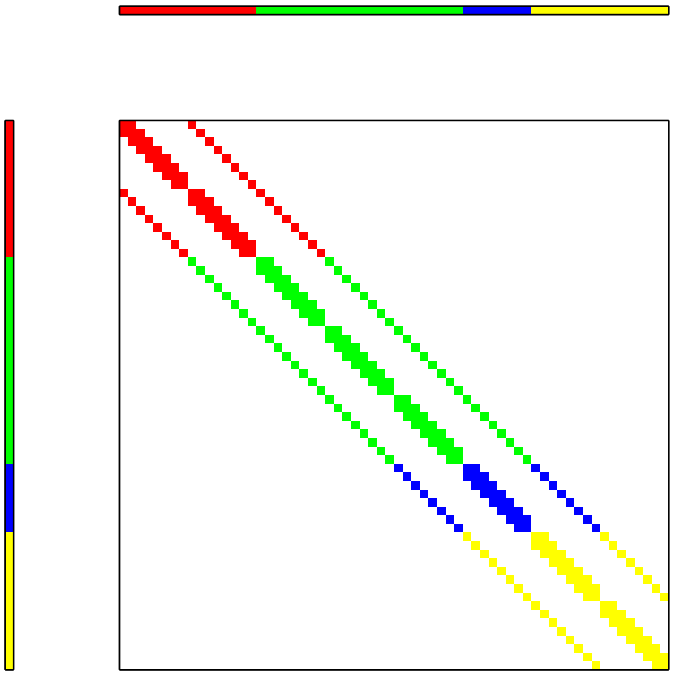


Fig. 2. Heterogeneous block–row partitioning $[2, 3, 1, 2]$ for $p = 4$ and $k = 8$.

As was shown earlier, the matrix A originating from our discretisation of the system (1) is a block tridiagonal matrix. Since each block–row roughly contains the same number of non–zeros, we chose a simple non–homogeneous block–row partitioning. An example of such a partitioning is illustrated in Fig.2 for $p = 4$ and $k = 8$. The input and output vectors are distributed in the same manner. More specifically, the effective speed of p servers is calculated using (8), together with the total flop count of a single CG iteration step. The size of each task is then determined accordingly.

The standard Conjugate Gradient method was implemented first and is shown in Algorithm 2. Jacobi preconditioning is used and there are two natural synchronisation barriers, namely the two inner products for computing α and ρ. When implementing this scheme in GridSolve, an extra synchronisation point is introduced due to technical implementation details. By simply rearranging certain terms this can be avoided.

Algorithm 2 Standard CG with Jacobi preconditioning; Task i with three subtasks.

Require: File handles to vectors $\mathbf{x}, \mathbf{r}, \mathbf{z}, \mathbf{p}, \mathbf{q}$ on IBP data depot and parameter k.
Ensure: $K := \mathrm{diag}(A)$
1: *// Each server i does the following.*
2: Read $\mathbf{r}^i$
3: Solve $\mathbf{z}^i$ from $K\mathbf{z}^i = \mathbf{r}^i$
4: Compute $\rho^i = (\mathbf{r}^i, \mathbf{z}^i)$
5: Update $\mathbf{z}^i$
6: –SYNCHRONIZE– (Client sums ρ^i)
7: Read $\mathbf{z}^i$ and $\mathbf{p}^i$
8: **if** $k = 1$ **then**
9: Set $\mathbf{p}^i = \mathbf{z}^i$
10: **else**
11: Set $\beta^i = \rho/\rho_{\mathrm{old}}$
12: Set $\mathbf{p}^i = \mathbf{z}^i + \beta^i \mathbf{p}^i$
13: **end if**
14: Compute $\mathbf{q}^i = A\mathbf{p}^i$
15: Compute $\alpha^i = \rho/(\mathbf{p}^i, \mathbf{q}^i)$
16: Update $\mathbf{p}^i$ and $\mathbf{q}^i$
17: –SYNCHRONIZE– (Client sums α^i)
18: Read $\mathbf{x}^i, \mathbf{r}^i, \mathbf{p}^i$, and $\mathbf{q}^i$
19: Set $\mathbf{x}^i = \mathbf{x}^i + \alpha\mathbf{p}^i$
20: Set $\mathbf{r}^i = \mathbf{r}^i - \alpha\mathbf{q}^i$
21: Update $\mathbf{x}^i$ and $\mathbf{r}^i$
22: Check convergence; continue if necessary
23: Clients sets $\rho_{\mathrm{old}} = \rho$

To increase the granularity we have also implemented the Chronopoulos/Gear variant of CG [5, 21], which has a single synchronisation point, see Algorithm 3. Furthermore, this scheme introduces an additional $2n$ flops in each iteration step compared to the original scheme. Note that we employ matrix–free storage; each matrix element is recomputed when it is needed. Generating the matrix resulting from discretising the Poisson equation with varying diffusivity requires a significant amount of computation, increasing the granularity even further.

2.6 Implementation details

In this section we will discuss some specific issues concerning the various implementations. In the normal operation of GridSolve in combination with DSI, if an input parameter of a task is a DSI file handle, the middleware automatically retrieves the relevant data from the IBP depot before the task is started on the server. For our purposes a task needs full control over a DSI file, so instead we pass the DSI file handle explicitly.

Also, in the current implementation of IBP, reading and writing from and to the IBP depot are blocking operations [22]. Although read operations by differ-

Algorithm 3 CG method with Jacobi preconditioning; Chronopoulos/Gear variant; Task i.

Require: Handles to $\mathbf{x}, \mathbf{r}, \mathbf{w}, \mathbf{p}, \mathbf{q}$, and $\mathbf{s}$ on IBP data depot and parameters α and β.
Ensure: $K := \mathrm{diag}(A)$ and initial values: Solve $\mathbf{w}$ from $K\mathbf{w} = \mathbf{r}$; $\mathbf{s} := A\mathbf{v}; \rho := (\mathbf{r}, \mathbf{w}); \mu := (\mathbf{s}, \mathbf{w})$; $\alpha := \rho/\mu$.
1: *// Each server i performs the following*
2: Read $\mathbf{x}^i$ and $\mathbf{p}^i$.
3: Depending on bandwidth of matrix read appropriate portions of vectors $\mathbf{q}, \mathbf{r}, \mathbf{w}$, and $\mathbf{s}$.
4: Set $\mathbf{p}^i = \mathbf{w}^i + \beta\mathbf{p}^i$
5: Set $\mathbf{q}^i = \mathbf{s}^i + \beta\mathbf{q}^i$
6: Set $\mathbf{x}^i = \mathbf{x}^i + \alpha\mathbf{p}^i$
7: Set $\mathbf{r}^i = \mathbf{r}^i - \alpha\mathbf{q}^i$
8: Check convergence; continue if necessary
9: Solve $\mathbf{w}^i$ from $K\mathbf{w}^i = \mathbf{r}^i$
10: Compute $\mathbf{s}^i = A\mathbf{w}^i$
11: Compute $\rho^i = (\mathbf{r}^i, \mathbf{w}^i)$
12: Compute $\mu^i = (\mathbf{s}^i, \mathbf{w}^i)$
13: Update $\mathbf{x}^i, \mathbf{r}^i, \mathbf{w}^i, \mathbf{p}^i, \mathbf{q}^i$, and $\mathbf{s}^i$ on the depot
14: Return ρ^i and μ^i to client
15: –SYNCHRONIZE–
16: Client sums ρ^i and μ^i for all i
17: Client sets $\beta = \rho/\rho_{\mathrm{old}}$
18: Client computes $\alpha = \rho/(\mu - \rho\beta/\alpha)$
19: Client sets $\rho_{\mathrm{old}} = \rho$

ent tasks can be performed on the same DSI file concurrently, write operations cannot, even when the write regions do not overlap. In the Chronopoulos/Gear scheme a task has to perform six write operations sequentially. Hence if a single DSI file is used to store data, large communication imbalance may occur in this case. We hope to overcome this imbalance by using separate DSI files for each vector and letting each task update the vectors in a random order. By using the DSI functionality, it is also theoretically possible to interrupt the CG iteration process and restart at a later date, using possibly different computational resources.

At the end of each iteration step of the Chronopoulos/Gear variant, it may happen that DSI data is inadvertently overwritten. Specifically, we cannot guarantee that every task has finished reading the data from the previous iteration before other tasks have updated the new data. We therefore use two different DSI files representing the previous and current data and let the client swap the corresponding file handles at the end of each iteration step.

Furthermore, each server node in our experimental setup has ATLAS [23] as a BLAS implementation which is used for the various `daxpy` and inner product operations. Each task recomputes its portion of the sparse matrix A every timestep and stores it using compressed row storage (CRS) format.

Although detection of convergence is an important issue in iterative methods, this is currently not implemented.

3 Experiments

In the previous sections we discussed several implementations and various suggestions for increasing the granularity. In this section we will perform several experiments and investigate the effect of these suggestions on the performance.

Our testbed is a local network of computers, which is a multi–user system consisting of different processors with dynamic workloads. The servers in the network are ten single core (AMD Athlon 64 Processor 3700 at 2.4GHz) and two dual core CPU nodes (Intel Core 2 CPU 6700 at 2.66GHz) with 3 GB and 8 GB of memory respectively and running Linux 2.6.18. In order to perform controlled and repeatable experiments we decided on using idle processors and as a result the partitioning is fixed and homogeneous throughout the experiments. We measure the wall clock times of five CG steps for different values of n.

In the ideal case the IBP depot would be located on a dedicated node connected through a high–speed network to the computational nodes. Unfortunately such a resource was not available at the time of testing, and therefore we opted to start the depot on one of the dual core nodes. As a result we expect a huge communication overhead. The client and the agent are both running on a single core node while the servers are started on the remaining nodes. In a typical Grid environment the client program would be located on the user's desktop machine.

We differentiate between four implementations:

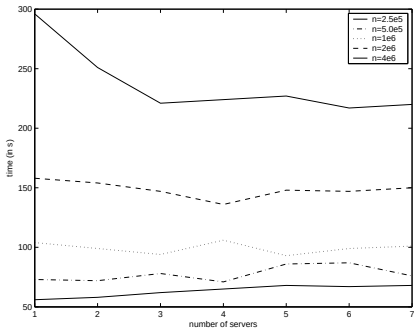
(a) Standard CG without varying diffusivity using a single DSI file;
(b) Chronopoulos/Gear scheme without varying diffusivity using a single DSI file;
(c) Chronopoulos/Gear CG with varying diffusivity using a single DSI file; and
(d) Chronopoulos/Gear CG with varying diffusivity using separate DSI files which are updated in a random order.

Figure 3(a) shows the total wall clock time of the first implementation for different values of n using up to seven servers. It also demonstrates that communication overhead is particularly an issue for small n, which is hardly surprising. In this case using more servers does not result in improved execution times, and even results in larger wall clock times due to communication overhead. For large n this implementation performs slightly better. The other implementations give similar results for small n and we will therefore concentrate on results for large systems.

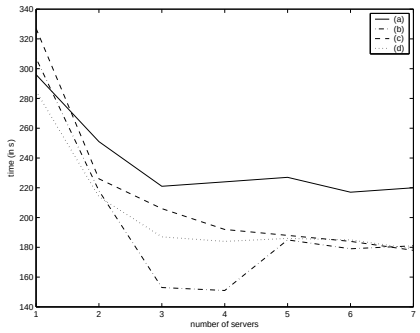
In Fig. 3(b) results are given of the four different implementations for $n = 4 \cdot 10^6$. Here we clearly see the decrease in execution times when using the Chronopoulos/Gear variant for a large number of servers. It is interesting to note that with implementation (b) and using three to four servers we observe a large drop in execution time. We do not have an explanation for this behaviour.

Note that despite our attempts to improve the granularity of the computation, we do not observe any significant improvements.

Although we observed that using separate DSI files for the vectors only improved the overall running time of the Chronopoulos/Gear scheme when using a small number of servers, we noted that in this case the tasks finish at roughly the same time, in contrast to the case of using a single DSI file. This is illustrated in Fig. 4 where the wall clock times of the separate tasks are shown after five CG steps of Chronopoulos/Gear, broken down in both computation and communication. Note that Fig. 4(b) shows that by using separate DSI files the communication becomes more balanced, which is an encouraging result. When using a single DSI file, Fig. 4(a) reveals increasing wall clock times for each subsequent task, which can be explained as follows. Although the client initiates a sequence of non–blocking calls, at the end of the first task the updates to the DSI file appears to block subsequent updates by other tasks. These figures also clearly reveal the amount of communication overhead.



(a) Different values of n, first implementation.



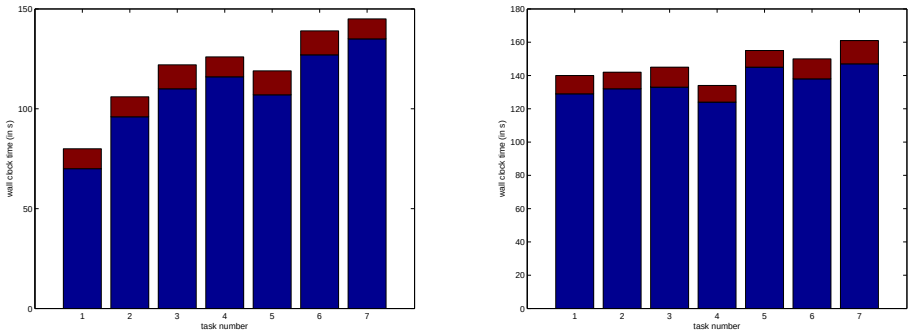
(b) $n = 4 \cdot 10^6$, all four implementations.

Fig. 3. Wall clock times of CG implementations in GridSolve.

4 Concluding remarks and future work

In this work we have described a case study where we have experimented with solving large sparse systems in parallel on Grid computers using a specific Grid middleware and an architecture–aware load balancing algorithm. A sparse iterative solver was implemented in GridSolve, which is mature grid middleware for accessing remote computational resources. Several suggestions for improving the granularity were given and implemented.

Our contribution is threefold. We have used a grid middleware to solve sparse linear systems with CG in a distributed manner. Using the middleware we have



(a) Single DSI file. (b) Multiple DSI files.

Fig. 4. Breakdown of wall clock time of tasks in communication (bottom part) and computation.

also implemented a simple but effective architecture–aware partitioning algorithm to divide the computational work. And finally, we have increased the granularity by using a custom version of CG that has a single synchronisation point.

Clearly our method is only suitable for solving very large systems. Furthermore, it is naturally suited for linear systems with specific classes of coefficient matrices, such as Poisson and Toeplitz matrices. This allows us to use matrix–free storage. Although we did not have a dedicated DSI depot at our disposal the results are promising and we expect that using such a dedicated resource will greatly improve the scalability of our approach.

There are many possible improvements and we will give some suggestions for future work. The current implementation of GridSolve forces us to use bridge communication. SmartGridSolve is an extension of GridSolve which is currently being developed. It will perform similarly to SmartNetSolve [15], allowing for communication between the computational servers as well as data persistence. By combining this with sophisticated (possibly weighted) hypergraph partitioning techniques such as used in Mondriaan [24] we hope to greatly improve our load balancing algorithm.

To further increase the granularity of the computations, we plan to use more sophisticated preconditioning techniques, such as block ILU and deflation.

Another possible improvement is incorporating network capabilities into the partitioning algorithm.

5 Acknowledgments

The authors would like to thank the GridSolve team for their prompt response pertaining to our questions. This work was supported by the Delft Centre for

Computational Science and Engineering within the framework of the project entitled 'Development of an Immersed Boundary Method, Implemented on Cluster and Grid Computers, Application to the Swimming of Fish'.

References

1. Foster I., Kesselman C.: *The Grid: Blueprint for a new Computing Infrastructure,* second edn. Morgan Kaufman Publishers (2004).
2. Anderson D. P., Cobb J., Korpela E., Lebofsky M., Werthimer D.: *SETI@home: an experiment in public-resource computing,* Commun. ACM **45** (2002) 56–61.
3. Hestenes M. R., Stiefel E.: *Methods of conjugate gradients for solving linear systems,* Journal of Research of National Bureau Standards **49** (1952) 409–436.
4. Dongarra J., Li Y., Shi Z., Fike D., Seymour K., YarKhan A.: *Homepage of NetSolve/GridSolve* (2007). `http://icl.cs.utk.edu/netsolve/`, latest version of GridSolve is v0.16.0 (as of 2007-06-03).
5. Chronopoulos A. T., Gear C. W.: S*–step iterative methods for symmetric linear systems,* J. Comput. Appl. Math. **25** (1989) 153–168.
6. Mittal R., Iaccarino G.: *Immersed Boundary Methods,* Annual Review of Fluid Mechanics **37** (2005) 239–261.
7. YarKhan A., Seymour K., Sagi K., Shi Z., Dongarra J.: *Recent Developments in GridSolve,* International Journal of High Performance Computing Applications (IJHPCA) **20** (2006) 131–141.
8. Seymour K., Nakada H., Matsuoka S., Dongarra J., Lee C., Casanova H.: *Overview of GridRPC: A Remote Procedure Call API for Grid Computing,* In: GRID '02: Proceedings of the Third International Workshop on Grid Computing, London, UK, Springer-Verlag (2002) 274–278.
9. Lee C., Nakada H., Tanimura Y.: *GridRPC Working Group* (2007) `https://forge.gridforum.org/projects/gridrpc-wg/`.
10. Caron E., Desprez F.: *DIET: A scalable toolbox to build network enabled servers on the Grid,* International Journal of High Performance Computing Applications **20** (2006) 335–352.
11. Seymour K., YarKhan A., Agrawal S., Dongarra J.: *NetSolve: Grid enabling scientific computing environments,* In Grandinetti, L., ed.: Grid Computing and New Frontiers of High Performance Processing. Elsevier (2005).
12. Tanaka Y., Nakada H., Sekiguchi S., Suzumura T., Matsuoka S.: *Ninf-G: A reference implementation of RPC-based programming middleware for Grid computing,* Journal of Grid Computing **1** (2003) 41–51.
13. Sato M., Boku T., Takahashi D.: *OmniRPC: a Grid RPC system for parallel programming in cluster and Grid environment,* In: CCGRID '03: Proceedings of the 3st International Symposium on Cluster Computing and the Grid, Washington, DC, USA, IEEE Computer Society (2003) 206–213.
14. Caron E., Del-Fabbro B., Desprez F., Jeannot E., Nicod J. M.: *Managing data persistence in network enabled servers,* Sci. Program. **13** (2005) 333–354.
15. Brady T., Konstantinov E., Lastovetsky A.: *SmartNetSolve: High level programming system for high performance Grid computing,* Rhodes Island, Greece, IEEE Computer Society (2006) CD-ROM/Abstracts Proceedings.
16. Lastovetsky A., Zuo X., Zhao P.: *A non–intrusive and incremental approach to enabling direct communications in RPC-based grid programming systems,* Technical report (2006).

17. Zuo X., Lastovetsky A.: *Experiments with a software component enabling NetSolve with direct communications in a non–intrusive and incremental way,* In: Proceedings of the 21st International Parallel and Distributed Processing Symposium (IPDPS 2007), Long Beach, California, USA, IEEE Computer Society (2007).
18. Desprez F., Jeannot E.: *Improving the GridRPC model with data persistence and redistribution,* In: ISPDC '04: Proceedings of the Third International Symposium on Parallel and Distributed Computing/Third International Workshop on Algorithms, Models and Tools for Parallel Computing on Heterogeneous Networks (ISPDC/HeteroPar'04), Washington, DC, USA, IEEE Computer Society (2004) 193–200.
19. Beck M., Arnold D., Bassi A., Berman F., Casanova H., Dongarra J., Moore T., Obertelli G., Plank J., Swany M., Vadhiyar S., Wolski R.: *Middleware for the use of storage in communication,* Parallel Comput. **28** (2002) 1773–1787.
20. Seymour K.: *Private correspondence* (2007).
21. Dongarra J. J., Duff I. S., Sorensen D. C., van der Vorst H. A.: *Numerical Linear Algebra for High Performance Computers,* Society for Industrial and Applied Mathematics, Philadelphia, PA, USA (1998).
22. Zheng Y., Bassi A., Beck M., Plank J. S., Wolski R.: *Internet Backplane Protocol: C API 1.4,* Technical report (2004). `http://loci.cs.utk.edu/ibp/documents/IBPClientAPI.pdf`
23. Whaley R. C., Petitet A.: *Minimizing development and maintenance costs in supporting persistently optimized BLAS,* Software: Practice and Experience **35** (2005) 101–121. `http://www.cs.utsa.edu/~whaley/papers/spercw04.ps`
24. Vastenhouw B., Bisseling R. H.: *A two-dimensional data distribution method for parallel sparse matrix-vector multiplication,* SIAM Rev. **47** (2005) 67–95.

Proceedings of the International Multiconference on
Computer Science and Information Technology pp. 541–555
ISSN 1896-7094
© 2007 PIPS

Automatic First- and Second-Order Adjoints for Truncated Newton

Uwe Naumann[1,2], Michael Maier[1], Jan Riehme, and Bruce Christianson[2]

[1] LuFG Informatik 12, Department of Computer Science, RWTH Aachen University
52056 Aachen, Germany
[2] Department of Computer Science, University of Hertfordshire
Hatfield, AL10 9AB, United Kingdom

Abstract. The analysis and modification of numerical programs in the context of generating and optimizing adjoint code automatically probably ranges among the technically and theoretically most challenging source transformation problems known today. A complete compiler for the target language (Fortran in our case) is needed to cover the technical side. This amounts to a mathematically motivated semantic transformation of the source code that involves the reversal of the flow of data through the program. Both the arithmetic complexity and the memory requirement can be substantial for large-scale numerical simulations. Finding the optimal data-flow reversal schedule turns out to be an NP-complete problem. The same complexity result applies to other domain-specific peephole optimizations. In this paper we present a first research prototype of the NAGWare Fortran compiler with the ability to generate adjoint code automatically. Moreover, we discuss an approach to generating second-order adjoint code for use in Newton-type algorithms for unconstrained nonlinear optimization. While the focus of this paper is mostly on the compiler issues some information on the mathematical background will be found helpful for motivational purposes.

1 Motivation

Suppose that we are interested in the gradient of a given Fortran implementation of a multivariate scalar function $y = f(\mathbf{x})$, $f : \mathbb{R}^n \to \mathbb{R}$, with respect to the elements of its input vector $\mathbf{x}$. Classical numerical differentiation by finite difference quotients computes approximations to the gradient entries by perturbing the corresponding elements in $\mathbf{x}$. There are two major problems with this approach.

1. In infinite precision arithmetic we would make the perturbation parameter h as small as necessary in order to get the desired accuracy of the approximation. Unfortunately, floating-point arithmetic is everything but infinite in precision making the right choice of h a matter of trial and error. Moreover, in most cases we do not even know what to aim for when trying out different values of h. It may be dangerous to trust the derivatives obtained by finite difference approximation.
2. Unidirectional (forward or backward) finite differences require $n+1$ function evaluations. Even for a very simple nonlinear computation such as $f = \prod_{i=1}^{n} x_i$ the runtime can amount to several hours for $n \geq 10^7$.

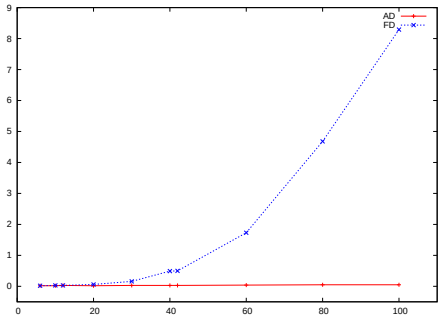


Fig. 1. Runtime (in sec.) of Truncated Newton algorithm for unconstrained minimization of the Elastic Plastic Torsion problem. Derivatives are provided by automatic differentiation (AD) or approximated by finite differences (FD). Compilation time is negligible.

Second-order accuracy can be obtained by centered finite differences. The computational cost though is twice that of the unidirectional approaches.

A semantic source transformation technique known as automatic differentiation [7] can be used to transform numerical codes into tangent-linear and adjoint codes. Adjoint codes compute the gradient with machine accuracy and at a computational cost exceeding that of the original function by merely a constant factor. Accurate second derivatives can be computed by tangent-linear versions of adjoint codes.

In this paper we present a research prototype of the NAGWare Fortran compiler that can generate adjoint code automatically by modifying the original program at the level of the compiler's intermediate representation. The benefit of this approach in the context of numerical algorithms is illustrated in Figure 1 where we compare the runtimes (vertical axis in seconds) of a Truncated Newton algorithm for unconstrained minimization of the Elastic Plastic Torsion problem from the Minpack test problem collection [2] based on the finite difference (FD) and automatic differentiation (AD) approaches for increasing problem sizes n (horizontal axis). In FD mode both the gradients and the Hessian vector products are approximated by finite difference quotients. An adjoint code for the efficient evaluation of accurate gradient information is generated by the compiler automatically in AD mode. The Hessian vector products are evaluated by a tangent-linear version of the adjoint code obtained by operator overloading with automatic type changes performed by the compiler. Obviously, finite differences do not scale well. For $n = 300$ the optimization took approximately 6 minutes on a state-of-the art high-end PC whereas the compiler-generated derivative code lead to a runtime of less than one second. Many more steps had to be performed by the algorithm due to inaccurate numerical approximations of the gradients and the Hessian vector products. Refer to Section 6 for further numerical results.

2 Numerical Background

In the following we recall briefly the main ideas behind the Truncated Newton algorithm originally proposed by Dembo and Steihaug [5].

Suppose that some smooth function f has a vector $\mathbf{x}$ of inputs and a scalar output y, and that we want to find the value of $\mathbf{x}$ for which $y = f(\mathbf{x})$ is a minimum.

Probably the most famous way of doing this is Newton's method (e.g., [9]), which consists of evaluating the gradient $\mathbf{g} = f'$ and Hessian $H = \mathbf{g}' = f''$ of f at a trial point $\mathbf{x}$ and solving the linear equations $H\mathbf{d} = -\mathbf{g}$ for $\mathbf{d}$. Provided that lots of assumptions are satisfied, this gives the value of $\mathbf{d}$ which minimizes the quadratic approximation $q(\mathbf{d}) = f(\mathbf{x}) + \mathbf{g} \cdot \mathbf{d} + \frac{1}{2}\mathbf{d}H\mathbf{d}$ to $f(\mathbf{x} + \mathbf{d})$, and $\mathbf{x} + \mathbf{d}$ will be a better approximation to the optimal value of f than was the starting point $\mathbf{x}$. This process is then repeated iteratively.

Newton's method has two drawbacks: it tends to be distracted by saddle points unless we start off close to the minimum, and the labour of evaluating the complete Hessian and then solving a huge system of equations at every iteration is so immense as to be impractical if $\mathbf{x}$ has a very large number of dimensions.

At the other extreme, a simple approach which is guaranteed to work eventually is to follow the gradient (like walking downhill) by setting $\mathbf{d}$ to be some multiple of $-\mathbf{g}$. This method, which is called *steepest descent*, converges terribly slowly in practice, because the gradient (and hence the correct direction in which to walk) changes continuously as we go along. Really we should adjust the downhill direction $\mathbf{d} = -\mathbf{g}$ by a multiple of $-H\mathbf{d} = H\mathbf{g}$ to allow for the change in gradient caused by the curvature as we move along, and then by a multiple of $H^2\mathbf{d} = -H^2\mathbf{g}$ to allow for the effect on the gradient of the first adjustment, and so on.

Truncated Newton (TN) is a way of iteratively approximating the solution to Newton's equation $H\mathbf{d} = -\mathbf{g}$ for a given *outer* step, by a series $\mathbf{d}_i$ of *inner* steps. These inner steps exactly minimize $q(\mathbf{d}_i)$ in the search space spanned by

$$\{\mathbf{g}, H\mathbf{g}, H^2\mathbf{g}, \ldots, H^{i-1}\mathbf{g}\}.$$

The inner iterations are stopped when the approximation $\mathbf{d}_i$ to the Newton step is *close enough*. As normal, a line search along $\mathbf{d}$ is then performed before the next outer step in order to ensure global convergence.

The TN approach has several advantages over the full Newton method. First, it can avoid saddlepoints by detecting directions of negative curvature (search directions in which f is concave instead of convex). Second, it does not require the full Hessian, but only directional Hessians (vectors of the form $H\mathbf{p}$), which can be efficiently and accurately calculated by the second-order adjoint codes introduced in Section 3.3, even for very large dimensions of $\mathbf{x}$. Thirdly, TN can automatically exploit symmetries in the function f: the number of inner iterations required in theory for exact convergence is not the number of dimensions of $\mathbf{x}$, but the number of distinct eigenvalues of H, which is often much smaller.

The reason for this is that, if $\lambda_1, \lambda_2 \ldots \lambda_k$ are the distinct eigenvalues, then $(H - \lambda_1 I)(H - \lambda_2 I) \ldots (H - \lambda_k I) = 0$, so $H^k\mathbf{g}$ is already a linear combination of lower powers.

Even where the number of distinct eigenvalues of H is large, approximate equalities between them often lead in a smaller number of inner iterations to a good enough approximate Newton solution.

It turns out that an effective way to construct the TN sequence of inner approximations $\mathbf{d}_i$ is via a series of two-dimensional searches on q. At the $i-$th stage the two

search directions (starting from $\mathbf{d}_i$) in which q is minimized are the previous direction of movement $\mathbf{d}_i - \mathbf{d}_{i-1}$, which we can think of as a momentum term, and the direction opposite to the updated gradient $\mathbf{g}_i = \mathbf{g} + H\mathbf{d}_i$ at $\mathbf{d}_i$, which is the steepest descent direction for q (not f) at the current point $\mathbf{d}_i$. (Indeed one way to think of the TN algorithm is as the Conjugate Gradient algorithm applied to q instead of f.) The TN algorithm starts with a one-dimensional search in the steepest descent direction $-\mathbf{g}$. Initialize i to zero.

$$\mathbf{d}_0 = 0; \quad \mathbf{g}_0 = \mathbf{g}; \quad \Delta_0 = 0; \quad \mathbf{p}_0 = -\mathbf{g}_0$$

Now for each i form $H\mathbf{p}_i$ and stop if $\mathbf{p}_i H\mathbf{p}_i < \epsilon \mathbf{p}_i^2$

$$\begin{aligned} \beta_i &= \frac{\mathbf{g}_i^2}{\mathbf{p}_i H\mathbf{p}_i} \\ \mathbf{d}_{i+1} &= \mathbf{d}_i + \beta_i \mathbf{p}_i \\ \mathbf{g}_{i+1} &= \mathbf{g}_i + \beta_i H\mathbf{p}_i \\ \Delta_{i+1} &= \Delta_i + \beta_i^2 \mathbf{p}_i H\mathbf{p}_i/2 \end{aligned}$$

stop if $\mathbf{g}_{i+1}^2/\mathbf{g}_0^2 < \min(\mathbf{g}_0^2, 1.0/i^2)$

$$\alpha_{i+1} = \mathbf{g}_{i+1}^2/\mathbf{g}_i^2; \quad \mathbf{p}_{i+1} = \alpha_{i+1}\mathbf{p}_i - \mathbf{g}_{i+1}$$

Increment i.

The coefficients α_i, β_i are chosen so that $\mathbf{g}_{i+1}$ is orthogonal to both $\mathbf{g}_i$ and $\mathbf{p}_{i-1}$. This ensures that the search directions $\mathbf{p}_i$ generated by the TN algorithm are conjugate, $\mathbf{p}_i H\mathbf{p}_j = 0$ for $i \neq j$, and that the gradients $\mathbf{g}_i$ are orthogonal, $\mathbf{g}_i \cdot \mathbf{g}_j = 0$ for $i \neq j$. With respect to the partial orthogonal basis $\mathbf{g}_i$, the Hessian H is tridiagonal, since $\mathbf{g}_i H\mathbf{g}_j = 0$ for $i > j + 1$.

If TN stops because $\mathbf{p}_i H\mathbf{p}_i/\mathbf{p}_i^2$ is close to zero then we take $\mathbf{d}_i$ as the step direction for the outer iteration. If TN stops because $\mathbf{p}_i H\mathbf{p}_i/\mathbf{p}_i^2$ is significantly negative then we have found a direction of negative curvature, and take the descent direction $\mathbf{p}_i$ as the step direction for the outer iteration.

If TN stops because $\mathbf{g}_{i+1}$ is small (Newton itself is only quadratically convergent, so there is no point in trying to do better than this) then we know that H is positive definite in the space spanned by the $\mathbf{g}_i$. (Any linear combination of the $\mathbf{g}_i$ can be written as $\mathbf{d} = \sum_i \gamma_i \mathbf{p}_i$ whence by conjugacy $\mathbf{d}H\mathbf{d} = \sum_i \gamma_i^2 \mathbf{p}_i H\mathbf{p}_i$.) Note that $\mathbf{g}_i^2$ is not guaranteed to decrease at each step, although q does decrease at each step, by the amount $\beta_i^2 \mathbf{p}_i H\mathbf{p}_i/2$. (This is because $\mathbf{g}_0 \cdot \mathbf{p}_i = -\mathbf{g}_i^2$, whence $\mathbf{g}_0 \cdot \beta_i \mathbf{p}_i = -\beta_i^2 \mathbf{p}_i H\mathbf{p}_i$.) Hence $\mathbf{d}_i$ is a descent direction, with $\mathbf{g}_0 \cdot \mathbf{d}_i = -\mathbf{d}_i H\mathbf{d}_i$.

Another condition under which it is desirable to stop the inner iterations is when $\mathbf{d}_i$ moves into a region where the quadratic model q breaks down. One way to detect this is periodically to compare $f(\mathbf{x}) - f(\mathbf{x} + \mathbf{d}_i)$ with Δ_i and stop if the relative error is more than a specified proportion. Alternatively $\mathbf{g}(\mathbf{x} + \mathbf{d}_i)$ can be periodically compared with $\mathbf{g}_i$ in some norm. This breakdown could be due to $\mathbf{d}_i$ growing too large (trust regions are sometimes used in conjunction with TN to monitor this) but it could also be due to the accumulation of roundoff errors in the values for $\mathbf{g}_i, \mathbf{p}_i$. The TN algorithm responds well when provided with accurate second derivatives, which allow a much larger number of inner iterations to be taken before this second form of breakdown occurs.

3 AD Compiler's View on Numerical Code

In this section we introduce parse tree manipulation algorithms [1] for the compiler-based generation of tangent-linear, adjoint, and second-order adjoint numerical programs. We consider (Fortran) implementations $F(\overset{\downarrow}{\mathbf{x}}, \underset{\downarrow}{\mathbf{y}})$ of multivariate vector functions $\mathbf{y} = F(\mathbf{x})$ where $\mathbf{x} \in I\!R^n, \mathbf{y} \in I\!R^m$. An overset downarrow marks an input. An underset downarrow marks an output.

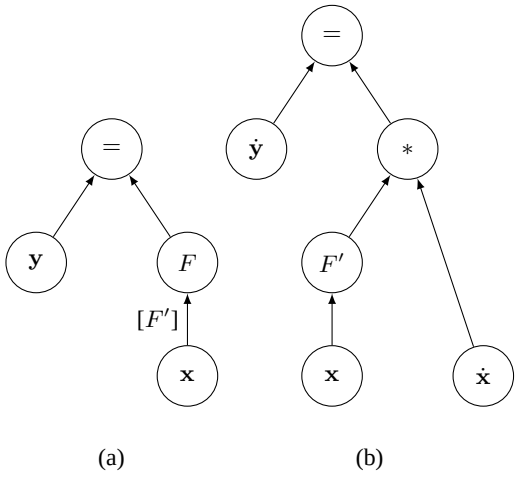


Fig. 2. From F (a) to $\dot{F}$ (b) by Parse Tree Manipulation

3.1 Tangent-Linear Code $\dot{F}$

The signature of the tangent-linear code is $\dot{F}(\overset{\downarrow}{\mathbf{x}}, \overset{\downarrow}{\dot{\mathbf{x}}}, \underset{\downarrow}{\dot{\mathbf{y}}})$ where

$$\dot{\mathbf{y}} = F'(\mathbf{x}) * \dot{\mathbf{x}}, \quad \dot{\mathbf{x}} \in I\!R^n, \ \ \dot{\mathbf{y}} \in I\!R^m \quad .$$

$F' \equiv F'(\mathbf{x})$ denotes the $(m \times n)$-matrix of the partial derivatives of all outputs with respect to all inputs (the Jacobian matrix). The gradient of an output is represented by the respective row in F'.

The tangent-linear parse tree manipulation algorithm for assignments of the form $\mathbf{y} = F(\mathbf{x})$ is illustrated in Figure 2. The right-hand side is linearized by labeling the edge $(\mathbf{x}, F)$ with the symbolic partial derivative $F' = F'(\mathbf{x})$ of its target with respect to its source. The tangent-linear assignment overwrites $\dot{\mathbf{y}}$. The right-hand side of the tangent-linear assignment is build top-down for each node by summation over all incoming edges of the products of the respective edge's label with the tangent-linear subtree of its source. All leaf nodes are replaced by their respective tangent-linear symbols,

that is, $\dot{\mathbf{x}}$ and $\dot{\mathbf{y}}$. The partial derivatives are functions of the respective edge's source. This dependence is also represented in the tangent-linear parse tree.

For $y = x_1 * x_2$ and with $F' = (x_2, x_1)$ we simply get the product rule of differentiation as

$$\dot{y} = F' * \begin{pmatrix} \dot{x}_1 \\ \dot{x}_2 \end{pmatrix} = x_2 * \dot{x}_1 + x_1 * \dot{x}_2 \quad .$$

The recursive nature of the algorithm will become clearer in the context of second-order adjoints illustrated by the transition from Figure 3 (a) to Figure 3 (b).

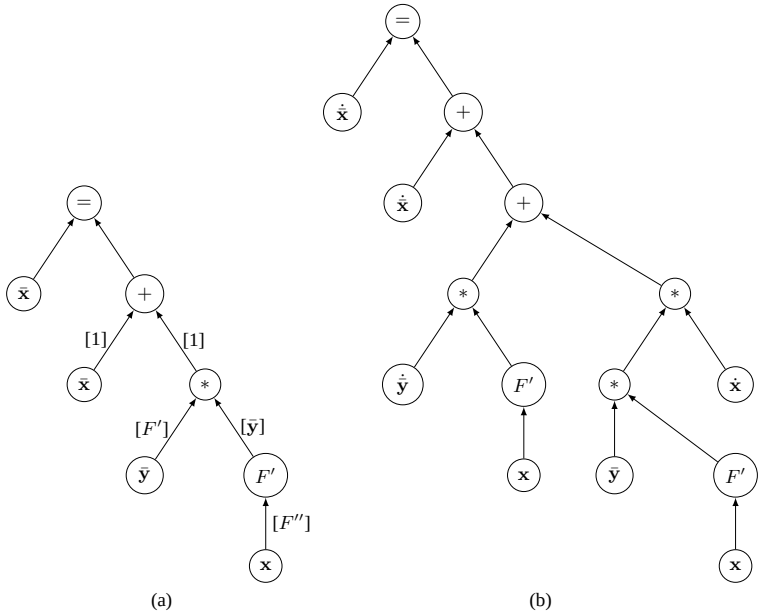


Fig. 3. Linearized Parse Tree of $\bar{F}$ (a); Parse Tree of $\dot{\bar{F}}$ (b)

3.2 Adjoint Code $\bar{F}$

The signature of the adjoint code is $\bar{F}(\overset{\downarrow}{\mathbf{x}}, \underset{\downarrow}{\overset{\downarrow}{\bar{\mathbf{x}}}}, \overset{\downarrow}{\bar{\mathbf{y}}})$ where

$$\bar{\mathbf{x}} = \bar{\mathbf{x}} + F'(\mathbf{x})^T * \bar{\mathbf{y}}, \quad \bar{\mathbf{x}} \in I\!R^n, \ \ \bar{\mathbf{y}} \in I\!R^m.$$

The adjoint parse tree manipulation algorithm for assignments of the form $\mathbf{y} = F(\mathbf{x})$ starts with the linearization of the right-hand side as in Section 3.1. As an immediate consequence of the chain rule the adjoint assignment increments $\bar{\mathbf{x}}$ with the product of the row vector $\bar{\mathbf{y}}$ with the local Jacobian matrix $F'(\mathbf{x})$. The incrementation is due to the data-flow reversal that accumulates adjoints of variables that are read by the original assignment. The corresponding instances can potentially be read by several assignments.

Hence, the respective adjoints need to be composed incrementally out of contributions from all of those assignments. Moreover, all adjoints need to be initialized properly (e.g., to zero for local program variables). The algorithm is illustrated in Figure 3 (a). The adjoint parse tree is linearized in order to derive the second-order adjoint code in the next section. Refer to Section 4.1 for a description of the globalization of the data-flow reversal.

For $y = x_1 * x_2$ we get $(\bar{x}_1, \bar{x}_2) = (\bar{x}_1, \bar{x}_2) + \bar{y} * (x_2, x_1)$ resulting in the two scalar assignments

$$\bar{x}_1 = \bar{x}_1 + \bar{y} * x_2; \;\; \bar{x}_2 = \bar{x}_2 + \bar{y} * x_1 \quad . \tag{1}$$

3.3 Second-Order Adjoint Code $\dot{\bar{F}}$

Second-order adjoint code can be generated by building a tangent-linear version of the adjoint code. The signature of the second-order adjoint code is $\dot{\bar{F}}(\overset{\downarrow}{\mathbf{x}}, \overset{\downarrow}{\dot{\mathbf{x}}}, \underset{\downarrow}{\overset{\downarrow}{\bar{\mathbf{x}}}}, \overset{\downarrow}{\bar{\mathbf{y}}}, \overset{\downarrow}{\dot{\bar{\mathbf{y}}}})$ where

$$\dot{\bar{\mathbf{x}}} = \dot{\bar{\mathbf{x}}} + \dot{\bar{\mathbf{y}}} * F'(\mathbf{x}) + \bar{\mathbf{y}} * F''(\mathbf{x}) * \dot{\mathbf{x}} \quad \dot{\bar{\mathbf{x}}} \in I\!R^n, \;\; \dot{\bar{\mathbf{y}}} \in I\!R^m \quad .$$

Products of the Hessian matrix $F'' = F''(\mathbf{x}) \in R^{n \times n}$ with a vector $\dot{\mathbf{x}}$ can be computed by setting $\dot{\bar{\mathbf{x}}} = \dot{\bar{\mathbf{y}}} = 0$ and $\bar{\mathbf{y}} = 1$ at roughly twice the cost of running the adjoint code. The benefit for TN is illustrated in Figure 1. Mathematical details are outlined in Section 2.

In order to generate second-order adjoint code for the simple assignment $y = x_1 * x_2$ we apply the rules for tangent-linear parse tree construction from Section 3.1 to Equation (1) to obtain

$$\dot{\bar{x}}_1 = \dot{\bar{x}}_1 + \dot{\bar{y}} * x_2 + \bar{y} * \dot{x}_2; \;\; \dot{\bar{x}}_2 = \dot{\bar{x}}_2 + \dot{\bar{y}} * x_1 + \bar{y} * \dot{x}_1 \quad .$$

Setting $\dot{\bar{x}}_1 = \dot{\bar{x}}_2 = \dot{\bar{y}} = 0$ and $\bar{y} = 1$ we get

$$F'' \begin{pmatrix} \dot{x}_1 \\ \dot{x}_2 \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} \dot{x}_1 \\ \dot{x}_2 \end{pmatrix} = \begin{pmatrix} \dot{x}_2 \\ \dot{x}_1 \end{pmatrix}$$

as expected. Figure 3 (b) illustrates the general approach.

4 Implementation

The conceptual insight into derivative code generation gained so far needs to be commented on further in order to understand the work of an AD compiler. Various additional issues arise when implementing the theory.

4.1 Data-Flow Reversal

The easiest way to reverse the flow of control is to enumerate all assignments in the original program and to push their indexes onto a control stack during an appropriately instrumented program execution (the augmented forward sweep). The original order is reversed by popping the indexes from the control stack. The following code listing is built on the example $f = \prod_{i=1}^{n} x_i$ from Section 1.

```
1   ...
2   integer i
3   call push_c(1)
4   f=x(1)
5   do i=2,n
6     call push_c(2)
7     call push_d(f); call push_d(i)
8     f=f*x(i)
9   end do
10  ...
11  do while (.not.empty_c())
12    int i
13    call pop_c(i)
14    if (i==1)
15      ...
16    else if (i==2) then
17      call pop_d(i); call pop_d(f)
18      b_x(i) = b_x(i) + f*b_f
19      b_f=x(i)*b_f
20    end if
21    ...
22  end do
```

Pushes to and pops from the control stack are performed by the subroutines `push_c` and `pop_c`. The Boolean function `empty_c` tests for emptiness of the control stack.

Instances of program variables may be required in reverse order if they occur as arguments of nonlinear intrinsics or arithmetic operators. However these instances may be lost during the augmented forward sweep due to overwrites. Such instances need to be made persistent by pushing them onto a data stack before the corresponding memory gets overwritten. The required values need to be restored prior to their use in the computation of a local partial derivative. For example, `f` is overwritten repeatedly in line 8. A push of the old instance of `f` onto the data stack needs to be inserted in front of line 8. The matching pop precedes the corresponding adjoint statements in the backward sweep. Special care must be taken for address computations in order to reference the correct adjoint memory. See lines 7 and 17. Various domain-specific static code analyses and optimizations must be performed to be able to handle large-scale numerical simulation programs. They are part of our to do list. A complete description of the issues arising in adjoint code generation is beyond the scope of this paper. To our knowledge there is no single publication that covers it all. Refer, for example, to [6] for further reading.

4.2 AD by Overloading

A very robust and convenient way to implement tangent-linear code is by overloading of a programming language's arithmetic operators and intrinsic functions for a derived data type that contains in addition to the original function value v a component for the derivative $\dot{v}$. All active program variables (those that potentially carry nonzero derivatives) need to get their types changed to the new data type. This is done automatically

by the compiler as outlined in Section 5.2. A static data-flow analysis can compute a conservative estimate for this set [8]. The overhead of the additional function call can be eliminated by the compiler through inlining. Below we show excerpts from the Fortran module that we use for propagating directional derivatives in the context of second-order adjoint codes as described in Section 5.1.

```
1  module compad_module
2  implicit none
3  private
4  public :: compad_type, tls_type
5  public :: assignment(=), operator(*) ...
6
7  type tls_type
8    double precision :: v=0
9    double precision :: d=0
10 end type tls_type
11
12 type compad_type
13   type(tls_type) val
14   type(tls_type) drv
15 end type compad_type
16
17 interface assignment(=)
18   module procedure tls_assign_tls
19 end interface assignment(=)
20
21 interface operator(*)
22   module procedure tls_times_tls
23 end interface operator(*)
24 ...
25 contains
26
27 elemental subroutine tls_assign_tls(y,x)
28   type(tls_type), intent(out) :: y
29   type(tls_type), intent(in) :: x
30   y%v=x%v; y%d=x%d
31 end subroutine tls_assign_tls
32
33 elemental function tls_times_tls(x1,x2) result(y)
34   type(tls_type), intent(in) :: x1,x2
35   type(tls_type) :: y
36   y%v=x1%v*x2%v; y%d=x1%d*x2%v+x2%d*x1%v
37 end function tls_times_tls
38 ...
39 end module compad_module
```

Conceptually, adjoint codes can be implemented by overloading too. However, the interpretive overhead that results from the mapping of the real address space onto a virtual one may be significant. Moreover, static code optimization is even more relevant

in adjoint codes. Refer to [4] and [12] for further details on the use of overloading techniques in the differentiation-enabled NAGWare Fortran compiler.

5 AD Compiler Support

In this section we describe a hybrid approach to second-order adjoint code generation that uses extensive parse tree manipulation techniques in addition to overloading. Our focus is on the discussion of the modifications of the compiler's internal representation needed to make the generation of second-order adjoint code fully automatic for the set of test problems currently under consideration. We expect further work to become necessary to cover the full Fortran standard. However, a large step has been made already.

5.1 A Hybrid Approach

We use parse tree manipulation techniques to generate the adjoint code as described in Section 5.3. Second-order adjoints are computed by overloading the adjoint code as outlined in Section 4.2. The use of the second-order adjoint code is best illustrated with an example: Consider the Fortran routine

```
1 subroutine F(x,y)
2   double precision :: x
3   double precision :: y
4   y=sin(x)
5 end subroutine
```

The parse tree is manipulated internally to produce adjoint code such that adjoints are associated with original variables *by address*; that is, v (`val`) and $\bar{v}$ (`drv`) are the two components of a derived type `compad_type`. Both of them are of type `tls_type` (with components `v` and `d` of type `double precision` as shown in Section 4.2). Thus, directional derivatives of the adjoints (second-order adjoints) can be computed by overloading. Conceptually, the following code is generated automatically by the compiler:

```
1 subroutine dbF(x,y)
2   use compad_module
3   type(compad_type) :: x
4   type(compad_type) :: y
5   y%val=sin(x%val)
6   x%drv=x%drv+cos(x%val)*y%drv
7 end subroutine
```

The second-order adjoint code is never unparsed. Instead the routine `dbF` is used in a driver routine (for example, in the context of TN) as illustrated below.

```
1 program main
2   use compad_module
3   external dbF
4   type(compad_type) x,y
5
```

```
6   x%val%v=1.;   x%val%d=.5
7   x%drv%v=0.;   x%drv%d=0.; y%drv%v=1.;   y%drv%d=0.
8
9   call dbF(x,y)
10
11  print*, "x= ", x%val; print*, "bx= ", x%drv;
12  print*, "y= ", y%val; print*, "by= ", y%drv
13 end program
```

Here we simply print the results producing the following (edited) output:

```
1  x=     1.000000    0.5000000
2  bx=    0.540302   −0.4207354
3  y=     0.841470    0.2701511
4  by=    1.000000    0.0000000
```

The interesting value is that of $\dot{\bar{x}} =$ `x%drv%d` (line 2, second value) holding the product of the second derivative of $\sin(x)$, that is, $-\sin(x)$, with $\dot{x} = 0.5$ at point $x = 1$. Take the value of $y = \sin(x)$ (line 3, first value) and divide it by -2 to verify numerical correctness.

5.2 Support for Overloading

The compiler provides support for overloading by automatic type changes and by importing a module named `compad_module`. As simple as it might sound, changing the types of all floating-point variables from `REAL` or `DOUBLE PRECISION` to `compad_type` while parsing the source code yields a number of technical challenges.

`use compad_module` statements must be inserted into all scopes that are to be differentiated.

Calls to structure constructors of type `compad_type` need to be generated for right-hand sides of initializations, for `DATA` entries, and whenever a named or literal constant is passed as an actual parameter to a subroutine that expects a dummy parameter of type `compad_type`.

The types of named floating-point constants is also initially changed to `compad_type`. Derivatives with respect to constants are known to be identically zero. Hence, their type should remain unchanged meaning that the changes that have already happened need to be reversed. Fortran offers a large variety of syntactical constructs making the development of robust parse tree manipulation algorithms a major technical challenge.

5.3 Support for Parse Tree Manipulation

The adjoint parse tree manipulation algorithm is an extension of our original approach. See [10] for details. That used association *by name* for augmenting the original memory space with adjoint variables, that is, v and $\bar{v}$ were two separate variables of type `double precision` linked via some naming convention, for example `v` and `_v`.

Our new method uses association *by address*. Moreover, the components of the corresponding derived type can be of derived type themselves, thus allowing for overloading to be applied to the adjoint code as described above.

6 Case Study

We consider the application of TN to a Fortran implementation of the well-known Rosenbrock function

$$f(x_1, x_2) = (1 - x_1)^2 + 100(x_2 - x_1^2)^2.$$

The Hatfield NOC OPTIMA TN algorithm expects the user to provide the following three subroutines:

- CALFUN to compute the function value FF at point X;
- CALGRD to compute the gradient G at point X;
- CALHP to compute the product HP of the Hessian at point X with a vector P.

Both CALGRD and CALHP use the second-order adjoint routine dbRosenbrock that is generated automatically by the compiler. All three routines are shown in the following code listing.

```
1  SUBROUTINE CALFUN(X,N,FF,...)
2  ...
3  DOUBLE PRECISION X(N)
4  t1 = (1-X(1)); t2 = t1*t1; t3 = X(2)-X(1)*X(1); FF  = t2 + 100*
       t3*t3
5  RETURN
6  END
7
8  SUBROUTINE CALGRD(X,N,FF,G,...)
9  use compad_module
10 ...
11 external dbRosenbrock
12 DOUBLE PRECISION X(N),G(N)
13 type(compad_type) ctx(N), ctf
14
15 ctx(1)%val%v = X(1); ctx(1)%val%d = 0.; ctx(1)%drv%v = 0.; ctx
       (1)%drv%d = 0.
16 ctx(2)%val%v = X(2); ctx(2)%val%d = 0.;  ctx(2)%drv%v = 0.; ctx
       (2)%drv%d = 0.
17
18 ctf%val%v = 0.; ctf%val%d = 0.; ctf%drv%v = 1.; ctf%drv%d = 0.
19
20 call dbRosenbrock(ctx,ctf)
21
22 G(1) = ctx(1)%drv%v; G(2) = ctx(2)%drv%v
23 RETURN
24 END
25
26 SUBROUTINE CALHP(X,G,N,P,HP,...)
27 use compad_module
28 ...
29 external dbRosenbrock
```

```
30 DOUBLE PRECISION X(N),G(N),P(N),HP(N)
31 type(compad_type) ctx(N), ctf
32
33 ctx(1)%val%v = X(1); ctx(1)%val%d = P(1); ctx(1)%drv%v = 0.; ctx
       (1)%drv%d = 0.
34 ctx(2)%val%v = X(2); ctx(2)%val%d = P(2); ctx(2)%drv%v = 0.; ctx
       (2)%drv%d = 0.
35 ctf%val%v = 0.; ctf%val%d = 0.; ctf%drv%v = 1.; ctf%drv%d = 0.
36
37 call dbRosenbrock(ctx,ctf)
38
39 HP(1) = ctx(1)%drv%d; HP(2) = ctx(2)%drv%d
40 RETURN
41 END
```

The gradient at the current point X (lines 15 and 16) is computed by setting the adjoint ctf%drv%v of FF to one (line 18). Its entries are accumulated in the adjoint components of X (line 22).

The product of the Hessian with the vector P at point X is computed by initializing the directional derivatives of X(1) and X(2) to P(1) and P(2), respectively (lines 33 and 34). Again, the adjoint of FF needs to be set to one (line 35). The entries of the corresponding directional Hessian occur as the directional derivatives of the adjoints of X (line 39).

The semantic transformation of the original function evaluation routine is taken over entirely by the compiler. Writing corresponding code by hand and debugging it may require substantial manpower for larger problems.

With exact first- and second-order adjoints TN performs 21 major and 38 minor iterations. Using finite difference approximations for the first and second-derivatives makes these numbers grow to 1222 and 1975, respectively. The stopping criterion is satisfied if the Euclidean norm of the gradient is less than 10^{-5}. Starting from $\mathbf{x} = (-1.2, 1)$ both algorithms converge to the correct solution $(1, 1)$.

To investigate the impact on runtime we consider extended versions (with variable n) of the Rosenbrock and Powell functions [11]. For increasing n we compare the numbers of minor and major iterations performed by TN based on exact derivatives (AD) and finite difference approximations (FD). Additionally we show the respective wall clock times in seconds. The Euclidean norm of the gradient is driven down to 10^{-6} with finite difference perturbations of 10^{-8}. We can afford an even lower value of the residual ($\leq 10^{-9}$) for AD. With AD the minimum is found in all cases. FD fails occasionally.

6.1 Extended Rosenbrock Function

n	Minor AD	Minor FD	Major AD	Major FD	Time AD	Time FD
10	1498	10042183	176	9825984	0.15	49.88
20	4791	7950184	390	6175438	0.53	383.65
50	15561	8588441	809	8474209	2.08	4645.32

With finite differences the TN algorithm terminated due to missing further decrease in the function value. The convergence criterion could not be satisfied in this case.

6.2 Extended Powell Function

n	Minor AD	Minor FD	Major AD	Major FD	Time AD	Time FD
10	102	248445	15	62119	0.01	2.66
20	109	248464	15	62124	0.01	14.89
50	121	248490	16	62131	0.02	189.33

The inaccuracy of the finite difference approximation (of the projected Hessian in particular) leads to unwanted search steps taken by the TN algorithm. Consequently the algorithm exhibits a poor convergence behavior. Exact derivatives are the remedy. The ability to compute those automatically makes things highly convenient.

7 Conclusion and Outlook

Writing a compiler for the automatic generation of robust and efficient adjoint and second-order adjoint codes is a highly challenging task. Substantial source code analysis and manipulation is required at the level of the compiler's internal representation. The availability of such a compiler facilitates a faster transition form pure numerical simulation to derivative-based optimization of nonlinear mathematical models. Thus, the compiler is likely to become a central element in the toolbox of every computational scientist or engineer.

We are in the process of developing a research prototype of the NAGWare Fortran compiler that provides the previously mentioned functionalities. Our current test set comprises problems from the Minpack test problem collection as well as several others. Feasibility and usefulness of the adjoint compiler approach has been illustrated with the help of a Truncated Newton algorithm for unconstrained nonlinear optimization where gradients and projected Hessians are provided automatically by the compiler for a given implementation of the objective function. The efficient gradient evaluation via adjoints combined with the accuracy of the obtained numerical values resulted in a speedup of an order of magnitude when compared with derivatives approximated by finite differences.

References

1. A. Aho, R. Sethi, and J. Ullman. *Compilers. Principles, Techniques, and Tools*. Addison-Wesley, 1986.
2. B. Averik, R. Carter, and J. More. The Minpack-2 test problem collection (preliminary version). Technical Report 150, Mathematical and Computer Science Division, Argonne National Laboratory, Argonne, IL, 1991.
3. M. Bücker, G. Corliss, P. Hovland, U. Naumann, and B. Norris, editors. *Automatic Differentiation: Applications, Theory, and Tools*, volume 50 of *Lecture Notes in Computational Science and Engineering*. Springer, 2005.

4. M. Cohen, U. Naumann, and J. Riehme. Towards differentiation-enabled Fortran 95 compiler technology. In *Proceedings of the 18th ACM Symposium on Applied Computing, Melbourne, Florida, USA, March 9–12, 2003*, pages 143–147, 2003.
5. R. Dembo and T. Steihaug. Truncated-Newton algorithms for large-scale optimization. *Math. Prog.*, 26:190–212, 1982.
6. R. Giering and T. Kaminski. Recipes for adjoint code construction. *ACM Transactions on Mathematical Software*, 24:437–474, 1998.
7. A. Griewank. *Evaluating Derivatives. Principles and Techniques of Algorithmic Differentiation.* SIAM, Apr. 2000.
8. L. Hascoët, U. Naumann, and V. Pascual. To-be-recorded analysis in reverse mode automatic differentiation. *Future Generation Computer Systems*, 21:1401–1417, 2005.
9. C. T. Kelley. *Solving Nonlinear Equations with Newton's Methods*. SIAM, Philadelphia, 2003.
10. M. Maier and U. Naumann. Intraprocedural adjoint code generated by the differentiation-enabled NAGWare Fortran compiler. In G. Montero B.H.V. Topping and R. Montenegro, editors, *Proceedings of the Fifth International Conference on Engineering Computational Technology*, page paper 112. Civil-Comp Press, Kippen, Stirlingshire, United Kingdom, 2006.
11. J. Moré, B. Garbow, and K. Hillstrom. Testing unconstrained optimization software. *ACM Trans. Math. Softw.*, 7(1):17–41, 1981.
12. U. Naumann and J. Riehme. Computing adjoints with the NAGWare Fortran 95 compiler. In *[3]*. 2005.

7th International Multidiscipinary Conference on Electronic Commerce (ECOM-07)

We would like to invite original papers on all aspects of electronic commerce and related issues. The conference is meant to address the academic community as well as representatives of business, industry, government, and information technology sector. Thus - apart from research papers - practical presentations of existing solutions are very welcome, too.
The areas of E-Commerce, E-Governance etc. are interdisciplinary by nature: they involve people with their background in economy, management, artificial intelligence, computer science, sociology, psychology, law and so on. We feel that a meeting of such people may help to cross the boundaries between the traditional disciplines, but also between the communities of theorists and practitioners. This can create a better understanding of the mechanisms underlying electronic markets and organizations.

This year we would like to emphasize as a special track

- Usability in Systems and Applications

The general list of other topics includes (but is not limited to):

- Electronic Markets and Electronic Marketing
- Languages and Models for E-Commerce
- Artificial Intelligence in E-Commerce
- Electronic Contracting and Legal Aspects of E-Commerce
- Electronic Interaction and Negotiation
- Virtual Enterprises
- Technology and E-Commerce
- Internet Computing and Electronic Commerce
- Social Side of E-Commerce
- Futurology of E-Commerce

International Programme Committee

- Leszek Borzemski, Wroclaw University of Technology, Poland
- Jen-Yao Chung, IBM, USA
- Matjaz Gams, Jozef Stefan Institute, Slovenia
- Jerzy Grzymała-Busse, University of Kansas, USA
- Grzegorz Karasiewicz , Warsaw University, Poland
- Leszek Maciaszek, Macquarie University, Australia
- Juho Maekioe, Research Center for Information Technologies, Germany
- Olle Olsson, Swedish Institute of Computer Science, Sweden
- Anne-Marie Oostveen, Oxford Internet Institute Oxford University, UK
- David Ruiz, Universidad de Sevilla, Spain
- Andrzej Targowski, Western Michigan University, USA

Organizing Committee

Jacek Wachowicz, Gdansk University of Technology, Poland

Urszula Świerczyńska-Kaczor, Świętokrzyska Academy in Kielce, Poland

Proceedings of the International Multiconference on
Computer Science and Information Technology pp. 559–563
ISSN 1896-7094
© 2007 PIPS

Legal Aspects of Deep Links on the Internet

Artur Strzelecki

The Karol Adamiecki University of Economics in Katowice
Information Science Department
strzelecki@ae.katowice.pl

Abstract. Links are found everywhere and they are these what creates the World Wide Web. Possibility to use the whole network is the essence of polycentric decentralized structure of the Internet. The user may quickly and easily surf web sites. The search engines reach editorial texts and photos presented in the news through deep links. Skipping the home page of a given web site normally means noting lower amount of visits. The amount of visits has an impact on the income from adverts. Problem of an effective defense against unwanted deep links occurs especially in the media where generally contents are protected by copyright. Many e-commerce enterprises want to be on the highest position in the results of searching. Search engines copy up-to-date press news or directly refers to them. There arises dispute over ownership and the aim of using copied information. The article discusses the significance and effects of using deep links in up-to-date press contents coming from the search engines. There are presented practical examples of disputes occurring on the basis of deep links. The author proposes the method of using deep links as a selling propulsion in the news services.

1 Introduction

Link is an electronic anchor composed into the structure of the web site. Putting the link into foreign web site on someone's page does not require permission of the owner. It is assumed that creating the web site and placing it in the network is equivalent with consent to referring back to it. The owner of the web site often does not know that there were placed links to this web site ([4]).

Taking into consideration the criteria of reference to outside web site links can be divided ([6]). Surface links point to home page of a completely different web site. Deep links directly indicate to the substantial contents of concrete web site with omission of the home page.

In general, links are profitable for the web site. The more often a given link appears on various web sites the higher will be the position in the search engine results. Especially, enterprises that get income from e-business are interested in that effect. In some situations link may be treated as unconstitutional. It is possible to talk about exceptions of freedom of placing links in the web. In the concrete case it can be recognized as acts of dishonest competition or violation of copyright.

Mainly, it refers to putting deep links, using references leading directly towards inside content of the web site with omission of the home page. Such operation reduces the number of visits on the publisher's home page because the user goes directly to a given web page. When using the search engine the access to the materials is possible directly from the search engine results page omitting the home page of a given web site. In the results of searching very often appear first words of specified press material as a description of a given link. Omitting the home page of a given web site means putting down smaller number of visits. It is visible in the income from adverts. The adverts' proprietors express smaller interest or the web site must reconcile with the reduction of rates which they are willing to pay.

In reality, solution for the observed situations in case of links is not homogeneous. It is obvious that if the author creates a web site which he will promote on his own or put links into it leading from the outside, he must take into consideration the fact that others will put the links without his permission. He must also remember that no-one will endeavor to get permission for this link. The situation will look differently when it comes to deep links. They lead directly to the substantial contents of a foreign web site. If there happens the omission of the home page that can include adverts, the publisher can notice danger in it.

The compromise solution induce to recognize deep links but they suggest some guidelines for web sites. Every page of the web site must inform the user where in the structure of the web site it is. It should also include link to the home page and characterize the content of the whole web site in general.

2 The Links in the News Services

The problem of defending against unwanted deep links is especially in media. Most often, the content is protected with the copyright. Many e-commerce enterprises willing to be on the highest position of search rankings copy current press news. Even search engines themselves copy those news or refer to them directly.

Here arises dispute over ownership and the aim of using copied news. If the author or the newspaper does not want others to popularize the whole content of the composition, there are many technological solutions that can help to block the access.

On the other hand, every user of the web can get to the published information on his own without any restrictions from the publisher. Being familiar with URL address and manual inserting it into the address bar of the web browser is sufficient. The deep links are only the technological equivalents of inscribing such web site addresses though hyper-textual links net is the basis of web work. The web site can be compared with the poster. This what is visible on it is available to the public. The dissemination of the content happens when the web site is uploaded on the server. Every user can evoke its content immediately without necessity of using the link. Knowing the address is enough.

The result of searching is fragmentation of the news in the news service. They are not only in carefully edited articles on the first pages of newspapers but also in the result pages in the search engine. The search engines currently lead one of the biggest news service in the web. News service's of the newspapers are afraid of the reduction of their income by searching. Due to that fact they demand paying subscription. At the same time they do not operate deep links thanks to which press news get on the highest positions of searching of the search engines rankings. When looking in the www for the daily portion of news, analysis and opinions, it is difficult to come across materials from newspapers which defend oneself against deep links ([1]).

There are solutions which tone down the conflict between press publishers and search engines operators. For example by blocking the content to companies which professionally occupy oneself with searching the information. There are used new variants of adverts which are connected with the topic of the article. Those adverts do not show to every user of the web but only when the user is looking for the concrete news. It happens when contextualized advert refers to specific key words. There is possibility to prepare web site in such a way that it would be impossible to put links to its resources with the omission of the home page.

More and more common are portals which collect payment for browsing one's own records, for example gazeta.pl, rzeczpospolita.pl. In such circumstances, resources are saved by codes that are available when purchasing proper access. If there is lack of such protections it is not fair to have something against the user who guessed the URL to given resources. It is good to remember that the very fact of drawing someone away from another transactor does not disturb the competition. This action is not honest when one transactor ingratiate with someone else's clientele, standing between mentioned customers and another transactor.

3 The Link As a Sale's Motive Power

Currently the publishers open their own portals and are aware that in the Internet already exists different model of informing. Portals that protect themselves against opening to everyone lose their importance. It doesn't happen because texts and analysis are weak but rather because of not adapting to the new circulation of the news‘ ecosystem. Publishers may allow creating links to articles that are put in the web by themselves but limit total using of portal to paid subscribers.

Such action should lead to the discovery of the future subscribers. The publishers who do not offer additional benefits from paid subscription except articles do not pay attention to recipient's demand. Most probable is that many people will pay for the subscription to materials suggested by respectable sources. It can be friends who send links by e-mail, instant messages, blogs or by other news service. We may risk a statement that target pages of such links are places where publishers will catch new readers in the easiest way. If the user appears on

the page due to recommendation of trusted source and if the page is interesting enough, it will convince the guest to the value of the remaining part of portal. Eventually, the reader will decide to attach him with the portal.

The readers deliberately referred to the pages of newspapers and news services will state that when becoming subscriber of this magazine, they will be up-to-date with the most important events. The reader may think that if so many people refer to this newspaper, maybe he should become a regular reader. Before ordering subscription he will read the article from this source only if another person point it to him. When becoming the subscriber he will be informed about the most important events personally and as the first. The deep links can become the motive power for higher sale of newspapers and news service ([1]).

4 The Links in Search Engines

It is common opinion that using deep links through search engines is permitted till the user needs these news for his own sake. It relates to the situation when the publisher popularizes his own information in an unlimited way. So it is rightly assumed that if the author of the work places it in the web, does not protect it with technological barrier and does not express any clause about 'not referring', he suggest that he does not disapprove of being in the searching results ([5]).

The solution of these arguments present are different in many European countries. In Austria and Germany in case of Meteodata and Paperboy decisions were made in favour of the web users. In Denmark courts forbade using deep links in press articles offered by similar portals. In Spain cases are in the progress ([6]). In Belgium courts imposed on Google removing from Google News quotations coming from newspapers in French language. The decision was made after taking legal steps by French and German publishers federated into Copiepresse. Their intention was to defend copyright that they had right to. They stressed that Google did not get permission for using in its services fragments of news published by them ([8, 2]).

After the verdict considering Google Belgian department of MSN got a letter from Copiepresse. The representatives of MSN met with Copiepresse lawyers to decide the rules that will enable MSN to publish articles. Negotiations concerning possibilities of using links to electronic editions of press titles are conducted. The publishers admit that search engines provide precious services but relations with them should be made formal. Maybe buying licence from publishers or different solution plays a role.

In February 2007 court supported the fact that Google broke national regulations that protect the copyright. The press publishers federated in Copiepresse which is representing 19 publishers of press published in Belgium in French and German language present arguments. Using fragments of texts in portals together with titles breaks publisher's copyright. In portals those fragments are available even when on main pages of the press they go to paid archives. As a result publishers sustain a loss. The publishers of press do not approve of mech-

anisms of searching because complete articles that got to archives can be found cache. ([9, 3]).

5 Summary

Contrary phenomenon can be observed in the web promotion. Some authors do not want to place links to them. Others hire specialists to popularize their web sites through building the link structure. The defence against links is seen especially in media where presented content is most often protected with the copyright. In other trades e-business enterprises pay huge sums to be on the highest position in searching results. It enables reaching wider range of clients.

When building a web site it is essential to have in mind users conventions and be aware of the way of searching the net. Suitable determinations visible or invisible are adequate to the ways in which search engines work. Searching is a method of navigation in the Web—the new user interface.

The owners of the search engines constantly expose themselves to expensive and time-consuming legal cases. Their results can not be entirely predicted. On the other hand, searching portals reach an agreement with the owners of copyright. They encouraging them to cooperation by sharing the profits from adverts. In the objection covering placing the links is some paradox because search engines generate traffic on companies‘ pages and lead clients to them. Many users could not know about the existence and the offer of these news services.

References

1. Battelle J.: *The Search, How Google and Its Rivals Rewrote the Rules of Business and Transformed Our Culture Portfolio,* Penguin Group (USA), Inc. (2005).
2. Copiepresse vs. Google Inc. Case No. A15346/DV 5 September 2006 Court of the First Instance Brussels (2006).
3. Copiepresse vs. Google Inc. Case No. 06/10/928/C 13 February 2007 Tribunal de Premiere Instance de Bruxelles (2007).
4. Gienias K.: *Linki—mozliwosci i odpowiedzialnosc,* Available `http://www.vagla.pl` Accessed (04.04.2007).
5. Polanski P. P.: *Intellectual Property Law versus Customs and Values of the Internet Community,* Proceedings of the 19th Bled eCommerce Conference “eValues”, Bled, Slovenia (2006).
6. Szpringer W.: *Prowadzenie dzialalnosci gospodarczej w Internecie,* Wydawnictwo Difin, Warszawa (2005).
7. Waglowski P.: *Prawo w sieci. Zarys regulacji Internetu,* Wydawnictwo Helion, Gliwice (2005).
8. Waglowski P.: *Chcemy ogladalnosci bez posrednikow—belgijskie orzeczenie przeciwko Google,* Available `http://www.vagla.pl` Accessed (04.04.2007) .
9. Waglowski P.: *Google przegralo w sporze z belgijskimi wydawcami prasy,* Available `http://www.vagla.pl` Accessed (04.04.2007) .

Proceedings of the International Multiconference on
Computer Science and Information Technology, pp. 565 – 578
ISSN 1896-7094
© 2007 PIPS

Application of stochastic processes in Internet survey

Elżbieta Getka-Wilczyńska

Warsaw School of Economics, Institute of Econometrics,
Division of Mathematical Statistics,
al. Niepodleglosci 164, 02-554 Warsaw, Poland,
Elzbieta.Getka-Wilczynska@sgh.waw.pl

Abstract. In this paper Poisson processes and basic methods of the reliability theory are proposed to interpretation, definition and analysis some stochastic properties of process of Internet data collection. At first, the notion of uncontrolled sample is introduced and random size of it is defined as a counting process. At the second, the process of Internet data collection is considered as a life test of the population surveyed. The events which appear in Internet survey are interpreted as a lifetime, arrival, death of the element of the population and the basic characteristics of reliability of the length of the population lifetime are described, calculated and estimated by using the notions and methods of the reliability theory.

Keywords: Internet survey, uncontrolled sample, population, coherent system, estimation, reliability function

1 Introduction

Recently statistical research as for as sampling selection can be divided into representative surveys based on the probability sample and surveys based on the non-probability sample, e.g. an Internet surveys. After choosing the kind of the sample selection next stage of the survey is data gathering. In all surveys, the data is collected using an immediate interview, a telephone interview or a post and in recent years an interview over the Internet. Difference between these researches rely on following aspects. In representative surveys based on the probability sample the frame of sampling is known, respondents are drawn to the sample by a statistician in accordance with sampling design (sampling scheme), the methods of theory sampling are applied to data analysis and in these surveys an electronic questionnaire is only one of modes of data collection. The Internet surveys have several advantages, such as low costs of collecting information, the speed of the data transmission and a

possibility to monitor it. The usage of the electronic questionnaire in Internet survey makes the interview more efficient, lowers the workload of the respondents and controls the responds' quality. But in the Internet surveys the frame of sampling usually does not exist and the populations surveyed are sets of unidentifiable elements (sometimes generally described), drawing the sample is not possible and respondents are not randomly selected to the sample, but they participate in the survey with a subjective decision. Moreover, in the Internet surveys data sets are collected in an uncontrolled way and the respondents who took part in the survey form - we introduce here a notion - an uncontrolled sample. The methods of the sampling theory can not be used for the data from such the samples because the probability inclusions are not known and statistics are calculated on the basis Internet data refer only to the population surveyed.

However (besides these drawbacks of Internet data), when the certain assumptions are made about the population it is possible to define the random size of the uncontrolled sample [2] and process of Internet data collection as a random experiment (or a life testing experiment) by using methods of stochastic processes [4] and the reliability theory [1, 5].

We generally assume, that Internet survey begins at the moment $t=0$, when the electronic questionnaire is put on the website and the survey is conducted for the time $T>0$. A set $\{u_1, u_2, \ldots\}$ denotes the population of potential respondents. For each $n \geq 1$, the population of n units is surveyed and the respondent sent the questionnaires independently. By τ_j, $j=1,2,\ldots,n$, $n \geq 1$, we denote a moment of questionnaire record on the server after an initial moment $t=0$, from each respondent $u_k, k=1,2,\ldots,n$, $n \geq 1$, belonging to the population of the size $n \geq 1$, who took part in the survey as j - th.

The moment of the questionnaire record is an event that can be interpreted as the moment of arrival or birth of the respondent, who took part in the survey as j -th, when the size of the uncontrolled sample is defined, as the moment of failure or renew of j -th element of the population, when the population is treated as a coherent system, as the waiting time for the j -th questionnaire record after the initial moment $t=0$ equal the length of lifetime of j -th element of the population until the moment $t \geq 0$ or the moment of death of the j -th element of the population, when the length of the population lifetime is considered.

Theoretically, four cases which describe the relation between the time of the survey conducting, $T>0$, and the size of the uncontrolled sample can be considered. In the first case, the registering the questionnaires ends at the moment $T>0$ specified in advance, independently of the questionnaires' number recorded. The size of the uncontrolled sample is then a random value in the interval $[0, T]$ and depends on

the length of time of the survey and on a selection procedure applied in the survey (if it is used in the survey). An extreme situation occurs when no data was collected (an arrival set is empty or the questionnaire, which arrived were rejected by the selection procedure used in the survey). In the second case the sample size is specified in advance and the survey ends when the assumed number of responses has arrived, independently of the length of time of the survey. An extreme situation occurs when the length of time of the survey is infinite. In the third case, both the length of time of the survey and the sample size are specified in advance and the survey ends in earlier of the assumed moments. In the fourth case, the final moment of the survey is not specified in advance. The process of registering questionnaires lasts at the moment when the collected data set meets the demands of the survey organizers.

2 The first conception -random size of an uncontrolled sample

If the process of Internet data collection is considered as a process of registering questionnaires on the server in a fixed interval of the time $T>0$ (the time of the survey conducting) then the size of the uncontrolled sample at the moment $t \geq 0$ equals the total number arrivals until the moment $t \geq 0$ and is defined as a counting process - Bernoulli, Poisson or composed Poisson process [4].

Case 1 - the size of the uncontrolled sample as Bernoulli process

Definition 2.1. For fixed $n \geq 1$ the size of uncontrolled sample until the moment $t \geq 0$ is given by

$$N(t)=\mathit{card}\left\{1 \leq k \leq n : \tau_k \in [0,t)\right\}$$

and is equal to a sum

$$N(t)=N_1(t)+\ldots+N_n(t),$$

where $N_k(t)=\begin{cases} 0 & \mathit{if}\ \ \tau_k \geq t \\ 1 & \mathit{if}\ \ \tau_k < t \end{cases}$, τ_k, $k=1,2,\ldots,n$, $n \geq 1$, are independent random variables with uniform distribution over $[0,T]$, $T>0$, $\tau_{k-1} \leq \tau_k$ for $j \geq 1$, $\tau_0=0$ and at the initial moment $t=0$ no arrivals occur.

The value of the random variable $N(t), t \geq 0$ equals the total number of arrivals until the moment $t \geq 0$ and the process $\{N(t), t \geq 0\}$ can be described

in a following way. Each of n respondents, independently of others send only one questionnaire with the probability 1 in the interval $[0, T]$ for $T > 0$ (the time of the survey conducted). The probability of sending the questionnaire by the certain respondent in the interval of the length $\Delta \subset [0, T]$ is equal to ratio $\frac{|\Delta|}{T}$. In this way, each respondent generates a stream consisted of only one arrival. A summary stream obtained by summing these streams is called a bound Bernoulli stream, that is, it consists of finite number of events. To complete the definition of the counting process it remains to compute the distribution of $N(t)$ and the joint distribution of $N(t_1), N(t_2), \ldots, N(t_n)$ for any non-negative $t_0, t_1, \ldots, t_n$.

1) Let $P_k(t) = P\{N(t) = k\}$, $k = 1, \ldots, n$, be the probability of event, that at the moment $t \geq 0$ the total number of arrivals $N(t)$ equals k. Since the probability of arrival of the given respondent in the interval $[0, t] \subset [0, T]$ is equal to ratio $\frac{t}{T}$ and the arrivals came independently, hence the total number of arrivals $N(t)$ at the moment $t \geq 0$ is random variable with Bernoulli distribution

$$P_k(t) = \binom{n}{k} \left(\frac{t}{T}\right)^k \left(1 - \frac{t}{T}\right)^{n-k}.$$

2) If the intervals $\Delta_1, \ldots, \Delta_n$ are disjoint pairs and the interval $[0, T] = \Delta_1 \cup \ldots \cup \Delta_n$ is a sum of $\Delta_1, \ldots, \Delta_n$, then for any non-negative integers $k_1, \ldots, k_n$ such that $k_1 + \ldots + k_n = n$ holds

$$P\{N(\Delta_1) = k_1, \ldots, N(\Delta_n) = k_n\} = \frac{n!}{k_1! \ldots k_n!} p_1^{k_1} \ldots p_n^{k_n},$$

where $N(\Delta_i)$ is the number of arrivals which occur in the interval Δ_i, $p_i = \frac{|\Delta_i|}{T}$ for $i = 1, \ldots, n$, and $|\Delta_i|$ is the length of the interval $\Delta_i = t_i - t_{i-1}$, $i = 1, \ldots, n$.

Case 2 - size of the uncontrolled sample as Poisson process

Definition 2.2. For fixed $n \geq 1$ the size of uncontrolled sample until the moment $t \geq 0$ is given by

$$N'(t)=\mathrm{card}\left\{1\le k\le n:\ \tau_k\in[0,t)\right\}=\max\left\{n\ge 0:\ S_n<t\right\},$$

where $\tau_1,\tau_2,\ldots$, denote as before the successive moments of questionnaires record, $\tau_{k-1}\le\tau_k$ for $k\ge 1$ and $\tau_0=0$, $\left(z_k\right)_{k=1}^{\infty}$ is a sequence of independent and identically distributed random variables $z_k=\tau_k-\tau_{k-1}$ with exponential distribution $G(t)=1-e^{-\lambda t}$, $t\ge 0$, $\lambda>0$ and $z_k=\tau_k-\tau_{k-1}$ for $k\ge 1$ denotes $k-$ th spacing between $k-$ th and $(k-1)-$ th arrivals, $S_n=\sum_{k=1}^{n} z_k$ is a random variable with Erlang distribution given by

$$P\left(S_n<t\right)=1-\sum_{i=0}^{n-1}\frac{(\lambda t)^i}{i!}e^{-\lambda t}\ \text{ for } t\ge 0 \text{ and } \lambda>0,$$

Then

$$P\left[N'(t)=k\right]=P\left[N'(t)<n+1\right]-P\left[N'(t)<n\right]=P\left[S_{n+1}\ge t\right]-P\left[S_n\ge t\right]=\frac{(\lambda t)^n}{n!}e^{-\lambda t},$$

$N'(t)$, $t\ge 0$, the total number of arrivals until the moment $t\ge 0$, is a random variable with Poisson distribution with the parameter $\lambda>0$ and $\left\{N'(t):t\ge 0\right\}$ is Poisson process.

Moreover, if in the Poisson process (stream) in the interval $[0,T]$, $T>0$, n arrivals occur, then process (stream) of arrivals in this interval is the Bernoulli process (stream), [5]. This fact is shown below.

If $0\le t\le T$ and $0\le k\le n$, then

$$P\left[N'(t)=k\,|\,N'(T)=n\right]=\frac{P\left[N'(t)=k,\,N'(T)-N'(t)=n-k\right]}{P\left[N'(t)=n\right]}$$

$$=\frac{P\left[N'(t)=k\right]P\left[N'(T-t)=n-k\right]}{P\left[N'(T)=n\right]}=\frac{\frac{(\lambda t)^k}{k!}e^{-\lambda t}\frac{\left(\lambda(T-t)\right)^{n-k}}{(n-k)!}e^{-\lambda(T-t)}}{\frac{(\lambda T)^n}{n!}e^{-\lambda T}}.$$

Hence $P\left\{N'(t)=k\,|\,N'(T)=n\right\}=\binom{n}{k}\left(\frac{t}{T}\right)^k\left(1-\frac{t}{T}\right)^{n-k}$.

If the intervals $\Delta_1,\ldots,\Delta_n$ are disjoint pairs and $[0,T]=\Delta_1\cup\ldots\cup\Delta_n$, then for any nonnegative integers $k_1,\ldots,k_n$ such that $k_1+\ldots+k_n=n$ holds

$$P\left\{N'\left(\Delta_1\right)=k_1,\ldots,N'\left(\Delta_n\right)=k_n \mid N'(T)=n\right\}=$$

$$=\frac{P_k\left\{N'\left(\Delta_1\right)=k_1,\ldots,N'\left(\Delta_n\right)=k_n\right\}}{P\left\{N'(T)=n\right\}}=\frac{\prod_{i=1}^{n}P\left\{N'\left(\Delta_i\right)=k_i\right\}}{P\left\{N'(T)=n\right\}}=$$

$$\frac{\prod_{i=1}^{n}\left(\lambda|\Delta_i|\right)^{k_i}e^{-\lambda|\Delta_i|}}{\frac{(\lambda T)^n}{n!}e^{-\lambda T}}.$$

Therefore $P\left\{N'\left(\Delta_1\right)=k_1,\ldots,N'\left(\Delta_n\right)=k_n \mid N'(T)=n\right\}=\frac{n!}{k_1!\ldots k_n!}p_1^{k_1}\ldots p_n^{k_n}$.

Case 3- size of the uncontrolled sample with a selection procedure as composed Poisson process

In the Internet surveys the electronic questionnaire is available to all Internet users. and a part of the registered arrivals came from respondents who do not necessarily belong to the surveyed population. In this case only the arrivals of these respondents whose questionnaires qualified for the data set based on the selection procedure are included in the sample. By this assumption and the assumptions made in case 2 the size of the uncontrolled sample is defined as a composed Poisson process.

Definition 2.3. For fixed $n \geq 1$ the size of uncontrolled sample until the moment $t \geq 0$ is given by $Y(t)=S_{N'(t)}$, where

$$N'(t)=\mathit{card}\left\{1\leq k\leq n:\ \tau_k\in[0,t)\right\}=\max\left\{n\geq 0:\ S_n<t\right\},$$

$$S_{N'(t)}=\sum_{j=1}^{N'(t)}U_j,$$

a sequence $\left(U_n\right)_{n=1}^{\infty}$ of independent and identically distributed random variables and the Poisson process $\left\{N'(t):t\geq 0\right\}$ are independent. The arrivals $\tau_n, n=1,2\ldots$, are selected for the uncontrolled sample in the following way (the sequence of arrivals $\tau_n, n=1,2\ldots$ is thinned): the arrival $\tau_n, n=1,2\ldots$., is omitted with the probability *p*, $p\in[0,1]$ (independently of the process taking place), if the respondent does not belong to the population and the arrival $\tau_n, n=1,2\ldots$., is left with the probability $1-p$, otherwise. The random variable U_i is equal to 1, if the arrival τ_i remains, and 0, if the arrival τ_i is

omitted. The probability *p*, $p \in [0,1]$ is defined by the procedure of selection used in the survey and consequently, process $\{Y(t), t \geq 0\}$ is composed Poisson process with expected number of the arrivals $\lambda(1-p)$.

3 The second conception - length of the population lifetime

In the remaining part of this paper we assume that the population of n units for $n \geq 1$, is treated as a finite coherent system of n components and the Internet survey begins at time $t = 0$ and it is conducted for the time $T, T > 0$. In this case the process of Internet data collection can be considered as a random experiment or a life testing experiment in which the basic characteristics of length of the population lifetime are analysed by using the methods of reliability theory [1], [5].
A non-negative random variables $\tau_k, k = 1,2\ldots, n,\ n \geq 1$ with distribution function

$$F_k(t) = P(\tau_k < t), \text{ for } t \geq 0,\ k = 1,2,\ldots, n,\ n \geq 1,$$

and probability density function

$$f(t) = F'(t) \quad \text{and} \quad F(t) = \int_0^x f(x)dx$$

are interpreted as length of lifetime of k -th element of the population until the moment $t \geq 0$ or the moment of death of k -th element of the population until the moment $t \geq 0$. The probability

$$\bar{F}_k(t) = 1 - F_k(t) \text{ for } t \geq 0,\ k = 1,2,\ldots, n,\ n \geq 1$$

is called the probability of the lifetime length of k -th element at least $t \geq 0$ or the probability of event that *k*-th respondent is in the state of life at least $t \geq 0$ or the reliability function of the length of *k*-th element lifetime at the moment $t \geq 0$, (the reliability of the *k*-th element in short).

The conditional probability density function

$$\lambda_k(t) = \frac{f(t)}{\bar{F}(t)} \text{ for } t \geq 0,\ k = 1,2,\ldots, n,\ n \geq 1.$$

is called rate of changes of *k*-th element of the population.

The elements of the population are not renewed - each record of questionnaires decreases the size of the population in one and the element which arrived is not replaced by a new one. This way of selection is called random sampling without replacement. Additionally, we assume that the length of the population lifetime of population of the size n for $n \geq 1$ at the initial moment $t = 0$ is equal to the sum of the length of the particular elements lifetimes. We define the length of the population lifetime by using the structure function of the population as follows.

States of elements of the population

The state of i -th elements of the population (as the system) is defined by the values of the binary function

$$e_i(t) = \begin{cases} 0 & \text{if i -th element is in the state of life or i -th element did not arrive until the moment } t \\ 1 & \text{if } i\text{-th element is in the state of death or i -th element arrive until the moment } t \end{cases}$$

Then the state of all elements of the population of size n , for $n \geq 1$, is determined by n-dimension vector $e(t) = \left[e_1(t), e_2(t), \ldots, e_n(t) \right]$ and we assume that at the initial moment $t = 0$ all elements of the population of size n , for $n \geq 1$, are in the states of life. This assumption means that at the moment $t = 0$ no arrivals occurred.

States of the population

The state of the population of size n , for $n \geq 1$, at the moment $t \geq 0$ is defined by the values of the binary function

$$\phi_0(t) = \begin{cases} 0 & \text{if the population is in state of life (the survey (test) is conducted) in the moment t} \\ 1 & \text{if the popualtion is in the state of death (the survey (test) ended) until the moment t} \end{cases}$$

and at each moment $t \geq 0$ it depends on the states of the elements through the values of the function $\phi_0(t) = \phi\left[e_1(t), \ldots, e_n(t) \right] = \phi\left(e(t) \right)$.

In the process of Internet data collection treated as a life test of the population of size n , for $n \geq 1$, the population can be found at the moment $t \geq 0$ in the state of life during the conducting of the survey in following cases. In the first case, at the

moment $0 \le t \le T$, where T is the time of the survey conducting specified in advance and the number of death (arrivals) in interval of the length $t \ge 0$ is random but it is less than the size of the population. Otherwise, until the moment T specified in advance. In the second case, until the moment $t \ge 0$, in which the number of death (arrivals) is equal to the size of the sample specified in advance (it is less than the size of the population) and the time of the survey conducting, T, is not specified in advance. In the third case, until the earlier of the time of the survey conducting, T, and the moment $t \ge 0$, when the number of death (arrivals) is equal to the sample size, where both the length of time of the survey, T, and the sample size are specified in advance. In the fourth case, until the moment $t \ge 0$, when the collected data set (it can be a subset of the population or the population surveyed) meets the demands of the survey organizers and the final moment of the survey is not specified in advance.

Properties of the structure function

The structure function $\phi(e)$ is increasing, if for any two vectors e i e' is satisfied the condition:

$$\text{if } e < e' \text{, then } \phi(e) \le \phi(e'),$$

where $e < e'$, if for all $i = 1, \ldots, n$, $e_i \le e_i'$.

This property of the structure function introduce a partial order in a set of the binary vectors and means that additional death of the element can not change the state of the population from the state of death to the state of life.

The function $\phi(e)$ defines a division of a set $E = \{e\}$ of all *n* -dimension and binary vectors which describe the state of the population to two sets: $E_+ = \{e : \phi(e) = 0\}$, a set of states of population life and $E_- = \{e : \phi(e) = 1\}$, a set of states of population death. If the structure function is increasing, then the division of the set E to two sets E_+ and E_- is called a monotonic structure.

Length of the population lifetime

Let us denote by τ the length of the population lifetime and $\tau = \inf\{t : \phi[e(t)] = 1\}$. Then $F(t) = P(\tau < t)$ is the probability of ending of the survey (test) until the moment $t \ge 0$ or the probability of the event that the population is in the state of death until the moment $t \ge 0$ and $\overline{F}(t) = P(\tau \ge t)$ is

the probability of the conducting survey (test) at least $t \geq 0$, the probability of the event that the population is in the state of life at least $t \geq 0$ or the reliability function of the length of the population lifetime at the moment $t \geq 0$, (the reliability of the population in short).

4 Calculation and estimate of the reliability of the length of the population lifetime

The formula which expresses the relation between the reliability of the length of the population lifetime and reliabilities of elements at the moment $t \geq 0$ is given by

$$\bar{F}(t) = \sum_{e \in E_+} p(e) \quad ,$$

where $p(e) = \prod_{k=1}^{n} \bar{F}_k^{1-e_k}(t) F_k^{e_k}(t)$, (we adopt the convention $0^0 = 1$) is a probability of event that the population is in the state e .

If $\tau_k, k = 1,2 \ldots, n,\ n \geq 1$, are non-negative independent random variables, the elements are not renewed and the function $\phi(e)$ is increasing, then the reliability function of the length of the population lifetime $\bar{F}(t)$ is increasing respectively to each coordinate of the reliability function of the length of the element lifetime $\bar{F}_k(t)$.

Thus an upper or a lower bound on the reliability of the length of the population lifetime may be obtained from the upper or lower bounds on the reliabilities of the elements. When the number of the states is large (the number of all states is equal to 2^n) and the function $\phi(e)$ is very complicated, then a formulae given above is not efficient and the other methods of calculation are applied e.g. the method of path and cut, the recurrence method of Markov chain or generally, the Markov methods.

Length of the population lifetime for the series structure

The population (as a system)) of n - elements is called a series structure, when the population is in the state of life if and only if each element is in the state of life. In this case, the change of the state of any element causes the change of the population state. The length of the population lifetime in this case is equal to the waiting time of the first death and the size of the uncontrolled sample equals zero for the first death. Then the basic characteristics of the reliability function of the series structure are

given as follows. The probability of the length of the population lifetime (duration of the survey) at least $t \geq 0$ is equal to $\overline{F}(t) = \prod_{i=1}^{n} \overline{F}_i(t)$. From inequality [3]

$$1-\sum_{i=1}^{n} F_i \leq \prod_{i=1}^{n} (1-F_i) \leq 1-\sum_{i=1}^{n} F_i + \sum_{i<j} F_i F_j \leq 1-\sum_{i=1}^{n} F_i + \frac{1}{2}\left(\sum_{i=1}^{n} F_i\right)^2$$

it follows that $|F(t)-\sum_{i=1}^{n} \overline{F}_i| \leq \frac{1}{2}\left(\sum_{i=1}^{n} F_i\right)^2$.

The change rate of the population equals the sum of the change rate of the elements

$$\lambda(t) = -\frac{\overline{F}'(t)}{\overline{F}(t)} = -\left[\ln \overline{F}(t)\right]' = -\sum_{i=1}^{n}\left[\ln \overline{F}_i(t)\right] = -\sum_{i=1}^{n} \frac{\overline{F}_i'(t)}{\overline{F}_i(t)} = \sum_{i=1}^{n} \lambda_i(t)$$

The expected time of the length of the population lifetime is equal to

$$T = E(\tau) = \int_0^{\infty} \overline{F}(t)dt .$$

Length of the population lifetime for the parallel structure

The population (as a system) of n - elements is called a parallel structure, when the population is in the state of death if and only if all elements are in the state of death. In this case, the change of the state of the population (death of the population) takes places only if changes of all population elements occur - all elements of the population died and the size of the uncontrolled sample is equal to the size of the population (all elements of the populations arrived). Then the probability of the length of the population lifetime (duration of the survey) at least $t \geq 0$ is equal to $F(t) = \prod_{i=1}^{n} F_i(t)$ or $F(t) = F_0^n(t)$ and the expected time of the length of the population lifetime is equal to $T = E(\tau) = \int_0^{\infty} [1-F(t)]dt$ or $T = \int_0^{\infty} \left[1-F_0^n(t)\right]dt$, when $F_i(t) = F_0(t),\ i = 1,2,\ldots,n$.

Length of the population lifetime for the partial structure

The population (as a system) of n - elements is called a partial structure if all elements of the population are identical and the population is in the state of life if at least m elements of the population are in the state of life (that is, at most $n-m$ death occur) and the size of the uncontrolled sample equals $n-m$. Then the reliability function of the length of the population lifetime is equal to $\overline{F}(t)=\sum_{k=m}^{n}\binom{n}{k}\overline{F}_0^k F_0^{n-k}$.

The method of path and cut

In this method we define notions of minimal path and minimal cut that used to estimate the reliability function of the length of the population lifetime, [1].

Definition 4.1. The set of elements $A=\{i_1,\ldots,i_n\}$ of the population is called a minimal path if all the elements of this set are in the state of life (the population is in the state of life, the survey is being conducted) and no subset of the set A has this property.

From the monotonic property of the structure function, the set $A=\{i_1,\ldots,i_k\}$ is a minimal path if and only if $e\in E_+$, where e is a vector in which coordinates $i_1,\ldots,i_k$ take on the value zero and the remaining coordinates take on the value one, with any state greater than e belonging to E_- . Therefore, every minimal path determines a bordering state of the life of the population in which the occur of death of any element causes a change of the state of the population into the state of death one (ending of the survey (test)). In term of the size of the uncontrolled sample, it means, that the number of arrivals is equal to the number of elements of the minimal path is smaller per one than the size of the sample assumed in the survey.

Let $\{A_1,A_2,\ldots,A_m\}$ be a sets of all minimal paths with the corresponding bordering states $e^{(1)},e^{(2)},\ldots,e^{(m)}$. A_s is an event in which all elements of the minimal path A_s are in the state of life. Since $E_+=\bigcup_{s=1}^{m}A_s$, [5], the reliability function of the length of the population lifetime is calculated from the formula

$$\overline{F}(t)=P\left(\bigcup_{s=1}^{m}A_s\right)=\sum_{i=1}^{m}P(A_i)+\sum_{i,j}P(A_iA_j)+\sum_{i<j<k}P(A_iA_jA_k)+$$
$$+\ldots+(-1)^{m+1}P(A_1A_2\ldots A_m)$$

The number of the elements of the sum on the right is equal to 2^m-1 and the probability of any event which is given by $A_{i_1}A_{i_2}\ldots A_{i_k}$, where $i_1<i_2,\ldots<i_k$ is equal to

$$P\left(A_{i_1}A_{i_2}\ldots A_{i_k}\right)=\bar{F}_{s_1}(t)\bar{F}_{s_2}(t)\ldots.\bar{F}_{s_l}(t)\ ,$$

where s_1, s_2 ,. s_l are different indices of elements of the minimal paths (that is, in the case of the elements belonging to overlapping parts of different paths, each element is calculated only once). In the order to lower the number of calculation we introduce the following notions. The two minimal paths are called crossing, when they have at least one common element. The two minimal paths are called relevant, if there exists a chain of crossing paths which connects them. The relevant relation is the equivalent relation which divides a set of all minimal paths into classes of relevant minimal paths. Let $\left\{A_1\ldots A_{k_1}\right\},\left\{A_{k_1+1}\ldots A_{k_2}\right\}\ldots.$ be the successive classes of the relevant minimal paths. Because $F(t)=1-\bar{F}(t)=P\left(\bigcap_{i=1}^{m}\bar{A}_i\right)$ and the events belonging to different classes are independent, then $F(t)=P\left(\bigcap_{i=1}^{k_1}\bar{A}_i\right)P\left(\bigcap_{i=1}^{k_2}\bar{A}_{k_1+1}\right).....$.

A dual notion of the minimal path is a minimal cut (a critical set).

Definition 4.2. A set of elements $B=\left(j_1,j_2,\ldots,j_l\right)$ is called a minimal cut, if all elements of this set are in the state of death (the population is in the state of death, the survey (test) ended) and no subset of the set B has this property.

In this case we are interesting in those cut set in which the number of elements is equal to the size of the uncontrolled sample specified in advance (all elements belonging to the set B arrived).

If we denote by $\left\{B_1,B_2,\ldots,B_s\right\}$ a set of all the minimal cuts, then the probability of an event that the survey (test) ends until the moment $t\geq 0$ is equal to

$$F(t)=P\left(\bigcup_{i=1}^{s}B_i\right)=\sum_{i=1}^{s}P\left(B_i\right)+\sum_{i<j}P\left(B_iB_j\right)+\sum_{i<j<k}P\left(B_iB_jB_k\right)+\ldots+\ .$$

$$+(-1)^{m+1}P\left(B_1B_2\ldots B_m\right)=S_1-S_2+S_3+\ldots+(-1)^{s+1}S_s\ ,$$

From this formula we can obtain the estimation of the length of the population lifetime (during the survey) as the estimation of the reliability function of the population. In this case the number of elements in the successive minimal cuts is interpreted as the possible sample sizes which can be collected in the survey on

condition that the survey ends after collection of the sample of the assumed size. From proof of this formula for the non-crossing minimal cuts holds $S_2 \leq \frac{S_1^2}{2}$, $S_3 \leq \frac{S_1^3}{6}$, and so on, and in the case of the crossing minimal cuts the partial sums maintain an order $S_k = O\left(S_1^2\right)$.

Moreover, for any k, $S_1 - S_2 + \ldots - S_{2k} \leq F(t) \leq S_1 - S_2 + \ldots - S_{2k} + S_{2k+1}$. Thus there exists the possibility of the estimation of the length of the population lifetime with assumed precision because partial sums on the right of the last formula are the interchangeable upper and the lower bounds of the reliability function.

Conlusions

The process of Internet data collection is interpreted and analysed as a life test of population surveyed by using the notions and methods of the reliability theory. A random set of respondents who participate in Internet survey is called the uncontrolled sample and defined as the counting process by using Poisson processes. The proposed approach allows to study some stochastic properties of the process of the Internet data collection, the calculation and the estimate of the basic characteristics by the assumed assumptions.

References

1. Barlow R. E., Proschan F.: *Mathematical theory of reliability,* Wiley and Sons, Inc. New York, London, Sydney (1965).
2. Getka-Wilczyńska E.: *Random properties non-controlled sample,* Annals of Collegium of Economics Analysis, 13, p. 59-69, Warsaw School of Economics, (in Polish), Warsaw (2004).
3. Hardy G. H., Littlewood J. E., Polya G.: *Inequalities,* Cambridge University Press, Cambridge (1934)
4. Kingman J. F. C.: *Poissons' processes,* Polish Scientific Publishers, (in Polish), Warsaw (2001).
5. Sołowiew A. D.: *Analytic methods in reliability theory,* Technical - Scientific Publishers, (in Polish), Warsaw (1983).

Proceedings of the International Multiconference on Computer Science and Information Technology pp. 579–588
ISSN 1896-7094
© 2007 PIPS

Data model standardization for real-time e-commerce *

Przemysław Kacprzak, Mariusz Kaleta, Piotr Pałka, Kamil Smolira, Eugeniusz Toczyłowski, and Tomasz Traczyk

Warsaw University of Technology, Institute of Control and Computation Engineering, 00-665 Warsaw, Nowowiejska 15/19, POLAND,
P.Kacprzak@elka.pw.edu.pl, M.Kaleta@elka.pw.edu.pl, P.Palka@ia.pw.edu.pl, K.Smolira@elka.pw.edu.pl, E.Toczylowski@ia.pw.edu.pl, T.Traczyk@ia.pw.edu.pl

Abstract. Real-time activity is important trend rising in e-commerce. Such kind of activity requires specialized, fully standardized and integrated market systems. In this paper we present and discuss basic requirements for real-time e-commerce systems. We also present M^3 – Open Multi-commodity Market Data Model, which may provide flexible and universal market data and communication models for wide range of markets, and thereby facilitate systems integration for real-time e-commerce purposes.

1 Introduction

Nowadays we observe continuous development and broaden application area of e-commerce. Companies want to realize more and more new individual targets using the Internet and thereby e-commerce. One of important trends in modern economy is to operate according to the *real-time requirements*, this means that some tasks have to be realized in guaranteed and as short as possible time. Under such circumstances commodities and services delivered or accomplished too late or in improper amount become worthless. Good example of such activity is company making short series of untypical custom-made equipment. Such enterprise has to ensure that all necessary materials will be delivered in proper amount and time, what usually means in the same time. Moreover company has to prepare its own resources and productive power to be available in adequate amount and time. If such a company has its own dependent deliverers these requirements may be not so difficult to meet, but if company operates on the free market and has to buy all components from independent entities this task become quite challenging and may require some special mechanisms to ensure success. These special mechanisms may be provided by creating special kind of

* The research was co-financed by the Ministry of Science and Higher Education as the research project PBZ-MEiN-1/2/2006 "Energetic safety of the country"; and as the Science Network "New, ecological and safe technologies for energy generation and conversion", decision no. 60/E-87/SN/0056/2007.

market – *real-time market*, which should enable market entities to meet real-time requirements.

There exists a special group of commodities – non-storable goods, like electric energy [2], telecommunication bandwidth [5], transportation services [3], which require real-time market to ensure completely free and fluent trade. These goods have to be exactly balanced in each time moment and all unused resources physically disappear. In such a market entities trade by integrals of some time functions (e.g. function of electric power in time). In case of other commodities real-time market is not indispensable, but in some cases may be used to increase profits, better adjust to demand on the market and improve trade fluency. On the other hand "standard" commodities traded in the real-time market gain some features of non-storable goods. They are localized in time and space, become worthless if are delivered in improper time; storage may be very expensive and can not last too long. Moreover, delivery and other constraints (eg. limited transport capabilities) become much more critical.

2 Multicommodity real-time eMarket

Rise of *Business To Business – B2B* relations significance in e-commerce caused establishment of eMarketplaces [1]. They are trade platforms, which group in one place many customers and suppliers, and thereby allow to much more efficient offers matching, than in the case of bilateral trade, when suppliers and customers have to search each other in the whole Internet. EMarketplaces enable also to partially automate processes of transaction conclusion, best offers searching and demand examination.

Standard eMarketplaces work fine in traditional e-commerce, but real-time markets cause new demands. One of the most important problems is necessity of synchronization of deliveries from various suppliers. Buyer has to have a guarantee that he gets all needed commodities in proper amount and time. Very often some goods are complementary, what means that one good without the other is worthless for customer (eg. all raw materials necessary to produce final product), therefore market entity has to get all needed commodities in certain amount proportions and in the same time. In some cases there is also additional demand of exact goods balance without storage possibilities. All these demands may be easily met using the multicommodity turnover model [12], which allows market entities to trade in bundled packages of goods.

On real-time markets offers may be submitted or changed in short time before their realization time. A possibility has to exist to react for these events within adequate time, therefore all market procedures should be automated and take place without necessity of direct human interference. All operation should be performed by inteligent agents, whose targets and parameters are defined by a human.

Traditional e-commerce mechanisms are insufficient for real-time markets, therefore we propose idea of *Multicommodity real-time eMarket – RTeMarket*. Such a market should enable market entities to trade respecting real-time re-

quirements. This is enabled by possibility of complex commodities definition (in particular commodities may be precisely located in time and space), possibility of submitting complex multicommodity offers for bundled packages of goods, standardized data and communication model, which enables system integration and trade process automation, possibility of precise entities structure and infrastructure description. We try to facilitate meeting this requirements by developing an Open Multi-commodity Market Data Model – M^3 [7], which allows to integrate market data from multiple distributed sources in diverse systems and normalize market procedures and data modeling form market entity as well as from market processes point of view.

At present, there are no general world-wide standards for market information interchange. In some industries there are data interchange mechanism, which can be acknowledged as local standards, for instance, RosettaNet standards in electronic industry, MDDL (Market Data Definition Language) in financial sector and other industries standards specified on the basis of open standards like ebXML (Electronic Business using eXtensible Markup Language) or XBRL (eXtensible Business Reporting Language) [14–16]. However, these standards are focused on electronic communication of business and financial data like invoices, offers, business partner information and so on. In energy sector – a typical infrastructure market – UCTE (Union for the Coordination of Transmission of Electricity) initiate initiative called Electronic Highway focuses only on transport layer and technical aspects of the communication network. It is clear that existing standards can not meet the needs of the real-time markets, where many specific elements, related to balancing of many commodities and services under constraints and other real-rime requirements, may play important roles. Heterogenous systems and absence of general mechanism for data interchange create barriers for mechanism integration and developments that are especially important for the European and world-wide market evolutions.

Apart from above mentioned requirements, real-time market demands also appropriately designed market process structure as well as balancing and pricing algorithms, but these problems are not subject of this paper. We only consider if using M^3 we may model all data necessary for these processes and ensure proper communication between them.

3 Systems integration for RTeMarkets

Efficient usage of real-time e-commerce requires standardized data exchange model and possibility of various systems integration. Market entity may trade on many platforms, changing them dynamically and even create new platforms and become their operator to realize its own targets. Market entity should have possibility to perform these changes freely, fluently and in some sense transparently. Therefore processes of offers submitting and searching, as well as transaction conclusion should be standardized not only within one platform but for all platforms in given market segment. These standardization should ensure that any rules' change in one platform will be invisible for market entity, which still

should have possibility to efficiently operate on many trade platforms without changing anything in its software. Standardization and integration should also enable synchronization of trading using many platforms. Market entity should have possibility to buy or sell goods simultaneously in many platforms retaining guarantee of trade synchronization, balance and goods complementarity. All this requirements tends toward usage of inteligent automate agents, which should try to realize market entity targets in best possible way using simultaneously many trade platforms. Appropriate data model standardization may also allow to operate brokers, which group some eMarkets and are visible from outside as one "standard" trade platform.

3.1 M^3 approach

Although market balancing procedures on particular trade platforms may significantly vary, in case of practical used mechanisms there is usually possibility of determination some commonly used data and communication schemas. Proposed Open Multi-commodity Market Data Model (M^3) is flexible and universal solution, which allows to represent data concerning broad area of market system in standardized way, and thereby to easily integrate various trade platforms. M^3 treats elementary market balancing processes as a "black boxes" and take care only of their inputs and outputs, so it is independent from detailed market rules, as it was depicted on Fig. 1.

M^3 consists of several layers: formal mathematical model [7], conceptual data model, expressed in form of UML class diagrams [7, 6], exemplary relational database structure, XML schemas for static data [10], communication models [8, 9] and XML schemas for messages and Web Services definitions.

Conceptual data model, described in details in [7], consist of the following elements.

Infrastructure (network) – representing technical infrastructure required for the trade (eg. transportation network). During the balancing process, the infrastructure plays the role of a system of limited resources, and is a medium for delivery of commodities and services

Market entities structure – describing market players and relations between them. Market entities form a hierarchy, where given market entity may be composed of some other market entities, e.g. a corporation and subsidiaries.

Time structure – defining time periods used in balancing processes and relations between them. Well defined time structure is used to localize commodities in time and to organize market processes.

Commodities – describing goods traded on the market, which have specific type, parameters, and may be related to time structure and infrastructure

Offers – providing three types of offers: simple, integrated and grouping. Simple offer is described by admissible range of commodity volumes and a unit price. Integrated offer is a typical offer for multi-commodity turnover, where players trade with packages (or bundles) of commodities with fixed proportions of commodities in the offer. The most complex type of offers are grouping

offers. Grouping offer aggregate a set of other simple or integrated offers and describes relation between these offers. Grouping offers allow the market entities to define individual constraints.

Results – describing current schedule of all commodities' delivery, which is called programme, and market prices.

Dependencies between M^3 data model elements and example of their possible usage on the electric power market was depicted on Fig. 2.

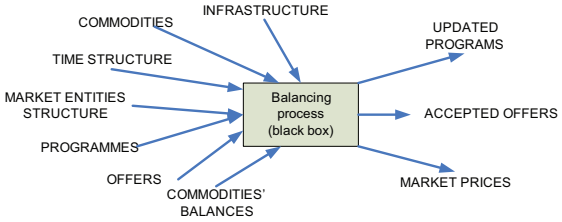


Fig. 1. Market balancing schema.

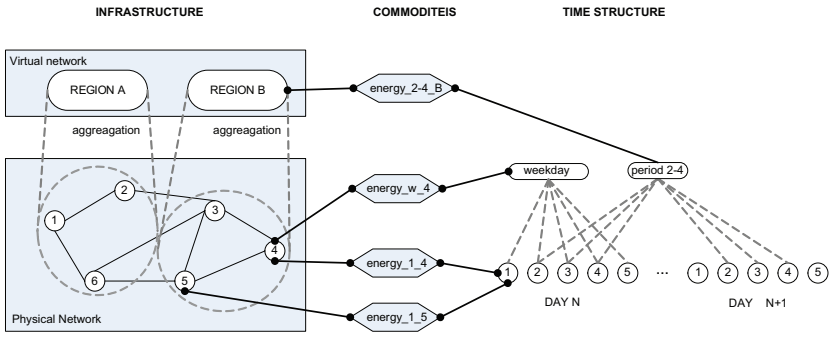


Fig. 2. M^3 elements dependencies and usage.

M^3 provides universal interface between market player and all trade platforms and balancing processes, as well as between various trade platforms. Moreover it may be used in data modeling and communication processes within one trade platform, e.g. in case of multistage balancing procedures. Apart from usage on real-world markets, M^3 may be used during research works to facilitate results obtained by different teams comparison as well as to facilitate simultaneous work within research team.

In M^3 data may be stored using XML files or a database. Information interchanged between market entities is to be expressed in XML, as a set of XML messages. All XML structures used in M^3 are defined using W3C XML Schema.

XML structures for static data [10] cover whole data model, and are designed mainly for data interchange between scientists and developers, who develop and test algorithms for balancing and optimizing turnover on multi-commodity markets. These structures may also be used – instead of a database – to store market data. XML messages, used in communication on the market, are composed of the same elements and types as used in the static data model. Messages schemas import necessary simple and complex types definitions, and global element declarations from static data schemas, and enclose them with an envelope containing communication details. Each message contains only those parts of static structures, which are necessary in particular data exchange process the message is designed for.

M^3 supports two topologies of communication on the market.

Centralized – where market participants communicate with one central entity, e.g. balancing operator. In this model, direct data interchange between participants is of course possible, but it is not covered by M^3 messages [8].

Distributed – where no central unit exists, and market participants directly communicate with each other using M^3 messages [9].

There exist, with respect to changeability, two categories of data. Current data, like offers, programmes, etc. – describing temporary market state; and metadata, like commodity kinds, market entities, networks, calendar, etc. – defining static objects appearing on the market, along with their identifiers. In both topologies, two models of message passing exist.

Directional – message is sent to one recipient or to explicitly listed group of recipients. This is a proper model for passing current market data, e.g. submitting offers, and for registering changes in metadata, e.g. introducing new entities or commodities to the central repository. In this model, each request should be individually replied. This communication model can be implemented by Simple Object Access Protocol (SOAP).

Broadcast – message is published to the public or to some group of entities (not explicitly listed). This model is appropriate for publishing market metadata, e.g. directories of market participants, catalogues of commodities, and for publishing results of balancing processes. This type of communication could be implemented by placing messages in some public repository (e.g. on public Web site), where every interested entity could withdraw them, or by constant polling a server by some client, more precisely by a software agent. The polling could be performed through SOAP protocol.

Messages designed for M^3 communication can be interchanged by any means, like ftp, e-mail, etc., but are especially suitable for use with SOAP and Web services. This is especially important in case of real-time market, where entities have to react immediately for incoming messages (eg. submitted offers or balancing results). Web services mechanism provides also immediate response for each request, which ensures sender that his message has been received by addressee and may inform if the message was accepted due to formal reasons (eg. if it is

in proper format). Immediate response and confirmation is very important in real-time markets, because allows market entity to continuous control of market processes and to react immediately to various events including communication problems and formal mistakes.

4 M^3 applications in real-time e-commerce

4.1 E-supply chains

E-supply chains [13] arise by using the Internet and e-commerce mechanisms to classical supply chains modeling all entities, activities, information, processes and resources involved in moving a product or a service from supplier to customer [11]. Modern technologies allow to integrate and automate all processes of supply chain management in all enterprisers involved into the chain. As it was shown, in real-time e-commerce entities very often need to define their demand in a complex way, therefore they have to use complex e-supply chains, which may model all real-time requirements. Market entities have to define commodities parameters, time and localization of delivery and dependencies between complementary goods. In our example of company making short series of untypical custom-made equipment, this entity has to submit offer which, in case of acceptance, ensure timely delivery of all needed materials in proper amount proportions, even if they are provided by different suppliers.

Assume that the company gets order query for an electronic detector in amount at least 100 pieces, and not more than 150, with price 350 for one piece, delivery time 10 days and response time 1 day. To realize such order company needs: 1 printed circuit-board, 2 programable units, 1 detector, 5 resistors and 6 capacitors for each pice of final product; 4 days in advance. Example of M^3 offer describing such situation is presented bellow.

```
<m3:Offer id="ex:o23565-78" offeredPrice="300.00">
  <m3:description>Offer for raw materials</m3:description>
  <m3:offerStatus status="m3:offer-open">
      <m3:durationPeriod endTime="2007-07-21T10:15:00"
                         startTime="2007-07-20T10:15:00" />
       </m3:offerStatus>
  <m3:offeredBy ref="ex:SmallCompany"/>
  <m3:volumeRange minValue="100" maxValue="150"/>
  <m3:BundledOffer>
   <m3:offeredCommodity shareFactor="1" ref="ex:circuit-board"/>
   <m3:offeredCommodity shareFactor="2" ref="ex:prog-unit"/>
   <m3:offeredCommodity shareFactor="1" ref="ex:detector"/>
   <m3:offeredCommodity shareFactor="5" ref="ex:resistor5K"/>
   <m3:offeredCommodity shareFactor="6" ref="ex:capacitor15u"/>
  </m3:BundledOffer>
</m3:Offer>
```

Offer is valid through one day (from 2007-07-20 to 2007-07-21), because after that time company has to give a response to the query. Such offer assure that, in case of acceptance, company gets at least 100 and not more than 150 "packages" of goods containing: 1 printed circuit-board, 2 programable units, 1 detector, 5 resistors and 6 capacitors each, and pay for each "package" not more than 300. Of course each commodity may become from different supplier, from company point of view only total balance is important. Time and place of delivery, as well as commodities parameters, may be defined by appropriate commodity definition, described in the next section.

The company submits such offer in RTeMarket and, if offer is accepted, the company may give a positive response to the order. The company may try to find necessary materials using one trade platform or many independent trade platforms. M^3 mechanisms (two phase commit protocol [4]) allows to ensure complete offer realization even in case of distributed trade environments and many eMarketplaces [9].

In case of fully integrated systems such checking of realizability may be performed automatically. Human need only to translate order query to needed raw materials, define maximum price and deadline for delivery, whereas automatic agent should check their accessibility and order them.

4.2 Commodities

M^3 allows complex commodities definition. Commodities may have many parameters which determine precisely requirements toward them. In particular they can be easily and unequivocally localized in time and space. In our example, for each needed material, the company has to define parameters, as well as time and place of delivery:

```
<m3:Commodity id="resistor5K" dref="ex:resistor"
  minBalance="0" maxBalance="0">
  <m3:description>
    resistor 5K delivered to Smallcompany on 2007-07-26
  </m3:description>
  <m3:parameter dref="ex:resistance>5000</m3:parameter>
  <m3:availableAt ref="ex:SmallCompanyLocation"/>
 <m3:CalendarScheduledCommodity ref="ex:D070726"/>
</m3:Commodity>
```

Needed commodity has to be type *ex:resistor* and have value of parameter *ex:resistance* equal 5000. Commodity must be available in *ex:SmallCompanyLocation* on period *ex:D070726* (2007-07-26). Definitions of time period – *ex:D070726* and localization – *ex:SmallCompanyLocation* have to be placed in time structure and network element respectively (for details see [8, 10]).

Possibility of complex offer and commodities definition is even more crucial in case of non-storable goods, like electric energy or telecommunication bandwidth.

Commodities on such markets has many various parameters and are strictly connected with infrastructure and time, which has to be precisely mapped in market model. More detailed samples of offers and commodities definition in M^3 may be found in [8] and on the project Web page [10].

4.3 Limited infrastructure

Infrastructure limiting trade possibilities is typical to some kinds of markets and usually connected to non-storable goods. Although in case of real-time markets limited infrastructure may play essential role also in case of storable goods. Typically role of the infrastructure may be played by limited in time delivery possibilities. For example, limited number of rail cars in specific place and time or limited space on railway track. If there is no delivery time limitation or this period is long enough, these constraints may be neglected (cars may be transported from other places, transport may be divided or shifted in time), but in case of real-time markets these constraints have to be respected and modeled.

In M^3 , limited infrastructure may be modeled using networks or grouping offers. Networks may reflect diverse constraints, mainly limited delivery possibilities, but also maximum amount of given commodity (telecommunication bandwidth), some dependencies between goods or possibilities of transfer of one good into another, etc. Networks are used to model general constraints concerning a whole market, so called *system constraints*. Virtual networks, which may aggregate physical networks give possibility to model constraints on various levels and constraints concerning various commodities. Grouping offers may be used to model market entities' individual constraints, e.g. limited speed of generation change for a power plant, maximum or minimum power generation in an electrical network node, etc. Detailed examples of infrastructure modeling may be found in [8] and on the project Web page [10].

5 Summary

Trends toward standardization and systems integration, caused by tendency to operate in real-time, become more and more important in modern e-commerce. Development of the real-time markets cause new requirements towards e-commerce mechanism and procedures. M^3 is a universal and flexible solution, which may significantly facilitate realization of this trends, providing common data and communication models. In contrast to existing standards for information interchange, M^3 is not limited to the particular type of market, and my be used on the broad range of various market systems. Existing local information exchange standards concentrate mainly on business and financial data, like invoices, orders, payments, formal information about entities and so on, whereas M^3 focus on data concerning directly trade process, market balancing and pricing. Thanks to complex commodities, market entities and time structure; possibility of exact infrastructure mapping; and support for multi-commodity turnover M^3 may become very useful tool for real-time markets, where such solutions are lacking.

Important feature of M^3 is possibility of application in multilateral distributed environments, what may significantly increase its practical applications area in e-commerce.

References

1. Bakos J. Y.: *Reducing buyer search costs: Implications for electronic marketplaces,* Manage. Sci., vol. 43, pp. 1676, 1997.
2. Baldick R., Helman U., Hobbs B. F., and O'Neill R. P.: *Design of efficient generation markets,* In *Proceedings of the IEEE*, number 11, pages 1998–2012, 2005.
3. Borndörfer R., Grötschel M., Lukac S., Mitusch K., Schlechte T., Schultz S., Tanner A.: *An Auctioning Approach to Railway Slot Allocation,* ZIB-Report 05-45 (2005).
4. Brockmeyer R. L. et al: *Extension of two phase commit protocol to distributed participants,* U. S. Patent, 5546582, 1996.
5. Courcoubetis, C., Weber, R., Coe, M.: *Pricing Communication Networks: Economics, Technology and Modelling,* John Willey & Sons, Ltd. (2003).
6. Kacprzak P., Kaleta M., Pałka P., Smolira K., Toczyłowski E., Traczyk T.: *M^3 — Data Model for Open Multi-commodity Turnover System, "Bazy danych. Nowe technologie.", tom 2 "Bezpieczeństwo, wybrane technologie i zastosowania",* ss. 289–300. WKiŁ Warszawa 2007. (in Polish).
7. Kacprzak P., Kaleta M., Pałka P., Smolira K., Toczyłowski E., Traczyk T.: *Open Multi-commodity Market Data Model for Network Systems.* Institute of Control and Computation Engineering, Technical Report Warsaw.
8. Kacprzak P., Kaleta M., Pałka P., Smolira K., Toczyłowski E., Traczyk T.: *Communication model for M^3 —Open Multi-commodity Market Data Model,* Proceedings of II Krajowa Konferencja Naukowa Technologie Przetwarzania Danych TPD'2007 (in press).
9. Kacprzak P., Kaleta M., Pałka P., Smolira K., Toczyłowski E., Traczyk T.: *Modeling distributed multilateral markets using Multi-commodity Market Model,* Proceedings of 28th International Scientific School information systems architecture and technology ISAT'07 (in press.)
10. Multi-commodity Market Model (M^3) Project Web page. `http://www.openM3.org`
11. Nagurney A.: *Supply Chain Network Economics: Dynamics of Prices, Flows, and Profits*, Edward Elgar Publishing, 2006
12. E. Toczyłowski. *Optimization of Market Processes under Constraints.* EXIT Publishing Company, Warsaw, 2002. (in Polish).
13. Viswanadham N., Gaonkar R.: *Understaning e-supllay chains*, Reseach Paper, `http://www.isb.edu/facultydetails/Viswanatham/Download/Vis.WP6.pdf`
14. ebXML website: `http://www.ebxml.org`.
15. MDDL website: `http://www.mddl.org`.
16. RosettaNet organization website: `http://www.rosettanet.org`.

Proceedings of the International Multiconference on
Computer Science and Information Technology, pp. 589 – 596
ISSN 1896-7094
© 2007 PIPS

A purchasing power indicator as a tool for predicting market attractiveness for SMEs in a changing currency environment

Jacek Wachowicz

Gdańsk University of Technology, Poland, Management and Economics Department,
ul. G. Narutowicza 11/12, 80-952 Gdańsk, Poland
jacek.wachowicz@zie.pg.gda.pl

Abstract. E-commerce gave SMEs possibility to act over the international markets. However, this introduced some new threats to business operations. Apart of technological, knowledge, workforce, increased competition they need to face changes of exchange rates and conjuncture fluctuations. This paper presents a concept of simple indicators, which may very quick select markets of a great potential due to financial market changes.

Introduction

Since its introduction, Web technologies have changed business strategies. They increased opportunities in doing business over the borders and created new economical markets. Surrounding conditions however may threat future international market development. During last few years markets observed massive changes in currency exchange rates, which caused extreme pressure in international trade efficiency. As we see the situation is getting less and less stable, the urgent need for creating an easy and complex factor allowing to determine in which direction market situation is changing, which might help determining profitability in international trade. As one can observe relative appreciation of some currencies and depreciation of others, usually there is only information of currency changes against only the local currency. This doesn't give any information on changes between other currencies, which may mislead in future prognosis (just like changes in parities between PLN/USD and PLN/EUR might be often far more influenced by changing USD/EUR parity rather then by a condition of Polish economy). In such environment no information is given allowing quick understanding, which markets are getting cheaper (more attractive as import sources) and which are getting more expensive (more attractive as export destinations). As a solution is proposed a complex index (Monetary Factor MF), which should show which currency is getting stronger (appreciates) against most of other currencies and which is getting weaker (depreciates) against most of other currencies – and shows relative changes of purchasing power for a currency under examination. In particular this shall ease understanding if the local currency appreciates or depreciates in case of different currency shifts (for instance in case of appreciation against USD and depreciation against EUR). This might turn much easier strategic financial planning in companies targeting international markets.

E-commerce revolution

Introduction of http:// protocol together with first graphical Web browser initiated unexpected introduction of Internet as a new business media, which begun a couple years later a true revolution in business. It gave possibility of worldwide communication at almost no cost. This drove creation of new digital markets with more and more services and goods offered through Internet. It allowed introduction of many digital substitutes (like digital music, movies, maps etc.) and creation of brand new markets (like information browsing, digital content or TCP/IP bandwidth delivery, VoIP services and many others). Computers and Internet brought many companies to become largest players of world's stock exchanges – just to mention corporations like Microsoft, Sun, Cisco, Dell, Google, Yahoo, eBay or may others.

On the other hand Internet created a huge demand on individual services and products. This gave a huge opportunities for Small and Medium Enterprises. Special favor for SMEs gives typical demand for highly-professional and individual products and services, which was driven by Internet. On the other hand ease of communication and digitalization of goods drove Internet markets to become more ad more international. This is always regarded to be a huge benefit of including this new channel into business strategy. Unfortunately it brings quite a lot of threats on the other hand. First of all, entering the global internet markets expose companies to international competition. This is supposed to force intensed price pressure. It force increased flexibility in terms of product specification and required time to delivery. Moreover it expose companies to threats of international culture, legal and tax differences. Internet and e-commerce raised a lot of problems in these fields and many of them seems to be either not solved or very difficult to judge, especially for SMEs, companies that can't hire specialized lawyers for searching answers to difficult, international legal aspects.

One can find some more fields of potential threats on the electronic business frontier. Especially difficult seem to be the ones that generate extra costs. This might be considered safety problems (due to virus or hacker attacks and hardware failures), knowledge problems (caused by lack of skilled workforce), and things far beyond of companies control – like currency exchange rates. The last one may be reduced by using proposed in this paper model of currency weight.

SME adoption

Small and Medium Enterprises are a very special sector of economy. They employ most of workforce worldwide. Their great feature is ability to adapt to changing economical environment. This turns them and – because of their number - the whole economy far more flexible and adaptable. This feature is as well a major force making SMEs to grow. But on the other hand SMEs are very limited in their activities. The most commonly pointed limitations are modest financial resources, dependency on owner's knowledge and initiative, inability in creation of big research projects. It makes introduction of cutting edge new technologies very difficult and slower (at the average) then in large companies.

Adoption of internet technologies into SMEs strategies and operations turned them to be far more dependant to changes in global economy. This affects SMEs – and as

statistics show, they are trying to expand exports activities, in which e-commerce activities may be of help. Following statistics may confirm that (Tables 1-3).

Table 1: Exports vs. GNP 2000–2006 (goods and services)

	2000	**2001**	**2002**	**2003**	**2004**	**2005**	**2006**
GDP dynamics [%]	**4,0**	**1,0**	**1,4**	**3,8**	**5,3**	**3,4**	**6,1**
GDP [bln PLN]	684,9	721,6	781,1	814,7	923,3	980,9	1057,7
Exports [bln PLN]	201,5	211	231	280,8	346,4	364,5	426,8
Exports/GDP [%]	**29,4**	**29,2**	**29,6**	**34,5**	**37,5**	**37,2**	**40,4**

Source: Przedsiębiorczość w Polsce [3]

Table 2: Export factors

employment	2000	2001	2002	2003	2004	2005	2006
Percentage of export incomes (%)							
10–49	3,0	4,0	6,0	7,3	8,0	8,0	8,0
50–249	6,0	7,0	9,0	11,9	12,0	12,0	13,0
>249	14,0	14,0	16,0	18,9	20,0	21,0	23,0
Wholesale incomes dynamice (%)							
10–49	108,7	101,8	98,2	100,3	116,7	101,1	109,3
50–249	115,0	97,9	99,2	109,4	118,0	101,8	111,5
>249	112,3	102,1	103,0	109,7	115,6	105,9	115,1
Export sales dynamice(%)							
10–49	83,7	109,4	209,8	121,6	107,8	92,0	115,7
50–249	140,3	102,9	144,5	143,9	112,2	103,6	114,4
>249	121,7	100,5	120,1	126,1	130,2	107,0	121,2

Source: Przedsiębiorczość w Polsce [3]

Table 3: Barriers In Developement for SMEs

Frequency of problems' observations In period:	Currency exchange rates, including their changes	Lack of qualified workforce	Resources and materiale prices growth	regulationsChanges In taxes and	High competition	Low demand	High salaries, salary pressure	Payment problems	Lack of liquidity
2Q2005	**26,8**	7,4	4,5	11,4	10,4	7,2	6,7	X	X
3Q2005	19,8	3,6	5,4	**13,6**	**12,6**	7,3	**7,6**	X	X
4Q2005	20,1	11,1	7,7	11,3	11,2	7,0	4,8	X	X
1Q2006	21,4	1,8	9,0	6,8	10,6	**10,0**	5,3	5,3	X
2Q2006	18,9	3,0	6,4	7,0	9,8	9,7	3,5	5,3	4,6
3Q2006	16,6	5,3	**10,5**	8,6	9,9	8,9	3,6	**6,1**	3,5
4Q2006	11,2	10,1	10,1	7,0	8,2	7,0	4,3	5,9	3,8
1Q2007	15,2	**10,7**	9,8	8,5	9,0	5,9	**5,2**	4,9	**4,9**
2Q2007	10,7	**10,7**	9,7	7,6	7,2	5,3	5,1	4,1	3,0

Source: Przedsiębiorczość w Polsce [3]

Model

As changes in currency markets turned to be significant and more rapid, the necessity for easy understanding of their nature and consequences for the international sales arises. Presented model should allow easy and fast determination of changing terms of trade, which is necessary for a medium to long-term planning of operations.
For building a model we need to explain some basic assumptions. It should give a quick answer if a currency gains or looses purchase strength.
Let us count a general base of currencies:

$$B_c = \sum_i M_i \cdot e_{i,c} \qquad \textit{(Equation 1)}$$

where c is the currency used for calculation, B_c – is a monetary base counted in currency c, M_i is monetary measure for the i-th currency, $e_{i,c}$ is an exchange rate between i-th currency and currency c.

$$MF_x = \frac{M_x \cdot e_{x,c}}{B_c} \qquad \textit{(Equation 2)}$$

Where MF_x is searched Monetary Factor, showing a purchasing power, for the currency x.
For having Monetary Factor calculated we need to choose a base currency c, which needs to have data on exchange rates of the base currency to other currencies, including the one that is of interest.
For calculation we need to have data on monetary base as well. Commonly used monetary aggregates are M0, M1, M2, M3. Their definitions are: [1]

- M0 is the most strict one and contains only money in circulation (i.e. excluding cash held by banks)
- M1 comprises money in circulation (i.e. excluding cash held by banks) and demand deposits of the non-financial sector, which covers state enterprises, private enterprises and households.
- M2 comprises monetary aggregate (M1), time and saving deposits in zloty, saving bonds, and foreign currency deposits of the non-financial sector, which covers state enterprises, private enterprises and households.
- M3 comprises monetary aggregate M2, plus repurchase agreements and debt securities issued with maturity up to 2 years.

In most countries definitions and methodologies of M0 and M1 aggregates are similar; bigger differences may be seen within M2 and M3 aggregates.
Dynamic changes over markets may be observed in two natural points of view – relative and absolute. The second one is easier to measure, as it is enough to compare changes in monetary base growth throughout countries of interest. Of course this has to be done in one preferred currency, which is exchangeable to all other currencies, for instance USD. Then we have to compare changes. This might be done using equation 3.

$$A_{o,c} = M_o \cdot e_{o,c} \qquad \textit{(Equation 3)}$$

where o is the currency observed, $A_{o,c}$ – is a monetary base of observed currency o counted in currency c, M_o is monetary measure for the currency o, $e_{o,c}$ is an exchange rate between currency o and a base currency c
The relative growth may be calculated as a ratio of Monetary Factors of currencies x and o. This is illustrated on equation 4:

$$R_{o,x} = \frac{MF_x(n)/MF_x(n-1)}{MF_o(n)/MF_o(n-1)} \qquad \textit{(Equation 4)}$$

Outputs

For practical calculations we need to gain primary data. As it was already mentioned, the most comparable monetary base data is M0 and M1. We shall use the broader one – M1. The data is quoted by United Nations for years 2004 and 2005. The exchange between them and USD shall be quoted by OECD. Because of the space limitations, in tables is presented data for countries with biggest M1 measures – and for Poland.

Table 4: Monetary measure M1 in selected countries

Country	M1 (2004)	M1 (2005)
Australia	2,4395E+11	2,689E+11
Canada	3,9571E+11	4,1635E+11
China	9,5815E+12	1,069E+13
France	3,5747E+11	3,9603E+11
Germany	6,4654E+11	7,1624E+11
Italy	5,4625E+11	5,9056E+11
Japan	4,1241E+14	4,3629E+14
Netherlands	1,5848E+11	1,7875E+11
Spain	2,2399E+11	4,4068E+11
Switzerland	2,2406E+11	2,3856E+11
United States	6,7809E+11	6,3552E+11
Poland	1,461E+11	1,7483E+11

Source: Unated Nations Statistics Division [4]

Table 5: Exchange rates in selected countries

Country	currency /USD (2004)	Currency /USD (2005)
Australia	1,359233	1,312783
Canada	1,301092	1,211708
China	8,276801	8,194317
France	0,8048583	0,804625
Germany	0,8048583	0,804625
Italy	0,8048583	0,804625
Japan	108,1469	110,0971
Netherlands	0,8048583	0,804625
Spain	0,8048583	0,804625
Switzerland	1,242742	1,245942
United States	1	1
Poland	3,650867	3,234025

Source: OECD Main Economic Indicators [2]

As a next step the comparable monetary bases shall be calculated. This shall be calculated in USD:

Table 6: Comparable monetary measure M1 in selected countries

Country	M1 (USD,2004)	M1 (USD,2005)
Australia	1,7948E+11	2,0484E+11
Canada	3,0413E+11	3,4361E+11
China	1,1576E+12	1,3046E+12
France	4,4414E+11	4,922E+11
Germany	8,0329E+11	8,9015E+11
Italy	6,7869E+11	7,3396E+11
Japan	3,8134E+12	3,9628E+12
Netherlands	1,9691E+11	2,2216E+11
Spain	2,783E+11	5,4768E+11
Switzerland	1,8029E+11	1,9147E+11
United States	6,7809E+11	6,3552E+11
Poland	4,0017E+10	5,4061E+10

Then we can calculate Monetary Factors for all currencies. This was done using only 6 currencies with biggest monetary base (narrow Monetary Factor – nMF) and using 22 currencies (broad Monetary Factor - bMF). As we can see, they are very coparable.

Table 7: Monetary Factors for selected countries

Country	nMF (2004)	bMF (2004)	NMF (2005)	bMF (2005)
Australia	0,02517	0,01847	0,02721	0,01911
Canada	0,04265	0,03129	0,04565	0,03206
China	0,16234	0,11911	0,17332	0,12173
France	0,06228	0,04570	0,06539	0,04593
Germany	0,11265	0,08265	0,11826	0,08306
Italy	0,09517	0,06983	0,09751	0,06848
Japan	0,53476	0,39237	0,52648	0,36976
Netherlands	0,02761	0,02026	0,02951	0,02073
Spain	0,03903	0,02864	0,07276	0,05110
Switzerland	0,02528	0,01855	0,02544	0,01787
United States	0,09509	0,06977	0,08443	0,05930
Poland	0,00561	0,00412	0,00718	0,00504

Having Monetary Factors calculated we may search for the currency, which gained most purchasing ability. As it was pointed above, we may search for the currency that gained most in an absolute measures, or which gained greatest relative growth. Finally we may use the relative factor to check which currencies are relatively faster gaining purchasing power comparing to the one of our interest. As an illustration, in the third column one may find such factor calculated for Polish Zloty (PLN). Factor grater than 1 means that the observed currency was gaining purchasing power faster then Polish Zloty – surprisingly the only one currency that was gaining it faster then

Polish Zloty was Euro issued in Spain, which may indicate that Spain may be very interesting market to enter, as there should be a lot of money searching to be spent. On the other hand, the greatest absolute gains of purchasing power were observed in Spain (again) and China and Japan. A very tight monetary situation might have been observed in United States, which might suggest a tough market to win contracts (due to shrinking purchasing power over the market).

Table 8: Monetary indicators for selected countries

Country	x(2005) -x (2004)	MF(2005)/ MF(2004)	[MF(2005)/ MF(2004)] / [PLN(2005)/ PLN(2004)]
Australia	2,5359E+10	1,03498063	0,84481797
Canada	3,9473E+10	1,02454686	0,83630125
China	1,4696E+11	1,02197235	0,83419977
France	4,8062E+10	1,0049824	0,82033147
Germany	8,6862E+10	1,0049088	0,8202714
Italy	5,5267E+10	0,98069503	0,80050656
Japan	1,494E+11	0,94237591	0,76922802
Nethe rlands	2,5253E+10	1,02314995	0,83516101
Spain	2,6938E+11	1,78463155	1,45673142
Switzerland	1,1181E+10	0,96308872	0,78613515
United St ates	-4,2573E+10	0,8499136	0,69375432
Poland	1,4043E+10	1,22509306	1

Conclusion.

Presented Monetary Factors calculation may be a simple tool for quick international market review in terms of searching for the best possible markets for gaining international contracts. They allow easy observation of currency purchasing power changes.This should be taken into account in phase of planning the whole e-commerce strategy, especially in a SMEs sector, as they cannot use a very complex analytical tools for international situation observations and prognosis.

References

1. OECD Statistics Portal, *Main Economic Indicators – Sources and definitions, accessed 12 Sep 2007,* `http://stats.oecd.org/mei/default.asp?lang=e&subject=14`
2. OECD Statistics Portal, Main Economic Indicators `http://stats.oecd.org/mei/default.asp`, accessed 9 Sep 2007.
3. *Przedsiębiorczość w Polsce,* Report of Ministry of economy. Warsaw 2007.
4. *United Nations Statistics Division,* International Monetary Fund International Financial Statistics, accessed 23 Aug 2007, `http://unstats.un.org/unsd/cdb/cdb_series_xrxx.asp?series_code=6470`

Proceedings of the International Multiconference on
Computer Science and Information Technology, pp. 597 – 606
ISSN 1896-7094
© 2007 PIPS

Even Swaps Method for Developing Assessment Capabilities of E-Negotiation System

Tomasz Wachowicz

The Karol Adamiecki University of Economics in Katowice
Department of Operations Research
ul. Bogucicka 14, 40-227 Katowice, Poland
wachowic@ae.katowice.pl

Abstract. One of the features of the e-negotiation systems is the capability to support negotiators in evaluating and comparing the offers. It is usually conducted by means of an additive scoring system, which results in the abstract scores assigned to all the offers. However the process of assigning the scores to the issues and options, required by an additive scoring system, may be perceived by some decision makers as an affected and vague. In the paper we consider thus an alternative approach that basis on the even swaps method. It is a part of multiple attribute decision making methodology called PrOACT, proposed by Hammond, Keeney and Raiffa, and focuses on finding the equivalent amounts as the balances between the unit of one issue with respect to the units of the others. The method is adopted to the negotiation actuality and programmed in a spreadsheet as a prototype software.

1. Introduction

E-Negotiation Systems (ENSs) usually have various capabilities. They can facilitate communication, support decision process, aid concession making and agreement preparation, evaluate and select strategies and tactics, give an access to expert knowledge etc. Taking into account the overall role they play in negotiation process they may be divided into three classes [8]: passive systems, active facilitative-mediation systems and proactive intervention-mediation systems. The passive ENSs facilitate communication and provides users with simple data presentation. They neither process data nor give any help on the merits of the case. The active facilitative-mediation systems give support to both the negotiators and the process. They allow to formulate the problem, define the structures of preferences, construct and evaluate the offers and analyze them with some formal methods. The proactive intervention-mediation systems have the same capabilities as the active ones but also allow to explore the negotiation knowledge. They analyze negotiation history and try to predict the future deriving from external knowledge bases. They may maintain the negotiation protocol and predict the counterpart strategies and tactics.

As we can see all but the passive systems are equipped with some analytic tools that allow at least for the offers evaluation and their comparison according to the negotiators' preferences. This kind of support comprises the assessment capability of

the ENS, which is very important for analyzing multi-issue negotiations. If we browse through some widely known and well described in literature ENSs [2, 10, 14, 15], we find that to accomplish the assessment capability of the ENS an additive scoring system is usually implemented. An additive scoring system [6, 7, 11] is a simple mathematical tool that allows for the offers evaluation by means of single aggregated criterion. It derives from utility theory and requires the problem decomposition and evaluation of issues and resolution levels within the issues respectively. The evaluation is conducted by assigning the subjective scores that reflect the decision maker's preferences for each possible resolutions levels of each issue. The scores are then aggregated to obtain a single score for each offer. The substantial advantage of an additive scoring system is that it never allows the offers to be incomparable. However, it requires of the negotiators some analytical skills and a quantitative way of thinking. Some negotiators may feel uncomfortable with this approach since the scores that they need to assign are in fact the abstract values of not clear explanation. Furthermore some researchers emphasize that these numerical scores are simply assigned to the issues and options instead of calculating them basing on the negotiators verbal evaluation [3, 13]. Hence the combination of additive scoring system with other decision support tools are proposed [17].

Another approach to the offers evaluation can be derived from multiple attribute decision making and basis on a concept of outranking [12]. It requires evaluation of the issue importance and analyzing the dominance between the offers. Each pair of offers is described then with a score that reflects a strength of domination, which is a sum of weights of issues, for which one offer dominates another. It allows to construct the offers hierarchy that help to find whether one offer is better or worse then the other one but – which is a disadvantage of this approach – it does not give the information of how much it is better/worse.

There are as well some other analytic tools for negotiation support based on more advanced methods like multi-attribute linear programming [1, 9] or game theory [5, 16] designed mostly for facilitating the mediators or arbitrators. They are rarely applied to ENSs since either represent the negotiation problem in too generalized way or are to complicated to use them without any external help of the analysts.

Taking into consideration all the above we propose a new approach to the offers comparison and evaluation. It bases on the even swaps method, which is a tool for multiple attribute decision making connected with a more general PrOACT approach. In this paper we show the foundations of this method, its principles and adaptation to the negotiation actuality. The even swaps method is going to be implemented in educational ENS that is being constructed by the author of this paper. Since the ENS has not been working tool yet, in the paper we show only the prototype implementation of the even swaps programmed in a spreadsheet and simulating only the assessment capability of the future ENS.

2. PrOACT And Even Swaps In Multiple Criteria Decision Making

The even swaps method is a straightforward approach to multiple criteria decision making. It requires a decision problem to be well structured and therefore is applied as a part of the PrOACT methodology [4, 7]. PrOACT divides any decision situation

into the five elements: problem, objectives, alternatives consequences and tradeoffs and then reassemble the results of their analysis into the right choice[1]. The algorithm of the analysis requires:

1. *Defining problem.*
   The decision problem needs to be well stated with no unwarranted assumptions. It is a key issue of decision process since the analysis of wrong stated problem results usually with a wrong decision.
2. *Specifying objectives*.
   Objectives are the decision maker's criteria that it is going to use to evaluate each decision. It is crucial to find a right criteria at this stage of process. Adding new to the decision process or removing unnecessary ones later (during tradeoff analysis) may be troublesome and require some tiring calculations.
3. *Creating alternatives*.
   Alternatives reflect the possible decisions that can be made to solve the problem. The carefulness is required for discovering the possible courses of action and creating a new alternatives since the final decision would not be better then the best identified alternative.
4. *Understanding consequences*.
   Having identified all the feasible alternatives the decision maker needs to evaluate how well do they satisfy its objectives. Therefore for each of the alternatives the vector of payoffs has been constructed, which consists of the quantitative or qualitative description of the alternative's consequences with respect to each decision criterion separately. It is necessary to compare then the alternatives and make the final decision. The comparison is usually made using the table of consequences, which consists of the vectors of payoffs.
5. *Making tradeoffs*.
   Decision maker eliminates from the dominated alternatives. It rarely leads to the identification of sole best alternative, since the objectives are usually conflicting and while comparing any two alternatives we often realize that former one is better then the latter one on some objectives, but worse on others. To chose the best alternative the tradeoff analysis is required. It allows to find the balance between each pair of alternatives by comparing the quality of their reciprocal surpluses on some objectives and shortages on others.

The five above elements of PrOACT allow to structure the decision problem and are the starting point to implement the even swaps method. The even swaps derive mainly from the concept of the vector domination, which says that if alternative A is better then alternative B on some objectives and no worse then B on all other objectives then we can consider that alternative A dominates B. Each dominated alternative can be eliminated from the decision process since it is in total worse then at least one alternative from the set of all feasible ones. If all the alternatives result in equal consequence for any objective it can be also ignored in the decision making process.

The even swaps method tries to adjust then the consequences of all considered alternatives to make them equivalent in terms of a given objective. It requires an increasing the value of an alternative in terms of one objective while decreasing its value by an equivalent amount in terms of another objective. The equivalent amounts

1 The authors of PrOACT approach call the decision to be smart, wise or right, which obviously means it is an optimal one with respect to the decision maker structure of preferences.

are defined subjectively by the decision maker and reflect its structure of preferences. After such an adjustment the considered objective can be eliminated and domination analysis can be conducted once again with respect to the rest of objectives. The process of the objectives elimination is repeated until the decision maker is able to find the best decision (the one that dominates all the others). The even swaps method applied in the PrOACT approach consists then in alternately assessment of the dominance that enables to eliminate alternatives and elimination of the objectives to make the dominance assessment possible on the next stage of decision process.

There is one substantial advantage of even swaps in comparison with other methods of alternatives comparison. Even swaps allow to compare virtually every two alternatives and consider one to be better, worse or equal to another, while some other methods (i.e. outranking) fail to find them incomparable. What is more, the even swaps method uses the true criteria to compare the alternatives instead of analyzing the abstract scores (like utilities, desirability etc.), which make the decision analysis more intuitive and easy to decision maker and let us to believe that it may facilitate better the decision process in any decision support tool including the ENS.

3. Negotiation Support With Even Swaps

Structuring negotiations according to the PrOACT requirements may be perceived as a little troublesome but it is concordant with the analysis of the negotiation case that is being conducted during the pre-negotiation phase. In this phase parties hold the initial talks, formulate the negotiation subject and agree the negotiation set by suggesting the issues and their resolution levels, which allow to construct the table of consequences. Let us consider a simple example of business negotiation with four issues: price, time of payment, time of delivery and returns conditions. After four steps of the PrOACT analysis we obtain a consequences table the part of which is shown in Table 1.

Table 1. Table of consequences for business negotiations

Objectives	**Offers**			
	1	**2**	**3**	**4**
Price	$40	$37	$41	$45
Time of payment	10 days	Upon delivery	10 days	60 days
Time of delivery	3 weeks	5 weeks	4 weeks	4 weeks
Returns	Above 5% of spoilage	Above 10% of spoilage	Above 15% of spoilage	Above 2% of spoilage
Score	70	55	45	60

The table consists of four selected offers (for real-word negotiations it may consists of hundreds) described by the resolution levels assured for all the issues considered. The resolution levels can be interpreted as the offers consequences but applying traditional additive scoring system we can obtain a single payoff for each offer. The last row of Table 1 shows the score that was calculated on the basis of the buyer party preferences. With domination analysis (and scores excluded) we can eliminate from the consequences table the offer number 3, since it results with the payoffs no better then the offer 1. However that is all what the domination analysis allows for. Thus the

three remaining offers are considered incomparable. The situation becomes far more better when we include the aggregating scores into our comparison. We see then clearly, that the offer number 1 is the best negotiation offer. The second best is the offer 4 with the score of 60. The difference between these two offers is of 10, but the problem is how this difference can be interpreted. May we use the ratio scale to say that the offer number 1 is 1,17 times better (1,17 = 70 / 60) then the offer number 4? Is the difference of 10 worth of fighting and spoiling the parties relationship?

We can avoid such problems by using the even swaps method. To compare the offers we need to ask negotiator about the tradeoff values. Let us assume that at the first stage of analysis we will compare the issue of price with time of payment. Negotiator needs to assess then what change in price will compensate any change in time of payment. The problem is to set the basis of this comparison, because depending on the basis value of time of payment, we receive different amounts of equivalent prices. We recommend use the best resolution levels of the issues that are going to be set equal, for we can easily interpret the changes then as the additional costs we need to bear to make any considered offer best.

Negotiator will use then as the basis the best value of 60 days of delay in payment. It starts with comparing offer 2 with offer 4 and needs to find how many dollars it is going pay more for the contract to have the payment delayed of 60 days. Let us assume that the negotiator decided to pay $6 for 60 days. Comparing the offer 1 with offer 2 we could assume consequently that for the delay of 50 days the negotiator is going to pay $5 but it does not have to be true since the preferences usually are not distributed linearly. Assume that the negotiator decided to pay $6 for 60 days difference because it receives this amount as interest from 2 months bank deposit. If the difference is 50 days it is enough only for 1 month deposit with gain of $3. The tradeoff then will be of $3 instead of $5.

Table 2 . Making even swaps between price and time of payment

Issues/ Objectives	**Offers**			
	1	**2**	**3**	**4**
Price	~~$40~~ $45	~~$37~~ $43	~~$41~~ $46	$45
Time of payment	~~10 days~~ 60 days	~~Upon delivery~~ 60 days	~~10 days~~ 60 days	60 days
Time of delivery	3 weeks	5 weeks	4 weeks	4 weeks
Returns	Above 5% of spoilage	Above 10% of spoilage	Above 15% of spoilage	Above 2% of spoilage

To simplify our case we assume the linear distribution of preferences. We recalculate now the consequences from Table 1 to render them equivalent in terms of time of payment. The results we obtain are shown in Table 2.

According to the even swaps procedure we can eliminate from the further analysis the issue of time of payment, since it results now in equal consequences for every offer under consideration. The new table of consequences is shown in Table 3.

Table 3. Table of consequences after making the first swap

Issues/ Objectives	Offers 1	2	3	4
Price	\$45	\$43	\$46	\$45
Time of delivery	3 weeks	5 weeks	4 weeks	4 weeks
Returns	Above 5% of spoilage	Above 10% of spoilage	Above 15% of spoilage	Above 2% of spoilage

Let us assume that negotiator has compared the rest of the issues and defined the following swaps (with linear distribution): \$1 of price for 2 weeks of delay in delivery and \$1 of price for 3% of spoilage. After completing the tradeoff analysis between price and time of delivery we obtain the table of consequences as shown in Table 4. Please notice that to make the issue of time of delivery to be irrelevant (with equal consequences for all offers) we have chosen the best resolution level of 3 weeks.

Table 4. Making even swaps between price and time of delivery

Issues/ Objectives	Offers 1	2	3	4
Price	\$45	~~\$43~~ \$44	~~\$46~~ \$46,5	~~\$45~~ \$45,5
Time of delivery	3 weeks	~~5 weeks~~ 3 weeks	~~4 weeks~~ 3 weeks	~~4 weeks~~ 3 weeks
Returns	Above 5% of spoilage	Above 10% of spoilage	Above 15% of spoilage	Above 2% of spoilage

We remove now the issue of time of delivery from the further analysis and make the tradeoffs for two last issues: price and returns conditions, taking as a base the best result of the return at the spoilage of above 2%. After making this swap we obtain the table of consequences described with only one objective – the price. To show the effect of even swaps we return to full initial table of consequences and add in the last row of the table the overall scores of offers in terms of price (see Table 5).

Table 5 . Table of consequences with the overall score of in terms of price after

Objectives	Offers 1	2	3	4
Price	\$40	\$37	\$41	\$45
Time of payment	10 days	Upon delivery	10 days	60 days
Time of delivery	3 weeks	5 weeks	4 weeks	4 weeks
Returns	Above 5% of spoilage	Above 10% of spoilage	Above 15% of spoilage	Above 2% of spoilage
Score in terms of price	\$46	\$46,66	\$50,83	\$45,5

The negotiator can compare now the offers looking at the score in terms of price and choose the best one, which is the offer number 4. It can easily interpret the score since it is not the abstract value as desirability or utility. The score shows the true

price of each offer while assuming that it fulfills all the other objectives at the highest possible level. Deriving from the negotiator's equivalent amounts of swaps we can easily recalculate the table of consequences to present it with the overall score in terms of all other objectives. Such a presentation can help the negotiator to make a final decision since the differences in the overall prices are in fact very small.

5. Implementing Even Swaps In ENS

The above concept of applying the even swaps method into the negotiation support is being implemented into an ENS, the configuration of which comprises of:

- the web-based core acting as the data base storing information about the negotiation case structure, offers and messages exchanged and the negotiators' individual preferences and
- individual NSSs working as the Excel Add-In, exchanging data with the web-based core, supporting negotiators in PrOACT procedures, conducting even swaps and offers analyzing.

The key elements of the client-side NSS have already been programmed, including the problem definition, identifying the objectives and their resolution levels that are typical for the vast majority of ENSs. All characteristic elements of even swaps have been programmed as the Even Swaps Creator, including definition of the pairs of issues for declaring the swaps (Fig. 1.), evaluation of the equivalent values for each swap (Fig. 2.) and defining the issue to calculate the overall score (Fig. 3.).

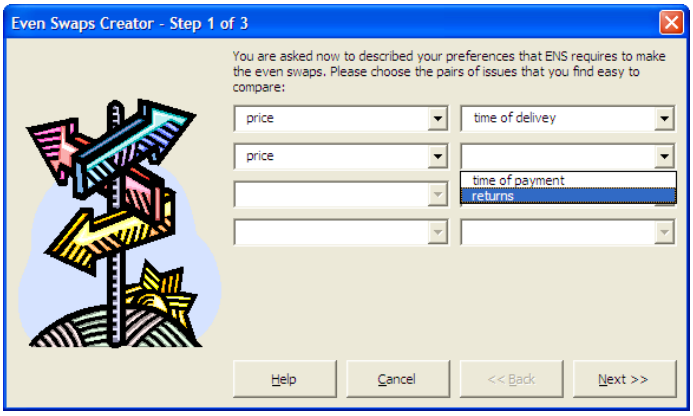


Fig. 1. Defining the pairs of issues to compare for even swaps evaluation

The Even Swaps Creator accomplishes the steps of the analysis we proposed in the section 4 and allows to structure the negotiator's preferences. The preferences describe all essential swaps and allow to compare the offers during the negotiation process. To make the comparison possible in the sense of overall scores the system forces the offers to be the compete packages that consist of all the issues with one feasible resolution level each. An example design of the offer exchange panel with the even swaps based scores is shown in Figure 4.

Further works over the ENS include developing the communication capability of the system by introducing the message panel, designing some graphical elements for data visualization (i.e. history graphs), programming the web-based core and it's connections with the negotiators' Excel Add-Ins. Diversification of evaluation methods is considered as well by introducing some classic approaches based on the concept of utility with more advanced extensions like AHP [17].

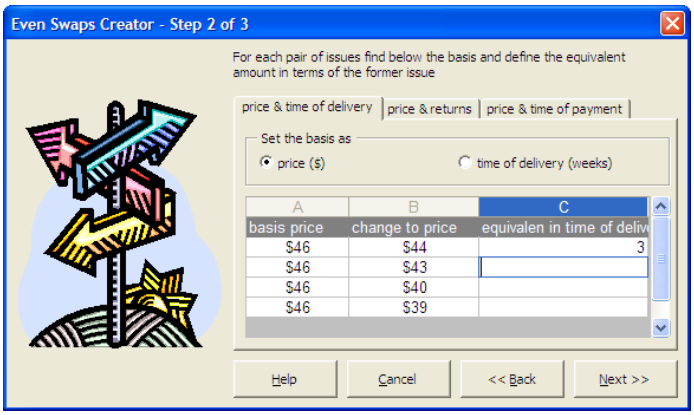


Fig. 2. Evaluating equivalent amounts for the swaps

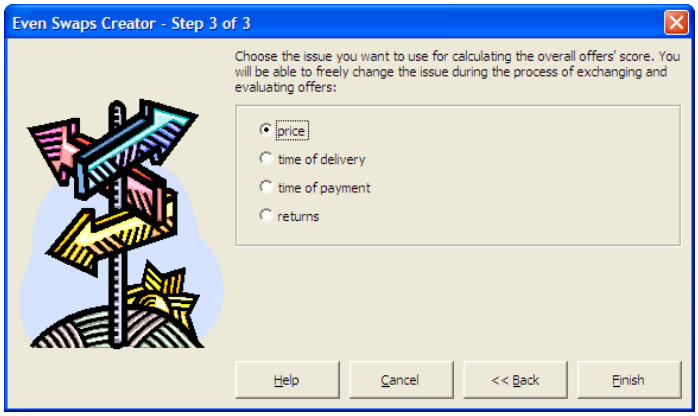


Fig. 3. Choosing the issue for creating the overall score of the offers



Fig. 4. Offer exchange panel

6. Summary

In this paper we have proposed the even swaps based approach for developing the assessment capability of ENS. It is an alternative to the classic methods based on the utility or desirability scores, especially an additive scoring system, that nowadays are commonly used in most of the ENSs. We have chosen the even swaps method since it allows negotiators to avoid comparing issues and resolution levels by means of some abstract values and requires only the analysis in terms of the objectives the parties decided to negotiate. The substantial merit of even swaps is they allow not only to consider one offer to be better or worse then another, but also to find the difference between them and present it as the overall score in terms of selected objective. That is remarkable since the outranking based multiple methods may result in incomparability, in which we are not able to consider one offer to be better, worse or even equal to another. Changing or adding new issues to the negotiation problem evaluated with even swaps is very easy. It requires only an additional swap analysis which boils down to find new equivalent values the number of which depends on the number of resolution levels declared for the new issue. It is a remarkable advantage in comparison with traditional additive scoring system, which requires both distributing scores among all new feasible resolution levels and rescaling the old ones.

To find the true value of the even swaps as a tool for negotiation support we need to complete the ENS. Since it is going to be equipped both with even swaps and additive scoring system we will be able to find how this two scoring methods are perceived by the negotiators and which of them is more acceptable and found easier by them.

References

1. Angur M. G., Lotfi V., Sarkis J.: *A Hybrid Conjoint Measurement and Bi-criteria Model for a Two Group Negotiation Problem, Socio-Economic Planning Sciences,* 30(3), 95 – 206 (1996).
2. Bellucci E., Zeleznikow J.: *Managing Negotiation Knowledge: from negotiation support to online dispute resolution,* John Zeleznikow, Arno R. Lodder, Second international ODR Workshop (odrinfo.info), Tilburg: Wolf Legal Publishers, pp 11-22 (2005).
3. Forman E., Selly M. A.: *Decision by Objectives,* World Scientific Publishing, Singapore. (2001).
4. Hammond J. S., Keeney R. L., Raiffa H.: *Smart Choices. A Practical Guide To Making Better Life Decisions,* Broadway Books, New York (2002).
5. John R., Raith M. G.: *Optimizing multi-stage negotiations,* Journal of Economic Behavior & Organization, Vol. 45 (2001) 155–173.
6. Keeney R., Raiffa H.: *Decisions with multiple objectives: preferences and value tradeoffs,* Wiley (1976).
7. Keeney R., Raiffa H.: *Structuring and Analyzing Values for Multiple-Issue Negotiations,* In: Young H. P. (ed.): Negotiation Analysis. The University of Michigan Press (1991) 131-152.
8. Kersten G.: *E-negotiation systems: Interaction of people and technologies to resolve conflicts,* InterNeg Research Papers, INR 08/04 (2004).
9. Kersten G., Michalowski W., Cray D., Lee I.: *An Analytic Basis for Deci-sion Support in Negotiations,* Naval Logistic Research, 38, s. 743-761 (1991).
10. Madanmohan T. R., Kersten G. E., Noronha S. J., Kersten M. J., Cray D.: *Learning Negotiations with Web-based Systems,* InterNeg Research Papers, INR 06/99 (1999).
11. Raiffa H., Richardson J., Metcalfe D.: *Negotiation Analysis,* The Belknap Press of Harvard University Press, Cambridge (2002).
12. Roy B. (1985). *Methodologie Multicritere d'Aide a la Decision,* Editions Economica, Paris.
13. Saaty T.: *The Seven Pillars of The Analytic Hierarchy Process,* In: Köksalan M. (ed.): Multiple Criteria Decision Making in the New Millennium. Proceedings of the 15th International Conference, MCDM, Springer. Lect. Notes Econ. Math. Syst. 507, Berlin. (2001).
14. Schoop M., Jertila A., List T.: *Negoisst: a negotiation support system for electronic business-to-business negotiations in e-commerce,* Data & Knowledge Engineering, 47 (2003) 371-401.
15. Thiessen E., McMahon J.: *Beyond Win-Win in Cyberspace,* Ohio State Journal on Dispute Resolution, 15: 643 (2000).
16. Wachowicz T.: *Application of Multiple Attribute Stochastic Dominance To Selection Of Negotiation Strategies In E-Negotiations,* In: Trzaskalik, T. (ed.): Multiple Criteria Decision Making '05., The Karol Adamiecki University Press, Katowice (2006).
17. Wachowicz T.: *Negotiation and Arbitration Support with Analytic Hierarchical Process,* In: Trzaskalik, T. (ed.): Multiple Criteria Decision Making '07. The Karol Adamiecki University Press, Katowice (in press).

Proceedings of the International Multiconference on
Computer Science and Information Technology, pp. 607 – 613
ISSN 1896-7094
© 2007 PIPS

The Brand Equity – Marketing and Financial Approach

Urszula Świerczyńska-Kaczor[1], Paweł Kossecki [2]

[1] Świętokrzyska Academy in Kielce,
Kielce, ul. Żeromskiego 5 Poland
swierczynska@pu.kielce.pl
[2] Media Development Loan Fund (USA)
37 West 20th Street, Suite 801, New York, NY 10011, USA
kossecki@poczta.onet.pl

Abstract. The authors emphasize two important issues of the management and measurement of brand equity on the Internet market: creating relationships between company and virtual communities and the process of valuing customers. Strengthening cooperation with virtual societies leads to enhancing brand equity in the areas of brand perception, customer's loyalty and company's reputation. In this way the potential of the company's brand and its portfolio are reinforced. Moreover, the brand equity depends on the value of the customers' base. We propose the method of valuing customers by referring to the level of customers' loyalty and their Customer Lifetime Value.

Key words: virtual communities, brand equity, Customer Lifetime Value

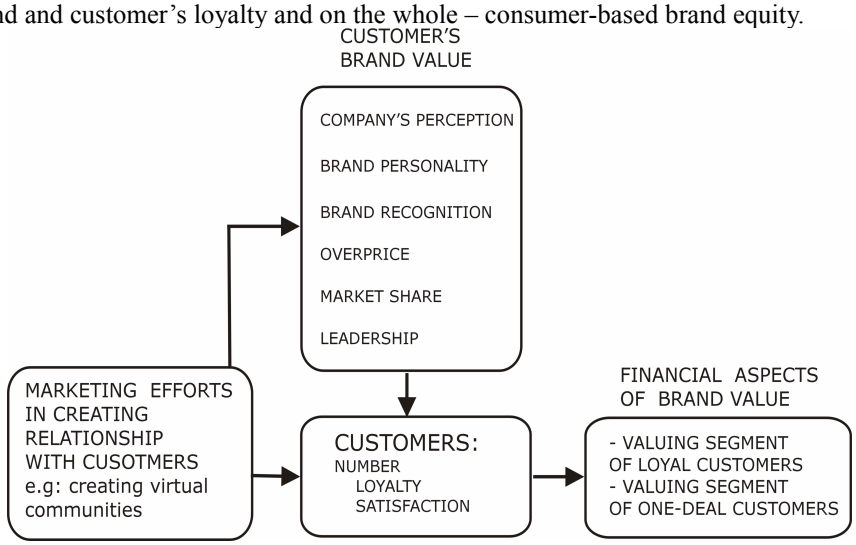
1. Introduction

There are varying approaches to the concept of brand equity in literature. Some of them are concentrated on customer's perception of brand (consumer-based brand equity), while others are focused on the brands financial performance characterized as market share or revenue (firm-based brand equity). In this article we point out two aspects in the process of management and measurement of brand equity on the Internet market:

1. Creating the stronger relationships with customers by utilizing virtual communities. We discuss the role of virtual societies in enhancing brand equity in the areas of customers' perception, company's reputation and customer's satisfaction. This issue is linked to consumer-based brand equity.
2. Valuing customers by referring to their loyalty. Our concept is based on dividing customers' base into two segments and then assessing both of them – the segment of loyal customers and the segment of customers whose focus is on one deal with a company on the Internet market. In this issue we note the significance of financial brand performance such as the number of attained customers and their profitability.

2. Virtual communities as a marketing tool in enhancing brand equity

In this article we analyze the element, which seems to be neglected in most conducted surveys, when looking to shape customer's satisfaction and loyalty [1] - the virtual community. Many virtual communities are connected directly or indirectly with different companies, resulting in an effect on the companies' reputation, perception of brand and customer's loyalty and on the whole – consumer-based brand equity.



Exibit 1 The proposed approach to management and measurement brand equity

To emphasize the influence of virtual communities to brand equity we refer to the concept of brand equity as a multidimensional construct composed of [5]:

1. overprice,
2. customer satisfaction and loyalty,
3. perceived quality,
4. leadership/popularity,
5. use value,
6. brand personality,
7. brand recognition,
8. organization,
9. market share,
10. price indexes and distribution

1 E.g. ease of navigation, flexibility, customization, reliability site design financial security interactivity accuracy of data information about company trust to the internet supplier [2]

Exibit 2 The process in which virtual communities can strengthen the brand equity

Communities used for...	**The benefits for customers or companies**	**The influenced area of brand equity**
creating involvement with a company	The informational, social, advantages to customers. Rarely financial benefits.	Customer's loyalty
hyping platform	More trustworthy information in blogs	Reputation
the source of data	Help to modify product according to an expectation of customers'	Customer's Satisfaction
product placement in on line games	Creating stronger relationship with brand	Enhancing Brand Personality
reverse product placement	Building knowledge of brand before selling, creating strong relationship with brand	Enhancing Brand Personality

Virtual societies are unique, they are a valuable information source for creating new products or modifying existing ones. Many companies are beginning to take advantage of Internet societies as a data source e.g. Procter&Gamble manages the Vocalpoint community – a platform for American mothers to express opinions about daily routines and it is also a way for Procter&Gamble to trial product's samples. InnoCentive is a virtual community for chemists to work out scientific problems. This is an example of the community which the connection between is even stronger – the members receive the payment for their work.

The virtual community is a good platform of promotion, such as product placement in on-line games or embedding news about companies in customer's posted blogs, forums or chats. So far, information in blogs can be perceived as more reliable than traditional Internet promotion, but a growing number of flogs lead to customer's mistrust. Flog is a fake blog, written by company posing as a blog written by 'a member of the public'' [2].

Furthermore, the virtual societies can be used as a pre-market (not necessary test market) before launching the product on the real world. Using reverse product placement the companies create a brand in the fictional world such as online games

2 One of the most well-known flog is Walmarting Across America, written by two Walmart "enthusiasts" who decided to journey across America

and then start selling the real product with the same brand on the Internet or traditional market. New energy drink 'Potion' is a recent example of successful reverse product placement – the drink was placed in the online game Final Fantasy and now the company sells the product to real customers [3]. There are many advantages for producers resorting to a reverse product placement strategy:

- it is easier to build brand personality in the virtual market,
- the product can help the customers achieve a goal in the virtual world, this leads to building a stronger relationship between brand and its users,
- in the Internet consumers do not experience the necessity of repairing the product, so the product is perceived as flawless

Through creating virtual communities company has also a chance to receive financial benefits. For example a survey conducted on the virtual auction eBay indicates that shoppers who took part in this platform were more profitable compared to non-members. Members of virtual societies were willing to pay more, more frequently won the auctions and their activity increased eBay revenues [1].

The relationship between company and virtual communities is different eg. there are companies managing the group, companies organizing the platform to exchange opinions, companies paying for finding solutions or companies interested in advertising on the web-site. If the company aim to manage the Internet community, the main goal is to encourage members to communicate. There are many examples of groups which stop functioning after a short period of time, there are virtual communities so called "gost town". To avoid this situation company must estimate the potential benefits and costs connected with member's 'work' with particular community. The members of virtual communities decide to participate in a project (forums, chats, wiki, blogs) if they notice some advantages: financial, social or informational.

A few examples of the benefit and cost for members of different communities are presented in the table below.

Exibit 3 The advantages and the cost of participating in the virtual communities

	Benefits for members	Cost
Vocalpoint	Informational – knowledge of using products Financial – the samples of products, discounts Social – the chance to discuss the problems	Time, energy
InnoCentive	Informational – new information Financial – money Social – belonging to the noticeable group; a chance to discuss problems within the group,	Time, energy, the risk of giving inadequate solution

	Benefits for members	Cost
Java Technology Forums	Informational – new information, answer to the problems Financial – non Social – belonging to the a special interest group; the chance to discuss problems within the group, to be recognizable as expert	Time, energy, the risk of giving inadequate solution
Wikipedia	Informational – finding proper information Financial – non Social – the chance to share your knowledge	Time, energy, the risk of putting inadequate information

Success in building working community depends on carefully prepared project, it means – defining the way of communication, the frequency of interaction, the profile of members. Ch. Meyer defines a few stages in building a long-lasting virtual community [6]:

- Defining the clear aim of its existence such as exchanging opinions or solving problems
- Defining the profile of member
- Creating the process of exchanging benefits
- Creating the environment of exchange: web site design
- Choosing technology – forum, e-mail, chat or internet questionnaire

3. Valuing loyal and non-loyal customers

Number of customers was used in the prior analyses of internet market as a main indicator of the company's value. In 1999, when the Internet market flourished, valuing companies was based on the volume of the customers' base, without considering their potential of generating profits e.g. one customer of the Internet bookstore Amazon was valued $3000, and one customer of the protal Yahoo! was valued to $1000. In our approach to brand equity the loyalty of customers is more important than the quantity of customers. In this concept we refer to the idea of Customer Lifetime Value.

According to S.Gupta, R. Lehmann, J.A. Stuart the value of the customer is understood as "the expected sum of future earnings based on key assumptions considering retention rate and profit margin" [4]. The customers' value is the sum of the lifetime value of all the cohorts of customers. In this model the number of cohorts and the company's lifetime approach infinity, but including the retention rate allows

us to discount the customers who stop buying. So, the value of all customers' cohorts at the present is:

$$VALUE = \sum_{k=0}^{\infty} \frac{n_k}{(1+i)^k} \sum_{t=k}^{\infty} m_{t-k} \frac{r^{t-k}}{(1+i)^{t-k}} - \sum_{k=0}^{\infty} \frac{n_k c_k}{(1+i)^k}$$

n_k - *the number of customers in the k cohort of customers*

m_{t-k} - *the margin generating by customer in k cohort*

k – k-nth cohort of customer
i – discount rate
t – period of time
r – retention rate (the percentage of customers which repeat purchasing in the t period)

c_k - *cost of attaining customer for the k cohort*

We modified this model by dividing the Internet customers into two segments. The consumer's loyalty measured by behavioral outcome - the frequency of purchase is the criteria of segmentation:

- One-deal customers – customers sensitive to price and sales promotion tools, who often change the Internet supplier looking for the most suitable transaction. It is difficult for a company to build a long term relationship with them. The retention rate for this group is lower than the 'average retention rate'. Few customers from this group repeat purchase in the next periods.
- Loyal customers - customers prefer dealing with reliable, trustworthy partners. They are less sensitive to price and more likely to consider the reputation of vendor in decision of purchase.

In estimating brand value the proportion between the volume of two indicated segments is pivotal. If the vendor focuses on attaining promotion-sensitive customers the future of the brand is more uncertain and the brand is more susceptible to competitors' movements. So, the present value of customers' base can be estimated as:

$$VALUE = V_{OD} + V_L$$

V_{OD} - value of one-deal customers

V_L - value of loyal customers

If all customers in the segment one-deal make only one transaction, the equity for valuing the customers' base is:

$$VALUE = \left(\sum_{k=0}^{\infty} \frac{n_{(OD)_k}}{(1+i)^k} \sum_{k=0}^{\infty} \frac{m_{(OD)_k} - c_{(OD)_k}}{(1+i)^k}\right) + \left(\sum_{k=0}^{\infty} \frac{n_{(L)_k}}{(1+i)^k} \sum_{t=k}^{\infty} m_{(L)_{t-k}} \frac{r^{t-k}}{(1+i)^{t-k}} - \sum_{k=0}^{\infty} \frac{n_{(L)_k} c_{(L)_k}}{(1+i)^k}\right)$$

$n_{(L)_k}$ - *the numbers of customers in loyal segment – k - cohort*

$m_{(L)_k}$ - *the margin generating by customer in loyal segment in k-cohort*

$n_{(OD)k}$ *- the numbers of customers in one-deal segment – k - cohort*

$m_{(OD)k}$ *- the margin generating by customer in one-deal segment in k-cohort*

k – k-nth cohort of customer

i – discount rate

t – period of time

r – retention rate (the percentage of customers which repeat purchasing in the t period)

c_k *- cost of attaining customer for the k cohort*

This equation illustrates the importance of the consumer's loyalty. If the company attracts customers for a short time, without creating a long-term relationship with them the value of the customers' base is significantly lower compared to the company with a larger loyal segment.

Conclusion

Strengthening cooperation with virtual societies leads to enhancing brand equity in the areas of brand perception, customer's loyalty and company's reputation. Moreover, enhancing customers' loyalty (e.g through virtual communities) leads to increasing the value of customers' base (what is illustrated by calculation of customers lifetime value). In this way the potential of the company's brand and its portfolio are reinforced.

References:

1. Algesheimer R., Dholakia P. M., *Czy opłaca się tworzyć społeczności klientów,* in Harvard Business Review Polska, March 2007.
2. Bansal H. S., McDougall G., Dikolli Sh. S., Sedatole K. L, *Relating E-satisfaction to Behavioral Outcomes: An Empirical Study,* Apirl 2003, http://www.ssrn.com/, SSRN – id434400.
3. Edery D., *Reverse Product Placement – wykorzystaj wirtualny świat* – Harvard Business Review Polska, January 2007.
4. Gupta S., Lehmann D. R., Stuart, J. A., *Valuing Customers,* (February 2003). HBS Marketing Research Paper No. 03-08, http://ssrn.com/abstract=459595
5. Martinez J. V., *Brand Equity Profile and the Measurement of its Components',* http://www.ssrn.com/ SSRN id895752,
6. Meyer Ch. *Najlepsze sieci to te, które działają w określonym celu,* in Harvard Business Review Polska, February 2007.

2nd International Workshop on Secure Information Systems (SIS'07)

The SIS workshop is envisioned as a forum to promote the exchange of ideas and results addressing complex security issues that arise in modern information systems. We aim at bringing together a community of security researchers and practitioners working in such divers areas as networking security, antivirus protection, intrusion detection, cryptography, security protocols, and others. We would like to promote an integrated view at the security of information systems.

As information systems evolve, becoming more complex and ubiquitous, issues relating to security, privacy and dependability become more critical. At the same time, the global and distributed character of modern computing – typically involving interconnected heterogeneous execution environments – introduces many new and challenging engineering and scientific problems. Providing protection against increasingly sophisticated attacks requires strengthening the interaction between different security communities, e.g. antivirus and networking. New technologies lead to the emergence of new threats and attack strategies, involving smart mobile devices, peer-to-peer networks, instant messaging, VoIP, mesh networks or even networked consumer devices, such as house appliances or cars. Furthermore, the increasing openness of the communications infrastructure results in novel threats and can jeopardize critical enterprise and public infrastructure, such as industrial automation and process control systems. Not only it is estimated that half of all Web applications and Internet storefronts still contain some security vulnerabilities, but secure commerce applications are also exposed to increasingly elaborate attacks, including spyware, phishing and other social engineering methods.

In order to develop a secure system, security has to be considered in all phases of the lifecycle and adequately addressed in all layers of the system. At the same time, good engineering has to take into account both scientific and economic aspects of every solution: the cost of security has to be carefully measured against its benefits – in particular the expected cost of mitigated risks. Most companies and individuals treat security measures in information system as a necessary, but often uncomfortable, overhead. The increasing penetration of computing in all domains of everyday life means that security of critical business systems is often managed and maintained by personnel who are not knowledgeable in the field. This highlights the importance of usability and ease of configuration of security mechanism and protocols.

Covered topics include (but are not limited to):

- Access control
- Adaptive security
- Cryptography
- Copyright protection
- Cyberforensics
- Honeypots

- Intrusion detection
- Network security
- Privacy
- Secure commerce
- Security exploits
- Security policies
- Security protocols
- Security services
- Security evaluation and prediction
- Software protection
- Trusted computing
- Threat modeling
- Usability and security
- Viruses and worms
- Zero-configuration security mechanisms

International Programme Committee

- Ishfaq Ahmad, UT Arlington, USA
- Sergey Bratus, Dartmouth College, USA
- Nicolas T. Courtois, University College of London, UK
- Lech J. Janczewski, University of Auckland, New Zealand
- Silke Holtmans, Nokia Research, Finland
- Kamil Kulesza, IFTR PAS, Poland and University of Cambridge, UK
- Miroslaw Kutylowski, Wroclaw University of Technology, Poland
- Josef Pieprzyk, Macquarie University, Australia
- Sugata Sanyal, Tata Institute of Fundamental Research
- Andreas Schaad, SAP Research, Germany
- Janusz Stoklosa, Poznan University of Technology, Poland
- Osamu Takata, Hitachi Europe Ltd.
- Johnson Thomas, Oklahoma State University, Tulsa, USA
- Thomas Walter, DoCoMo Labs Europe, Germany

Organizing Committee

Konrad Wrona, SAP Research, France
Krzysztof Szczypiorski, Warsaw University of Technology, Poland

Proceedings of the International Multiconference on
Computer Science and Information Technology, pp. 617 – 628
ISSN 1896-7094
© 2007 PIPS

Integrated, Business-Oriented, Two-Stage Risk Analysis

Andrzej Białas, Krzysztof Lisek

Institute of Innovations and Information Society
40-954 Katowice, ul. Wita Stwosza 7, Poland
{Andrzej.Bialas, Krzysztof.Lisek}@insi.pl

Abstract : This paper presents an integrated, business-oriented, two-stage risk analysis method related to the Information Security Management Systems (ISMS) concept. The current state of the work is presented, including risk analysis methods and their implementation. The concept assumes the integration of preliminary overviews as well as high- and low-level risk analyses. High-level risk analysis works with the needs of business processes and presents criticality of these processes. Low-level risk analysis works with assets and selects safeguards in a cost-effective manner. It is assumed that the presented risk analysis concept can be used in other management systems: business continuity and IT services management. The paper concludes the current state of the work and defines its further directions.

Keywords: Information security management, Risk analysis.

1 Introduction

The paper presents a business-oriented approach to risk management for information and e-business management. Risk management is a significant part of the extended version of the ISMS framework [1], [2], based on the PDCA (Plan-Do-Check-Act) scheme.

Present-day Information and Communication Technologies (ICT) are widely used for the management of large businesses, critical information infrastructures, or emerging e-services, upholding e-business, e-government or e-health applications. This situation deman ds technically and economically efficient solutions that should provide the right assurance for their stakeholders and users. Such demands can be satisfied by the common approach which takes into account:

– the needs of business processes with respect to information and e-services security,
– effective use of ICT providing managed IT services for business processes,
– detailed risk characteristics, including cost/benefit aspects,

– dependable and trustworthy solutions providing the right assurance.

Most of these issues are addressed in the information security management systems, where all activities concerning information security are based on risk analysis. Risk analysis is the key to establish a cost-effective and business-oriented management system. For this reason, the authors have developed a new approach to risk management with the following main assumptions:

– possibility to thoroughly recognize business processes and management expectations;
– gradation of business processes with respect to security;
– enhanced, two-stage, UML-model-based risk analysis that allows to assess the risk according to qualitative and/or quantitative measures, with a built-in *ROI* (*Return on Investment*) type mechanism that allows to select different safeguards;
– compliance with risk management requirements contained in ISO/IEC 27001 [1] and further requirements which will be contained in ISO 27005 (currently under development).

The concept of a two-stage risk analysis for risk-based information and e-services security management has been elaborated on the basis of:

– a study of the needs and requirements of various organizations,
– an analysis of the current state of standards, legal requirements and technology (overview of existing methodologies and tools),
– experiences sampled during deployment and case studies prepared in the course of the specific targeted project "Complex information and services security management system" [3].

The paper presents a holistic approach to risk management which allows to create a business-oriented and highly integrated risk analysis. The paper also concludes the current state of the work and defines its directions for the nearest future.

2 Needs and Requirements for Business-Oriented Risk Analysis Compatible with ISO/IEC 27001

Modern organizations, which want to ensure information security of their assets, need to establish and maintain effective information security management systems. These systems, in turn, must integrate a well balanced set of safeguards selected on the basis of the existing risks and business needs. Thus the following requirements were formulated to better specify the characteristics and needs of the organizations for a risk analysis compatible with ISO/IEC 27001:

- top-down approach – beginning from preliminary organization overviews, security needs of business processes analyses, high- and low-level risk analyses; the implemented ISMS is refined step by step but still focuses on business needs [4, 5];
- identification of the needs of business processes and criticality of these processes;
- identification of basic factors influencing the security of each asset, related to the given business process, e.g. possible threats, possible vulnerabilities, environment;
- possibility to select cost-effective safeguards and create a risk treatment plan;
- possibility to use some elements of the risk analysis in the risk management part of business continuity management and IT services management contained in the Integrated Security Platform (ISP) [4].

3 Summarizing the Current State of the Art

The development of risk analysis requires to perform an extensive study of the current state of the art including all available risk management standards, recommendations, best practices, guidelines, case studies, methodologies, their implementations and renowned supporting tools, particularly:

- key standards dealing with information security management systems, i.e. ISO/IEC 27001 and ISO/IEC 17799 (now ISO/IEC 27002) – to properly understand the ISMS implementation requirements in the field of risk management;
- auxiliary documents, i.e. IT security and risk management methodologies, such as: ISO/IEC [6], US NIST publications [7], German IT Grundschutz [8]–to supplement or extend the risk analysis features and facilities; the motivation for two-stage risk analysis methodology (called "combined approach") is discussed in [6];
- state of the art in business processes modelling – to identify their relationships with the above mentioned management systems, especially information security and IT services; "business orientation" is the key issue for the risk analysis;
- methods and tools related to risk management in the information security domain;
- ICT and general technical issues (communication protocols, network equipment, cryptographic applications, physical protection) – as a context of risk management;
- business continuity management, e.g. [9], IT services management [10], [11], as well as quality, environmental, occupational safety and health management systems which co-exist in the organization – to better understand risk management requirements in such systems;
- potential sources of statistical information on threats – to serve as input data in risk analysis.

There are a number of t ools, like Cobra [12], Cora [13], Coras [14], Cramm [15], Ebios [16], Ezrisk [17], Mehari (Risicare) [18], Octave [19], and Riskpack [20] that

specialize in risk management. However, they do not cover all aspects of business-oriented risk analysis compatible with ISO/IEC 27001. They focus on IT aspects only or do not set enough store by the significance of business process for the organization. Ebios can perform a detailed risk factors analysis but cannot operate on monetary values during the risk analysis. Cora can perform the ROI analysis. There is a UML-based advanced model of risk implemented in Coras, which uses the safety risk management methods (Hazop, FME(C)A, FTA) and allows a simple causality analysis. The available risk analysis methods focus rather on a detailed risk analysis for the whole of ICT systems in the organization. Only [8], [21] assume to implement the combined approach [5] which allows to identify the security domains of similar security requirements during the preliminary high level risk analysis. In the further course of the process, this approach also allows to perform a detailed risk analysis only for the critical domains, and to apply baseline protection for others. Thanks to this approach it is possible to avoid a costly detailed risk analysis for the entire organization, however it is not fully compatible with ISO/IEC 27001.

Please note that the assets inventories that focus on risk management are different from those that focus on security, business continuity or IT services management. Additionally, there are some differences between risk analysis requirements in such systems. Mehari and its Risicare risk supporting tool are able to identify relationships between IT service quality and risk value.

Risk analysis also need s some sources of information about threats. This information helps to prepare a better risk analysis based, to the highest possible extent, on real factors. Currently, these sources are scattered and the information has to be obtained from different places, e.g. information about current levels of threat in the Internet can be obtained from the Polish CERT [22].

There are a few risk analysis methods and tools but each of them is only partially compliant with the optimum risk analysis characteristics described above. The review of the current state of technology, including standards, best practices, methods and tools, helped to select the most useful features for risk analysis and to develop new features, especially in terms of business flexibility and usability.

4 Concept of Business-Oriented Risk Analysis

Information security management, safeguards selection, and incidents management are based on a fundamental risk analysis. According to ISMS requirements, every asset of the organization should be identified. Moreover, for each of these assets it is necessary to identify threats, vulnerabilities and impacts that may cause losses of information confidentiality, integrity and availability. In the end, a satisfactory risk

treatment plan should be made. Although the risk analysis is compliant with the ISO/IEC 27001 requirement, such risk management is too flat in relation to today's business needs. In modern organizations, the management focuses on business processes, their optimization and security [23]. Therefore, modern risk analysis also should be business-oriented and should include business processes in the risk assessment approach.

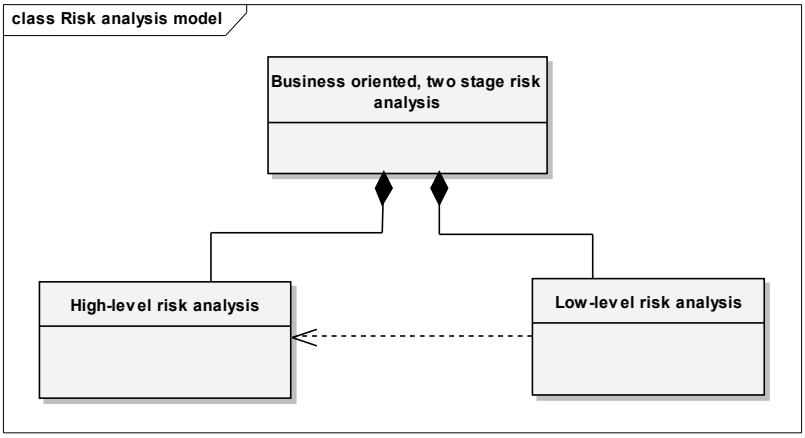


Fig. 1. Business-oriented, two-stage risk analysis.

The results of the preliminary business analysis conducted in the above mentioned specific targeted project [3] allow to define a concept of business-oriented risk analysis which carries out a holistic approach to all aspects of risk management, as it is shown in the Fig. 1.

T he two-stage risk analysis concept is based on earlier works [4], [5]. During the development process new factors and an integrated method of safeguards selection were added. The factors are responsible for visualizing a mutual relation between business processes and assets. The risk analysis divides all tasks connected with risk management into two stages: one for the tasks connected with security assessment of business processes (high-level risk analysis) and the second for the tasks connected with security assessment of assets related to these business processes, to perform the risk treatment and selection of safeguards (low-level risk analysis).

4.1 High-Level Risk Analysis

High -level risk analysis comprises the first phase of the risk management process and detects the level of information security risk for each business process recognized in the organization. This part of the analysis is responsible for creating the image that

presents the state of business processes criticality for the organization with respect to information security.

During this phase of risk management every identified business process is analyzed in terms of confidentiality, availability and integrity. The analysis process is based on information obtained as a result of preliminary activities conducted prior to the analysis. Most important are the results of the general organization overview prepared in accordance with the Organization Overview Criteria (OOC) [4], as well as the results of the detailed organization overview prepared in accordance with the Basic Security Needs Criteria (BSNC) carried out in the phase of the ISMS implementation prior to risk analysis. Risk management encompasses the following steps indicating what should be done for every business domain:

1. Characterize business processes criticality for the organization (C4O)
2. Characterize business domain dependency on ICT (ITDD), man dependency degree (MDD), and the dependency on other predefined criteria (xDD).
3. Identify protection needs (PN) concerning integrity, confidentiality and availability.
4. Determine business impact (BI) concerning integrity, confidentiality and availability.
5. Calculate high-level risk (HLRx) concerning integrity, confidentiality and availability.

High-level risk HLRn (where n stands for, respectively: I for integrity, C for confidentiality and A for availability) accumulates the levels of C4O, ITDD, MDD, xDD, PNn, and BIn in the following way:

$$HLRI[j] = C4O[j] * (ITDD[j] + MDD[j] + xDD[j]) * PNI[j] * BII[j] \quad (\mathbf{1})$$

$$HLRC[j] = C4O[j] * (ITDD[j] + MDD[j] + xDD[j]) * PNC[j] * BIC[j] \quad (2)$$

$$HLRA[j] = C4O[j] * (ITDD[j] + MDD[j] + xDD[j]) * PNA[j] * BIA[j] \quad (3)$$

Aggregated results represent g eneral security importance HLR of the j-th business process:

$$HLR[j] = \frac{HLRI[j] + HLRC[j] + HLRA[j]}{3} \quad (4)$$

where j represents the j-th business process.

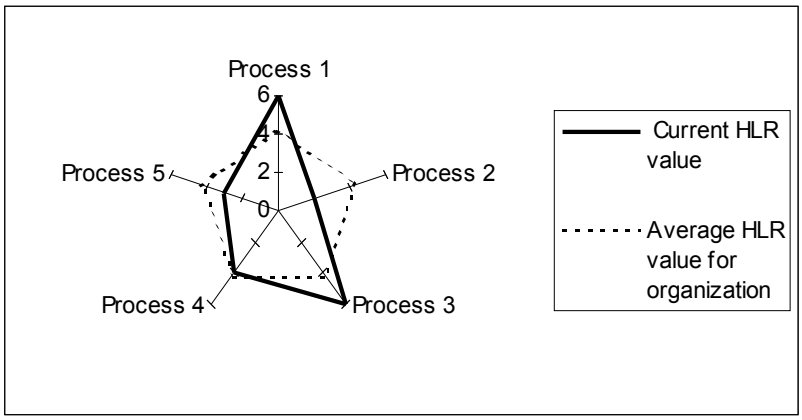


Fig. 2. Security risk map diagram.

Finally, based on HLR[j] results, it is possible to create a security risk map (SRM) for the whole organization, presented in the Fig. 2. SRM graphically presents the calculated level of information security for each business process in relation to the average one. With SRM, most critical business processes for the organization can be recognized to better perform the detailed risk analysis and risk treatment process.

4.2 Low-level risk analysis

Low-level risk analysis i s the next important phase of the risk management process. Its objective is to identify and determine the risk volume for each asset which is vulnerable to threats and, at the same time, exposed in the threat environment. The preliminary version [5] is enhanced and new factors, representing business security needs are added. The risk volume is a function of the following:

- Asset value
- Event possibility
- Vulnerability severity
- Efficiency of the existing or planned controls
- Special factors related to business processes, to which a given asset belongs.

The risk analysis comprises the identification of all possible risk cases and then the estimation of the risk volume. In other words, one has to identify all potential events which may have negative influence on the organization's operations and whose source are IT systems and their environment. Then the event possibility is estimated. Other factors that have to be assessed are threats severity and the probability of their occurrence in reality. This negative influence is the so called impact related to the loss

or violation of assets. The negative events possibility depends on many factors that have to be taken into account in the above mentioned estimations:

- how attractive is a given asset to the potential intruder?
- what is event possibility or frequency of occurrence?
- how easily can the asset vulnerability be exploited?
- what is the impact for a business process resulting from loss or violation of a given asset?

Low-level risk analysis uses the information about assets gathered in the Common Assets Inventory (CAI) [4]. The most important one is the assets business value which says how much a given asset is worth for the organization. Risk value can be estimated by means of the Courtney method which assumes that risk value is the product of two factors: the occurrence rate of the event (event possibility) and the volume of its consequences. Since it is difficult to obtain reliable data on the occurrence rate, it is estimated indirectly. We propose the following: the product of threat severity (TS) and vulnerability severity (VS), with respect to their maximal values. The discrete predefined measures are used. Below, one can find a risk volume assessment method that is an extension of this concept.

Risk value (RV[i]), determined in the course of the i-th analysis and expressed with respect to an arbitrary point scale applied to estimate the value of the organization's assets, is:

$$RV[i] = \frac{VS[i] * TS[i]}{VS_{MAX} * TS_{MAX}} * AV[i] * PS[i] * RF[i] * PIF[i] \tag{5}$$

where VS[i] is vulnerability severity level; VS_{MAX} – maximum arbitrary level of vulnerability severity; TS[i] – threat severity level; TS_{MAX} – maximum arbitrary level of threat severity, AV[i] – asset value in points estimated in the course of the i-th analysis, PS[i] – proportional asset value loss as a result of the threat analyzed in the i-th analysis, RF[i] – risk factor representing the business process level of criticality, and PIF [i] – process impact factor.

The latter two factors show a mutual relation between business processes and assets, therefore they need some explanation. Risk factor (RF) is calculated the basis of high-level risk analysis, where the most critical business process has the maximum value and the least critical one – the minimum. The other processes have the RF value calculated proportionally to the HLR value. RF has the same value for each asset in the same business process. Process impact factor (PIF) represents business process loss in the effect of asset loss or violation for the analyzed threat-vulnerability pair. PIF is assessed for each asset in the given business process.

Comparing the volumes of RV[i] (in the step i) and RV[i+1] (step i+1) allows to estimate the effect of the controls applied to reduce RV[i]. The volumes of VS[i+1]

and TS[i+1], with respect to the new situation (proposed controls implemented), are estimated and on this basis RV[i+1] is calculated. In other words, it is possible to estimate the efficiency of the actions undertaken.

Each organization should keep under control not only its security but also the cost of achieving and maintaining this security. The objective of all activities and investments in the domain of security is to improve the organization's position on the market or in the society. Economic efficiency can be achieved when one simple rule is obeyed – one should apply only those security measures which are necessary and sufficient. This can be done by implementing more and more efficient management methods and tools which combine qualitative and quantitative (monetary) methods and allow simple cost/benefit analyses, including ROI. The calculation covers current situation values (i) and new safeguards values (i+1).

$$ROI = \frac{RVC[i] - RVC[i+1]}{SC[i+1] - SC[i]} \quad (6)$$

R isk value expressed in currency RVC can be presented as:

$$RVC[i] = RV[i] * UV[cur] \quad (7)$$

where UV [cur] is unit value expressed in currency.

Safeguard s costs (SC) should be calculated with respect to comparable periods of time (the best thing to do is to give their average annual values). In case a given safeguard is purchased, the value in question will not be its total purchase cost but the value depreciated in the course of a given year plus the cost of work devoted to the maintenance of the safeguard throughout the year.

After analyzing different variants of safeguards (having different cost and risk reduction possibilities), the most cost-effective variant is chosen for implementation. This way the organization always chooses the safeguards that are most suitable to the needs of its business processes and, on the other hand, are economically justified.

Example 1. Simple low-level risk analysis

The s ample organization has "Asset 1" working in "the business process of marketing". Previous risk assessment for this asset gives the following information:

RV[i]	= 157	assessed current risk value,
RVC[i]	= 15 700 €	assessed annualized risk value in currency,
SC[i]	= 1 000 €	current annualized safeguards cost,
RF [i] = RF [i+1] = 6		criticality of marketing process (derived for high-level risk analysis).

Asset 1 starts to be used in a very important project and the risk needs to be reduced below 100. Risk reduction can be achieved by choosing 1 of 3 different possible safeguards having different costs presented in Table 1. For each of them the risk was assessed:

Table 1. Simple low-level risk analysis

Safeguards	SC[i+1]	RV [i+1]	ROI
1	1 500 €	99	11.6
2	5 000 €	50	2.675
3	3 000 €	80	3.85

The best choice is safeguard 1 because it has a risk level acceptable to the organization and, additionally, has the best ROI value of the analyzed safeguards.

5 Conclusions

There are a lot of met hods and tools developed in the realm of risk security analysis. They need constant improvement to catch up with the development and dissemination of new technologies in different areas of business and society. The paper deals with the risk analysis method addressing the needs of business processes. The motivation for the business-oriented, holistic, two-stage risk analysis includes the following:

- the need to integrate assets risk analysis with the business processes information security needs,
- the need to provide modular and scalable solutions for different types of organizations,
- the need to provide an enhanced risk analysis method adaptable to other risk-based systems (business continuity management, IT services management).

Please note that the presented method facilitates continuous risk analysis in the organization. The assumptions for this project were specified and evaluated on the basis of:

- the investigation of needs and requirements concerning risk analysis in different management systems which deal with business processes, their information security, continuity, IT services, quality, environment, etc.,
- the current state of technology and standards overview.

The concept of two -stage risk analysis was validated in the course of the specific targeted project [3]. Within the project a case study workshop was conducted during which the process of risk assessment was tested in the area of change management in the software accessible on the basis of outsourcing agreements. Additionally, a

computer supporting tool is being developed on the basis of the presented risk analysis method. During this process some elements must be developed, including the following:

- support to graphical business processes modelling,
- analytical and statistical facilities,
- preconfigured risk scenarios for different types of assets.

The ROI model should be extended to consider progress in IT technology – more efficient safeguards can be cheaper thanks to this progress. In the future the presented business-oriented, two-stage risk analysis method can be used in other risk-based management systems, such as business continuity management or IT services management. This will be possible thanks to the use of the RF and PIF factors that link business processes and their security needs with assets risk analysis and describe the criticality of these processes. According to the latest information from ISO/IEC [24] there are efforts to base other management systems (i.e. quality) on risk, so this method can be used more widely.

References

1. ISO/IEC 27001. *ISMS – Requirements* (2005).
2. ISO/IEC 17999. *Code of practice for information security management* (2005).
3. Reports from the specific targeted project No 6ZR9 2005C/06667 Complex information and services security management system, ISS/EMAG, COIG S. A., 2006-2007, (in Polish).
4. Białas A.: *Development of an Integrated, Risk-based Platform for Information and E-services Security,* In: Górski J.: Computer Safety, Reliability, and Security, 25th International Conference SAFECOMP2006, Springer Lecture Notes in Computer Science (LNCS4166), Springer Verlag Berlin Heidelberg New York 2006, ISBN 3-540-45762-3, pp. 316-329.
5. Białas A.: *Bezpieczeństwo informacji i usług w nowoczesnej instytucji i firmie (Information and services security in a modern organization and company),* WNT Publishing House, Warsaw 2006 (in Polish).
6. ISO/IEC 13335-3, *Guidelines for the management of IT Security (GMITS),* Part 3.
7. *National Institute of Stand ards and Technology,* `http://www.nist.gov`
8. *IT Grundschutz,* `http://www.bsi.bund.de`
9. BS 25999 -1. *Business continuity management,* Part 1: Code of Practice (2006).
10. ISO/IEC 20000-1. *IT – Service management,* Specification (2005).
11. ISO/IEC 20000-2. *IT – Service management,* Code of practice (2005).
12. Cobra. `http://www.riskworld.net`
13. Cora. `http://www.ist-usa.com/`
14. Coras. `http://coras.sourceforge.net`

15. Cramm. http://www.ogc.goc.uk
16. Ebios. http://www.ssi.gouv.fr
17. Ezrisk. http://www.ezrisk.co.uk/
18. Mehari, Risicare. http://www.clusif.asso.fr ; http://www.risicare.fr/
19. Octave. http://www.sei.cmu.edu
20. Riskpack. http://www.cpacsweb.com
21. Białas A.: *The ISMS Business Environment Elaboration Using a UML Approach,* In: Zieliński K., Szmuc T. (eds.): Software Engineering: Evolution and Emerging Technologies. IOS Press Amsterdam (2005) pp. 99-110.
22. CERT Polska. http://www.cert.pl
23. Rummler G. A., Brache A. P.: *Improving Performance: How to Manage the White Space in the Organization Chart,* 2nd Edition, Jossey-Bass Inc. Publishers 1995, ISBN: 978-0-7879-0090-8.
24. Andrukiewicz E.: *ISO/IEC 27005 – Zarządzanie ryzykiem w procesie budowania systemu zarządzania bezpieczeństwem informacji (Risk management in the process of building an information security management system),* Presentation from the Conference: Wyzwania bezpieczeństwa informacji (Information Security Challenge), Warsaw 2006 (in Polish).

Proceedings of the International Multiconference on Computer Science and Information Technology, pp. 21 – 29 ISSN 1896-7094
© 2007 PIPS

Securing Voice over Internet Protocol

Ahmad Ghafarian, Randolph Draughorne, Steven Grainger,
Shelly Hargraves, Stacy High, and Crystal Jackson

North Georgia College & State University
Dahlonega, GA 30005
aghafarian@yahoo.com, electronchaser6@earthlink.net
steven.grainger@verizon.net, shellzz@cox.net
stacy.high-brinkley@ngc.com, crystalejackson@hotmail.com

Abstract. In recent years, there has been significant increase in VoIP and internet telephony usage. The users, whether corporate or individuals are subject to the same security risks that have affected data networks for many years. This is mainly because voice networks are IP-based and all IP protocols for sending voice traffic contain flaws. In this paper, we study the security risks associated with the VoIP including vulnerabilities, man-in-the-middle attack, and denial-of-service. We will also review the protection measure that can be taken to make VoIP more secure, such as authorization, authentication, transport layer security, and media encryption.

Keywords: VoIP, Security, IPSec, Security Levels, Media Gateway Control Protocol

1 Introduction

Voice over Internet Protocol (VoIP) is a technology that has reached a level of maturity and reliability such that it can now be applied to the enterprise environment. VoIP has the potential to reduce communications costs considerably and opens a new path in the development of new devices. However, like all technologies VoIP comes with a number of inherent risks that while serious can be managed provided the enterprise takes the appropriate precautions.

This paper will examine the risks faced by the VoIP service provider and describe methods to reduce the risk for both the service provider and the enterprise. In addition, we examine the security risks associated with VoIP and has organized these risks in several categories from a layered perspective: weaknesses related to IP, the combined use of legacy and new technology, gateway considerations, security levels associated with VoIP, service provider challenges.

2 Internet Protocol Weaknesses

2.1 Resource Exhaustion (Denial of Service)

Resource Exhaustion, carried out via DoS (Denial of Service) attacks which reduces the number of available IP addresses, bandwidth, processor memory, and other router/server functions. A VoIP based DoS attack bombards a call processing/managing application with large amounts of simultaneous requests that it cannot process, causing the application to shut down, thereby denying service to authorized or intended users.

Before describing how to safely secure a VoIP network, its weaknesses must first be understood. VoIP is carried across the backbone of the Internet using Internet Protocol (IP) addresses to locate customers operating on the voice communications network [4]. However, IP has its own flaws, which are then inherited by all VoIP networks.

2.2 Network Sniffing

Network Sniffing attacks occur when an individual is observing network traffic. Typically, any system on a network sharing a transmission medium has the ability to view other system traffic (Univ. of London Information Security Group, 1998, pp.6-7).

2.3 Message Replay Attacks

According to the University of London Information Security Group [12], this type of attack occurs when network sniffing is done between two systems. Recording of the conversation is done during the sniffing which may be replayed to other parties in an altered state.

3 VoIP Security Levels

As illustrated in Figure 1 (see Appendix A), VoIP security can be divided into four levels: configuration security, signaling packet security, voice packet security, and data packet security [6]. The details of each level are described in the following subsections.

3.1 VoIP Configuration Security Level

The goals of VoIP security include authorization, authentication, integrity, privacy, and non-repudiation. Authorization is achieved through proper configuration, which is established during set-up of a new subscriber [10] by authorizing the device in the network system. Authentication can also be done either during configuration or at a later stage. Once the device is authorized to the network, the customer premise

equipment (CPE) must provide a secure identification number to the network server. An authentication key is then exchanged between CPE and the network server. The CPE gateway is authenticated, and then the server provides an encryption key. The encryption key is used for secure communication between CPE and the network server. Popular protocols that are used for this handshake and secure communication include: Session Security Layer (SSL), Transport Layer Security (TLS), File Transfer Protocol (FTP), Trivial FTP (TFTP), and Secure Hyper Text Transport Protocol (SHTTP). Configuration security protocols and methods of implementation are usually provided by individual service providers.

During the configuration of VoIP, separation of voice and data traffic is placed into different LAN segments. This process involves the creation of a separate VLAN for Voice and another VLAN for Data. The advantage of the separate VLANs is the isolation of the voice signals from data signals as they travel across the network.

Another security measure that must be taken into consideration during configuration is the creation of Security Association (SA). SA is a virtual secure connection between two or more devices. The SA process involves authentication and exchange of tokens, or certificate, to produce encryption keys. Once the SA is established, a security mechanism will perform key exchange. In addition to the establishment of a SA between a CPE and a configuration server, each pair of CPE also needs a SA. The reason SA establishment is recommended during the configuration stage is because it is a time consuming process to be established during the signal packet transmission. Since pre-establishment of SA between all CPEs is difficult to manage in terms of CPU usage, they are generally established as needed. It is also possible to reuse a previously established SA between two CPEs. Establishing a SA with IPSec requires between 2 to 10 seconds. However establishing SA with TLS requires 1.5 seconds [8].

3.2 VoIP Signal Packet Security Level

In VoIP, after the call is set, a complex series of packet exchanges must take place depending on a signaling protocol. The problem is that the computer systems are addressed by their IP addresses, but users enter an ordinary telephone number using the Universal Resource Indicator (ULI) to place a call. The first step in this process is converting analog voice signals to digital, using an analog-digital converter. Since digitized voice requires a large number of bits, a compression algorithm can be used to reduce the volume of data to be transmitted. This process is called signal pocketing. Next, the voice signals are inserted into data packets and hence voice packets are constructed [8].

The signal packet security set-up is engaged in the authentication process, which identifies one or more parties in a SA. The SA is not established on a per-call basis. This is because setting up signal security for each call would cause unwanted delays and latencies in the placing of VoIP calls. At the signal layer, the authorization/encryption process begins with an exchange of the keys generated at the configuration level or a new signal security key can be generated and exchanged independently. During earlier security implementation of VoIP, IP Security (IPSec) and Internet Key Exchange (IKE) protocols were used at the transport layer. Once the transport layer is secured, the signal layer and voice layer can run on top of the transport layer. Due to performance considerations, this approach has been changed

and currently the Transport Layer Security (TLS) protocol largely is deployed on top of TCP for signal security.

TLS is an advanced version of SSL and is used for privacy of the communications for VoIP users. TLS is composed of two layers: the TLS Record Protocol and the TLS Handshake Protocol. The TLS Record Protocol provides connection security with some encryption methods, such as the Data Encryption Standard (DES). The TLS Handshake Protocol allows the server and CPE to authenticate each other and to negotiate an encryption algorithm and cryptographic keys before data is exchanged. It is also more efficient, reducing the computational and consequent processing burden that other protocols generate.

Recently, Internet Engineering Task Force (IETF) proposed the use of TLS over Stream Control Transmission Protocol (SCTP) or User Datagram Protocol (UDP). Although SCTP has open stream sockets at the transport layer and provides a rich set of features, such as good reliability and multiplexing, it may be too ponderous and code-intensive for VoIP applications that require greater agility. TLS over UDP is not gaining market attention since SIP clients only need to support SIP over TCP; support for UDP is optional.

3.3. Voice Packet Security Level

The latest standard voice encryption standard is the Secure Real-Time Transport Protocol (SRTP) with advanced encryption standard (AES). SRTP provides message confidentiality, authentication, integrity checking, and replay protection for voice packets on a per-packet basis. Initially, the data, minus the header information, is encrypted using AES and then a hash of the header and encrypted data is created using Keyed-Hashing for Message Authentication Secure Hash Algorithm-1 (HMAC-SHA1) [5].

SRTP+ [5] is presented as an approach to overcome the overhead problem and reduce an attacker's ability to exploit this vulnerability. The first two techniques center on mapping a pseudorandom number generator (PRNG) number with an authentication tag, creating a mapping pattern that must be known by both terminal nodes at the time the authentication tag for the outgoing packet is generated and also for verifying the tag of an incoming packet. Packets that are not authenticated are assumed to be malicious and are discarded.

A third technique eliminates the need for either a PRNG or a hashing function. In this technique the sender calculates in advance a series of random numbers and uses one of these numbers as the authentication tag for each packet. The receiver stores the N random numbers after decrypting the payload. These random numbers correspond to the sequence numbers for the next N expected packets and are compared to the authentication tags for succeeding packets for authentication. Before the first packet can be authenticated, the first N random numbers must be sent to the receiver, possibly during the SRTP key exchange. Figure 4 illustrates this technique (see Appendix A) [5].

3.4 Data Packet Security Level

Data packets in a VoIP implementation may be protected at the IP level using the Internet Protocol Security (IPSec). IPSec uses two protocols, the a uthentication header (AH) protocol and the e ncapsulating security payload (ESP), to provide integrity, confidentiality and authentication of data communications over IP networks [2]. In conjunction with an arbitrary 32-bit security parameter index (SPI) and the destination IP address, the protocol uniquely identifies the SA for the specified header field of each data packet or the entire data packet depending on the operational mode. Although these mechanisms validate the authenticity and integrity of the packet, the data itself is not encrypted so the data can still be viewed by unintended parties.

The sequence number field of the header in used to prevent replay attacks and is required to be sent by the origin host but not necessarily processed by the destination host. The sender's counter and the receiver's counter are initialized to "0" at SA and incremented with the first packet sent using a particular SA, i.e. a sequence number (SN) of "1". The transmitted SN must never be allowed to "replay"; hence, anti-replay must always be enabled by the origin host.

The header fields associated with security are not significantly different between AH and ESP but since ESP supports encryption, an initialization vector (IV) [2] field is included to account for encryption algorithms that require cryptographic synchronization data. I PSec supports two encryption modes [2]: Transport and Tunnel. Transport mode encrypts only the data portion of each packet while the tunnel mode encrypts both the header and the payload by encapsulating the original packet in a new packet masking of the source and destination IP addresses.

Key management is critical and IPSec supports both the Internet Security Association and Key Management Protocol (ISAKMP)/Oakley protocols [1] . These protocols use two phases of negotiations to establish an SA prior to data transmission.

4 VoIP Solutions

IP weaknesses are not the only security issues to consider. Since VoIP is a relatively new technology, companies are currently in the process of implementing it. One common method is by using a hybrid approach combining older circuit-switched technology with new VoIP technology.

4.1 Circuit-switched Technology Vulnerabilities

- Toll Fraud is a classic IP attack where the attacker impersonates an employee or performs Console Cracking (asking the operator for an outside trunk) to make long distance calls. However, the attacker impersonates a valid user and IP address by plugging in their phone or spoofing the MAC Ethernet address.
- Eavesdropping occurs when an attacker intercept voice messages. This allows the attacker to listen to voice conversation at ease without the target knowing. Easily available programs such as VOMIT (Voice over Misconfigured Internet Telephony) perform this function.

- Call Hijacking is when an attacker a SIP Response and redirects a caller to a rogue SIP address. This allows the attacker to intercept the call.

4.2 VoIP Technology Vulnerabilities

- IP Phone – "The IP Phone is a new desktop phone configured and equipped specifically for IP-based phone calls" (ISS, 2004, p.4). Working much like a regular landline phone, the IP Phone's Operating System (OS) may support a web browser; the user can access the phone's webpage and attempt to modify phone features and options [3].
- PC-Based Phone – According to [7] (ISS), PC-based phones utilize a software application that provides the phone with IP capabilities when using a PC. However, since the phone can only be used in conjunction with a PC, it is as vulnerable as the OS and applications installed on the computer.

5 VoIP Gateway Vulnerabilities

VoIP gateway technologies are also a potential weak point. When VoIP is used externally, gateway technologies convert data packets from the IP network into voice before sending them over a public switched telephone network. When VoIP is used internally, the gateways route packetized voice data between the source and the destination. The concern here is that such gateways can be hacked into by malicious attackers in order to make free telephone calls. The trick to protecting against this lies in having strict access-control lists.

As with traditional telephony, eavesdropping is a concern for organizations using VoIP; and the consequences can be greater. Because voice travels in packets over the data network, hackers can use data-sniffing and other hacking tools to identify, modify, store and play back voice traffic traversing the network. A hacker breaking into a VoIP data stream has access to a lot more calls than he would with traditional telephone tapping. As a result, "one of the big differences is that a hacker has a much higher probability of getting intelligent information" from tapping a VoIP data stream than from monitoring traditional phone systems [13].

6 VoIP Security Challenges for Service Providers

According to Verizon Business [9] the issues repeatedly seen by their security services assessment group falls into a number of categories they see repeatedly when assesses existing enterprise VoIP systems:

- The enterprise is not using a VPN to gain access to a VoIP environment
- There is poor support for access controls and passwords
- There is widespread use of unauthorized devices such as personal soft phones in enterprise networks and thereby thwart firewall rules.

The major security challenge for service providers can be viewed in terms of creating a way to transport VoIP packets safely across the provider's infrastructure.

Service provides need to insure a secure solution to transit the VoIP data across their network and deliver it to the customer interface. The CPE is often an enterprise border firewall, whose configuration rule set is a sensitive issue because it is the first line of defense in a corporate security system. Making this issue even more complex is the large number of competing VoIP protocols both standards based and proprietary.

Mandating specific IP phones is not practical because it limits choices available to the customer community and IP phone technology is moving rapidly. Another problem with IP phones is the ability to support a direct connection to the customer's site reveals a matter of IP destination address exposure.

Therefore, it can be seen that the principle issue that service providers need to address is the ability to securely transit VoIP traffic across their network, deliver it to the CPE, and insure that the VoIP traffic is not compromised as it transits the service provider's network. Service providers need a solution set that solves the fundamental problems associated with VoIP. Service provider must also insure they have the proper network infrastructure to accommodate these advances and deliver VoIP traffic safely across their infrastructure.

6 Mechanisms for Securing VoIP

Because of the time-critical nature of VoIP, and its low tolerance for disruption and packet loss, many security measures implemented in traditional data networks are simply not applicable to VoIP in their current form; firewalls, intrusion detection systems, and other components must be specialized for VoIP.

Firewalls, gateways, and other such devices help keep intruders from compromising a network. However, another layer of defense is necessary at the protocol level to protect the voice traffic. In VoIP, as in data networks, this can be accomplished by encrypting the packets at the IP level using IPSec, or at the application level with SRTP. This way if anyone on the network, authorized or not, intercepts VoIP traffic not intended for them (for instance via a packet sniffer), the packets will be unintelligible.

To implement VoIP securely, start with the following general guidelines, recognizing that practical considerations might require modification and adjustment:

- Put voice and data on logically separate networks.
- At the voice gateway, which interfaces with the PSTN, deny access to H.323, SIP, or Media Gateway Control Protocol (MGCP) connections from the data network.
- As with any other critical network management component, use strong authentication and access control on the voice gateway system.
- Choose a mechanism to allow VoIP traffic through firewalls. Various protocol dependent and independent solutions exist, including ALGs for VoIP protocols and session border controllers. Stateful packet filters can track a connection's state, denying packets that aren't part of a properly originated call.
- Use IPSec or Secure Socket Shell (SSH) for all remote management and auditing access. If practical, avoid using remote management at all and do IP PBX access from a physically secure system.

- Use IPSec in tunnel mode when available instead of in transport mode because tunneling masks the source and destination IP addresses, securing communications against rudimentary traffic analysis.

7 Conclusion

This paper has addressed VoIP security in terms of its configuration, risks, and potential usage by service providers who need to manage those risks. While VoIP is certainly a viable technology great care must be taken in its use and configuration. A number of VoIP specific recommendations conclude the paper.

However, beyond these specific recommendations, an enterprise must use a layered security architecture, which provides the most effective defense against VoIP attacks. Defensive layering must start beyond the enterprise border and originate in the service providers network to insure the secure transport of VoIP to the enterprise. Enterprises must continue to review their security posture in terms of risk mitigation, not risk avoidance because new technology and vulnerabilities will always arise alongside the new technology, like VoIP.

References

1. Barbieri R., Bruschi D., Rosti E. (2002). *Voice over IPsec: Analysis and solutions.* 8 th Annual Computer Security Applications Conference. pp. 261.
2. Berger T. (2006). *Analysis of current VPN technologies.* First International Conference on Availability, Reliability, and Security. Pp. 108-115.
3. Bednarz P. (2002). *How VoIP is changing the network security equation.* Retrieved October 16, 2006 from `http://www.commsdesign.com/design_corner/showArticle.jhtml?articleID=16505939`
4. Casteel J. (2005). *Sound Choices for VoIP Security* . Retrieved October 20, 2006 from `http://www.ebcvg.com/pdf/dl/sound_choices_voip_security.pdf`
5. Garg S., Singh N., Tsai T. (2005). *SRTP+: An efficient scheme for RTP packet authentication.* Retrieved Nov. 2, 2006 from `http://pubs.research.avayalabs.com/pdfs/ALR-2004-001-paper.pdf`
6. Greenstreet D., Scoggins S. (2005). *Building Residential VoIP Gateways: A Tutorial* Part IV: VoIP Security Implementation. Retrieved Oct. 23, 2006 from: `http://www.analogzone.com/nett0913.pdf`
7. Internet Security Systems (2004). Retrieved October 16, 2006 from `http://www.iss.net/documents/whitepapers/ISS_VoIP_White_paper.pdf`
8. Kuhn R., Walsh T. J. and Fries Stephen, (January 2005). *Security Considerations for Voice over IP Systems.* National Institute and Standards and Technology Publications. Retrieved, Oct. 21, 2006 from: `http://www.arcert.gov.ar/webs/textos/SP800-58-final.pdf`
9. Marsan C. (2006). *VoIP Security Services Taking Hold.* Network World, Vol. 23, Iss. 26, pg. 25, July 2006.

10. Scoggins S, *Implement Maximum Security for VoIP.* Reprieved October 25, 2006 from
http://www.eetasia.com/ARTICLES/2006APR/PDF/EEOL_2006APR03_RFD_NETD_SECD_TA.pdf
11. The Internet Society. (2004). *Request for comments: 3711 The secure real-time transport protocol (SRTP).* Retrieved October 31, 2006 from: http://www.ietf.org/rfc/rfc3711.txt
12. University of London Information Security Group (1998). Internet Protocol Security Flaws.
13. Vijayan, J. (2002) *VOIP: Don't Overlook Security.* Computerworld, Security.

Appendix A: Figures

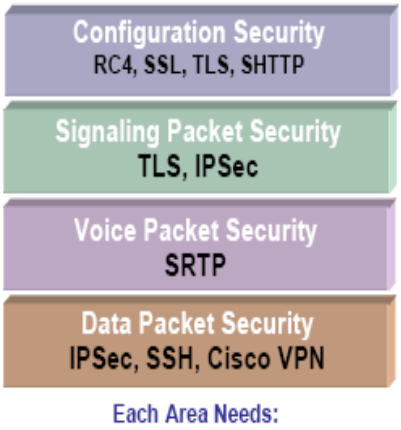


Fig. 1. VoIP Security Area

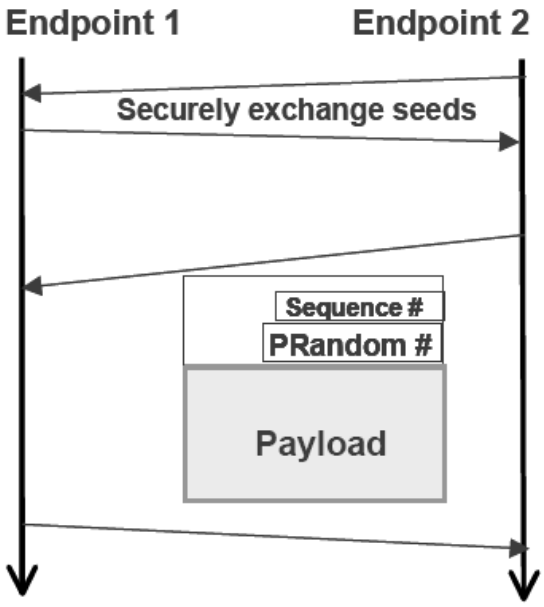


Fig. 2. SRTP+ Exchange for Technique 1

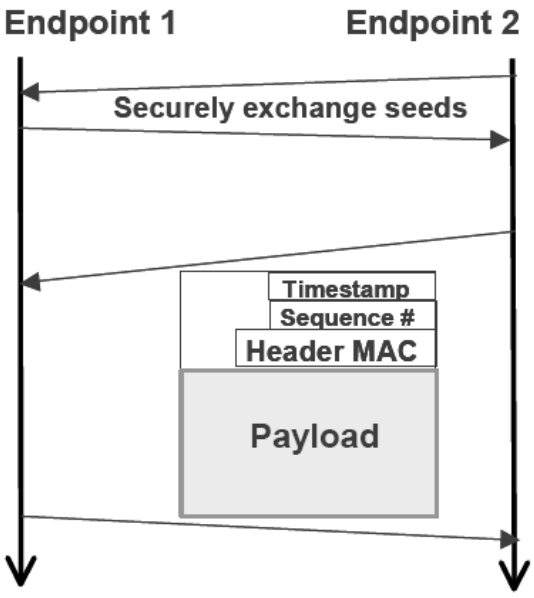


Fig. 3. SRTP+ Exchange for Technique 2

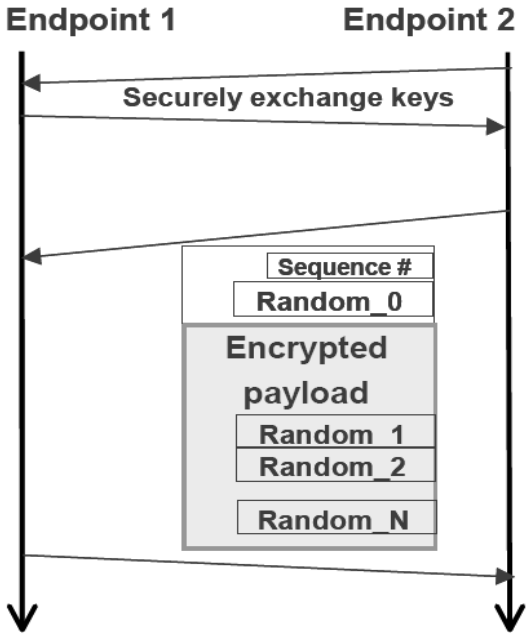


Fig. 4. SRTP+ Exchange for Technique 3

Proceedings of the International Multiconference on
Computer Science and Information Technology, pp. 641 – 649
ISSN 1896-7094
© 2007 PIPS

Anomaly Based Intrusion Detection Based on the Junction Tree Algorithm

Evgeniya Petrova Nikolova, Veselina Gospodinova Jecheva

Burgas Free University, Faculty for Computer Science and Engineering
62 Str. San Stefano, 8100 Burgas, Bulgaria,
enikolova@bfu.bg, vessi@bfu.bg

Abstract. The aim of this paper is to present a methodology for the attacks recognition during the normal activities in the system. Since the proposed approach uses the graphical representation method, we apply the junction tree algorithm (JTA). Some results from the accomplished simulation experiments are submitted as well.

1 Introduction

Intrusion detection is the process of monitoring the events occurring in a computer system or network and analyzing them for signs of possible incidents, which are violations or imminent threats of violation of computer security policies, acceptable use policies, or standard security practices. An intrusion detection system (IDS) is a system that automates the intrusion detection process. The basic functions of IDSs include recording information related to observed events, notifying the system administrators of important observed events and generating reports [1].

The task of an intrusion detection system (IDS) is modeled as a classification problem in a machine-learning context. It monitors system data (network or host) to distinguish intrusions and attacks or normal user activity. Based on the intrusion detection method IDS can be categorized into two main categories: misuse based and anomaly based. Misuse detection IDSs generate the alarms based on specific attack signatures. Despite of the fact they achieve a high level of accuracy, their major drawback is the little possibility of detecting novel attacks [10].

In this paper we direct our attention to the anomaly based IDS. A typical anomaly detection model will analyze data, compare to a preliminarily defined profile, run statistical analysis to determine if any significant deviation is found, and flag the event as a normal activity or an attack [2]. The advantage of anomaly based IDS is the potential to detect novel attacks or unknown attacks, variations of known attacks, and the ability to detect insider attacks or identity theft. Another benefit of this approach is that learning to describe normal activity can be automated.

Analyzing program behavior profiles for intrusion detection has emerged as a viable alternative to user-based approaches to intrusion detection [5]. Program behavior profiles are built by capturing system calls made by the program under analysis under normal operational conditions. If the captured behavior represents a compact and adequate signature of normal behavior, then the profile can be used to

detect deviations from normal behavior such as those that occur when a program is being misused for intrusion.

In this paper we are interested in finding methodology for the attacks recognition during the normal activities in the system, i.e. to discern the normal activity patterns from abnormal ones. For that purpose we apply the junction tree algorithm (JTA). The algorithm takes the form of message passing on a graph referred to as a junction tree, whose nodes are clusters of variables. Each cluster starts with one potential of the factorized density. By combining this potential with the potentials it receives from its neighbors, it can compute the marginal over its variables. Except finding marginal probabilities, JTA helps to find the most likely state of the distribution and to define the anomalies in the obtained sequence. For completeness, in Section 2 we give basic facts about JTA. In Section 3 we give a brief exposition of the applied methodology. Finally, some of our experimental results are provided in Section 4.

2 The Junction Tree Algorithm

Let $X=(X_1, X_2, ..., X_n)$ be a random vector with density p and $G=(V,E)$ be an undirected graph for which each node s has an associated random variable X_s. Denote $X_A=\{X_s, s\in A\}, A\subseteq V$. A subset of vertices all joined by edges is called clique.

The graph G can be used to enforce the random vector X (or on the distribution p) in different ways:

Markov property: X is Markov with respect to G if X_A and X_B are conditionally independent given X_S whenever S separates A and B.

Factorization: The distribution p factorizes according to G if it can be expressed as a product over cliques:

$$p(x)=\frac{1}{Z}\prod_C \psi_C(x_C),$$

where $Z=\int\prod_C \psi_C(x_C)dx_C$ and $\psi_C(x_C)$ is the potential for clique C.

Cluster nodes within cliques of graph with cycles to form a clique tree.

A clique tree with **running intersection property:** all nodes on the unique path between clique nodes C_1 and C_2 contain the intersection $C_1 \cap C_2$, is called a **junction tree.**

The goal of the junction tree algorithm (JTA) is to define a potential representation of the graph such that, coupled with a suitable algorithm to modify these potentials, will result in the marginals of individual or groups (cliques) could be obtained directly from the modified potentials.

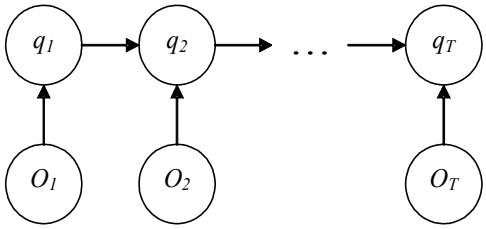
Algorithm: ([9])

- Given an undirected graph G. Form a triangulated graph $G^\wedge$ by adding edges as necessary. (Chord is a link joining two non-consecutive vertices of a loop. An undirected graph is triangulated if every loop of length 4 or more has a chord.)
- Form the clique graph (in which nodes are cliques of the triangulated graph).
- Extract a junction tree.
- Run sum-product on the resulting junction tree.

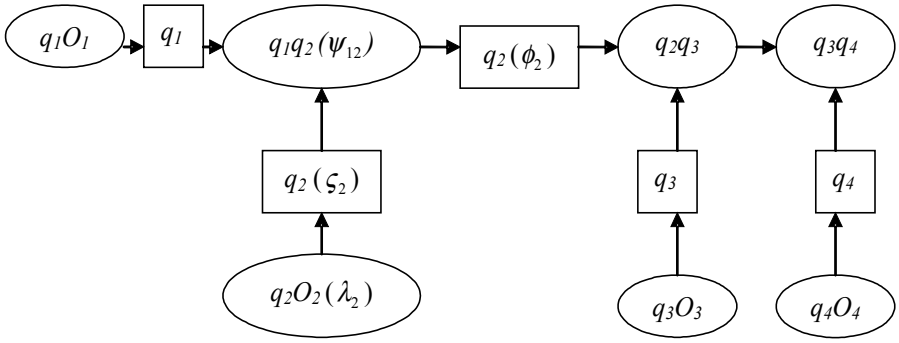
Except finding marginal probabilities, JTA helps to answer another natural question: what is the most likely state of the distribution? Running JTA produces the marginal probabilities. Consider a modified distribution p^* in which we wish to re-weight the states, making the higher probability states exponentially more likely than lower probability states. This can be achieved by $p^* \propto (p)^{\beta}$, where β is a very large positive quantity. This makes the distribution p^* very peaked around the most-likely value of p. In the JTA we need to carry out summations over state. Since in the limit $\beta \to \infty$ only the most-likely state will contribute, hence the summation operation can be replaced by a maximization operation.
More details not introduced here could be found in [7] and [8].

3 Outline of the Methodology

Let's consider the system work in the discrete moments of time $t=1, 2, .. , T$, when the system occupies some of the following N states: $S_1, S_2, ..., S_N$. Let $O=(O_1, O_2, ... O_T)$ is the observation sequence at the moments $t=1, 2, ..., T$. We applied the junction tree algorithm to the graphical model representing the system work, which is shown below



where $Q=(q_1, q_2, ..., q_T)$ is the state sequence at the moments $t=1, 2, ..., T$. Each q_t is one of the elements of the set $S = \{ s_1, ..., s_N \}$. Consider the following junction tree



First we initialize all clique and separator potentials with 1s. Then for each node in the original graph, we multiply its conditional probability distribution onto the clique to which it is assigned. If we assume initially that the nodes, containing observations

O, are hidden, the λ potentials at the node q_tO_t will be all 1s and the potentials along the node q_tq_{t+1} will be as follows:

$$\Psi_{1,2}(i,j) = P(q_2 = s_j \,|\, q_1 = s_i)\, P(q_1 = s_i)$$
$$\Psi_{t,t+1}(i,j) = P(q_{t+1} = s_j \,|\, q_t = s_i)$$

3.1 Forward steps

We initialize the potentials with evidence

$$\lambda_t^*(i) = P(O_t \,|\, q_t = s_i)\ ,\ \lambda_t^*(i) = \frac{P(O_t, q_t = s_i)}{P(O_t)} = \frac{\varsigma_t^*(i)}{1} = \varsigma_t^*(i)\ .$$

Run the forward algorithm, we get the following potentials:

$$\psi_{t-1,t}^*(i,j) = P(q_{t-1}, q_t, O_{1:t}) = P(q_t = s_j \,|\, q_{t-1} = s_i)\, P(q_{t-1} = s_i, O_{1:t-1})\, P(O_t \,|\, q_t = s_j)$$
$$= \psi_{t-1,t}(i,j)\, \frac{\phi_{t-1}^*(i)}{1}\, \frac{\varsigma_t^*(j)}{1}.$$

Marginalizing yields:

$$\phi_t^*(j) = P(q_t = s_i, O_{1:t}) = \sum_i \psi_{t-1,t}^*(i,j).$$

3.2 Backward steps

In the backward steps, we find that the potentials are

$$\psi_{t-1,t}^{**}(i,j) = P(q_{t-1} = s_i, q_t = s_j, O_{1:T}) = P(q_{t-1} = s_i \,|\, q_t = s_j, O_{1:t})\, P(q_t = s_j, O_{1:t}, O_{t+1:T})$$
$$= \frac{\psi_{t-1,t}^*(i,j)}{\phi_t^*(j)}\, \phi_t^{**}(j).$$

Marginalizing yields:

$$\phi_t^{**}(j) = P(q_t = s_j, O_{1:T}) = \sum_i \psi_{t-1,t}^{**}(i,j).$$

If we propagate messages from children to their parents in the rooted junction tree, we have by induction after the forwards steps (after collecting to root), $\psi_{t,t+1}^*(i,j) = P(q_t = s_i, q_{t+1} = s_j)$, $\phi_t^*(i) = P(q_t = s_i)$ and all the other potentials are unchanged (all 1s). After the backward steps we have

$$\psi^{**}_{t-1,t}(i,j) = P(q_{t-1} = s_i, q_t = s_j),$$

$$\varsigma^{**}_t(i) = \lambda^{**}_t(i) = \phi^{**}_t(i) = P(q_t = s_i).$$

As a result from the JTA we obtain the most likely state sequence during the examined period of time. Finally we compute the state transition probabilities for the result state sequence.

Each algorithm step at both forward and backward directions requires $O(M^2)$ operations. Since we have a total of T moments of time, the entire computing procedure will require $O(TM^2)$ operations, where T corresponds to the number of cliques and M^2 is the cardinality of the clique state space.

4 Experiments

The experiment data were obtained from Unix-based and Sun SPARCstations system examination during some period of time. The experimental data include normal user activity traces of some privileged processes that run with administrative rights (synthetic ftp, synthetic lpr, login, named and xlock) and their child processes patterns, as well as intrusion data. The input data files are sequences of ordered pairs of numbers, where the first number is the process ID (PID) of the process executed, and the second one is the system call number. Forks are considered as separate processes and their execution results are examined as normal user activity.

The methods for pattern generation are described in [4] and [5]. They have proved that short sequences of system call traces produced by the execution of the privileged processes are a good discriminator between the normal and abnormal operating characteristics of programs. The experimental results have also confirmed that short sequences of system call traces are stable and consistent during program's normal activities.

Experiments based on the described algorithm, were accomplished using corresponding software in C++. Based on the normal user activity patterns we evaluated the state transition probabilities during the system work in attack absence. The experimental data also contain abnormal user activity patterns, generated by some common intrusion tools (sunsendmailcp, lprcp, buffer overflow, some Trojans, etc.). We considered these data as current system activity data, which could be examined by the JTA. We used a slide window with length T =10 to cross the traces of current user activity. Given an unknown observation sequence, we find the most likely states sequence, as well as its state transition probabilities. Comparing the obtained state transition probabilities with the state transition probabilities of the normal user activity patterns we detect the intrusions presence. Some of the results from this comparison for the enumerated processes are summarized in Figures 1, 2, 3, 4 and 5.

In order to evaluate the false alarms rate we applied a method proposed in [6]. Figures 6, 7, 8, 9 and 10 represent the false positive rate (i.e. classifying normal as an attack) for the following processes: synthetic ftp, synthetic lpr, login, named and xlock. The false positive rate (i.e. classifying normal activity as an anomaly) for the

described method is 8,6%, and the false negative rate (i.e. classifying anomaly as a normal) in this case is 3%. The presented anomaly detection method could accurately verify a given unknown sequence to be normal or anomalous with 88,4% accuracy. Table 1 contains the values of the false positive rate, the false negative rate and the accuracy for the mentioned privileged processes.

Table 1 The false alarms rate and the algorithm accuracy

Process	False positive rate	False negative rate	Accuracy
synthetic ftp	5%	1%	94%
synthetic lpr	9%	4%	87%
login	7%	3%	90%
named	6%	1%	93%
xlock	16%	6%	78%

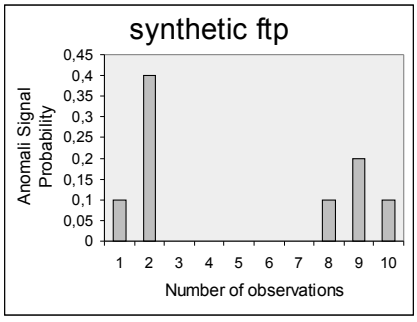


Figure 1

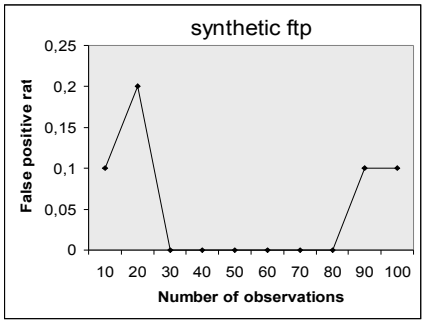


Figure 6

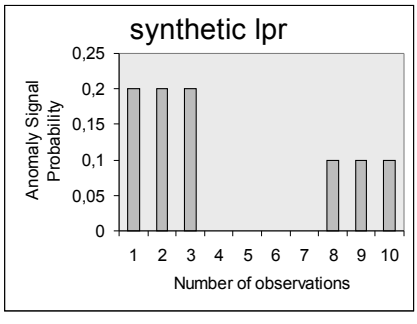


Figure 2

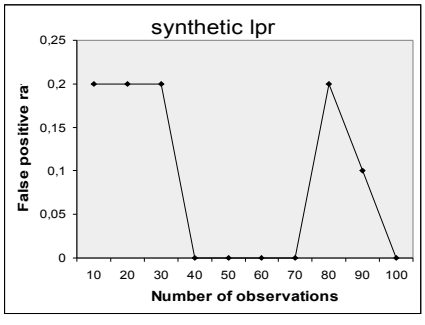


Figure 7

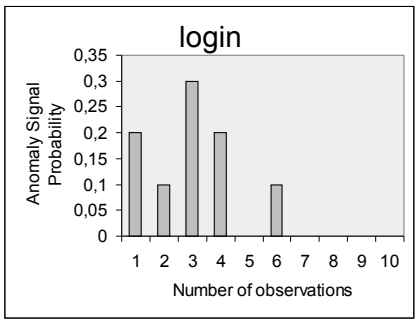


Figure 3

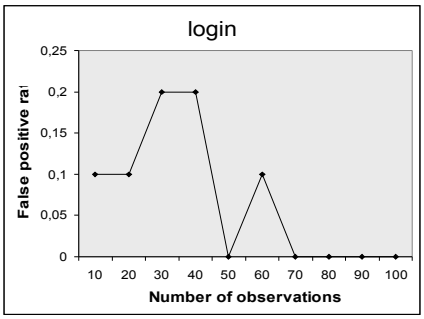


Figure 8

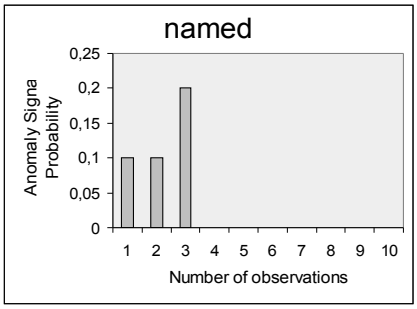


Figure 4

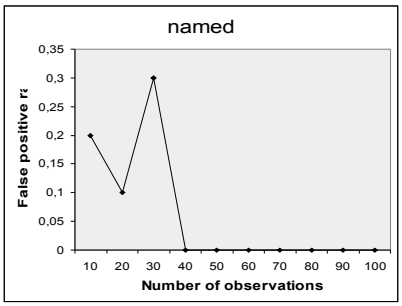


Figure 9

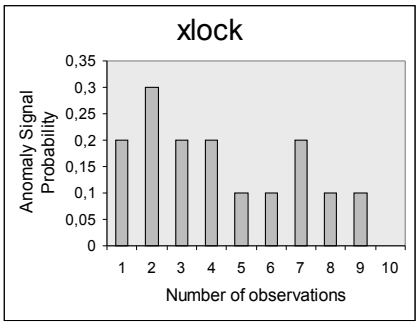


Figure 5

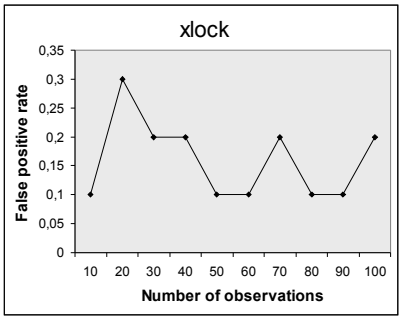


Figure 10

5 Discussions

This paper investigated the capabilities of the JTA in anomaly based intrusion detection. As a result of the JTA we found the joint distribution on the non-obvious variables in a clique with all the obvious variables clamped in their obvious states.

Then calculating the likelihood is easy since we just sum out over the non-obvious variables of any converged potential. From the received sequence it is easy to determine if it is anomalous, comparing to the state transition probability.

The major advantage of graphical representation models, such as the one in JTA, over many other types of predictive models, such as neural networks and decision trees, is that unlike those "black box" approaches, the graph structure represents the inter-relationships among the data set attributes. Human experts can easily understand the network structures and if necessary can easily modify them to obtain better predictive models. Other advantages of graphical representation models include explicit uncertainty characterization, fast and efficient computation, and quick training. They are highly adaptive and easy to build, and provide explicit representation of domain specific knowledge in human reasoning frameworks. Moreover, graphical representations offer good generalization with limited training data and easy maintenance when adding new features or new training data.

Another advantage of the anomaly-based approach in general is its potential to detect novel attacks, insider attacks or account theft. The drawback of the described method is its considerable computational price, since it has $O(TM^2)$ complexity. Since M^2 is the cardinality of the clique state space, its value could be significant in the case of a large system. Another disadvantage of the anomaly based IDS in general is the creation of the database containing the user profiles, which could be a task of considerable difficulty and requires some period of time the system is unprotected.

6 Conclusions and Future Work

The present paper proposes a method of intrusions access recognition in the anomaly based IDS. We applied a graphical representation model, precisely the Junction tree algorithm, as well as some probabilistic methods to distinguish the normal user activity from the intrusion one. Some experimental simulations were accomplished as well. The purpose of future work could be the algorithm optimization, as well as some different probabilistic method applications and the comparison of the obtained results.

References

1. Abraham A., Thomas J., *Distributed Intrusion Detection Systems: A Computational Intelligence Approach, Applications of Information Systems to Homeland Security and Defense,* Idea Group Inc. Publishers, USA, Chapter 5, 2005, pp. 105-135.
2. Chan P., M. Mahoney, M. Arshad, *Learning Rules and Clusters for Network Anomaly Detection,* Workshop on Statistical and Machine Learning Techniques in Computer Intrusion Detection, George Mason University, Technical Report CS-2003-06, 2003.
3. Forrest S., S. A. Hofmeyr, A. Somayaji, T. A. Longtaff, *A Sense of Self for Unix Processes,* In Proceedings of the 1996 IEEE Symposium on Security and Privacy, IEEE Computer Society Press, Los Alamitors, CA, pp.120-128.
4. Forrest S., S. A. Hofmeyr, A. Somayaji, *Intrusion detection using sequences of system calls,* Journal of Computer Security Vol. 6, 1998, pp. 151-180.

5. Ghosh A. K., A. Schwartzbard, M. Schatz, *Learning Program Behavior Profiles for Intrusion Detection,* In Proceedings of the 1st Workshop on Intrusion Detection and Network Monitoring, pp. 51–62, 1999.
6. Hoang X. D., J. Hu, P. Bertok, *A Multi-layer Model for Anomaly Intrusion Detection Using Program Sequences of System Calls,* 11th IEEE International Conference on Networks (ICON 2003), Sydney, Australia, 2003.
7. Jensen F. V., F. Jensen, *Optimal Junction Trees,* Uncertainty in Artificial Intelligence, 1994.
8. Lauritzen S. L., *Graphical Models,* Oxford Science Publications, 1996.
9. Lauritzen S. L., D. J. Spiegelhalter, *Local computations with probabilities on graphical structures and their application to expert systems (with discussion),* Journal of Royal Statistical Society, Series B, 50(2), 1988, pp. 157–224.
10. Tran T. P., T. Jan, A. J. Simmonds, *A Multi-Expert Classification Framework for Network Misuse Detection,* From Proceeding (544) Artificial Intelligence and Soft Computing, ISBN 0-88986-610-4, 2006.

Proceedings of the International Multiconference on
Computer Science and Information Technology, pp. 651 – 660
ISSN 1896-7094
© 2007 PIPS

Model based code generation for fast-deployment security applications

Gyula Simon, László Szabados, András G. Tóth

Department of Computer Science, University of Pannonia
H-8200, Veszprém, Egyetem u. 10, Hungary
simon@dcs.vein.hu

Abstract. A diverse set of sensors and actuators are key components of integrated security systems, which provide protection against various types of attacks and threats. Based on the type of the protected objects and environment the sensor/actuator components can be completely different, and the control logic, which makes decision based on the sensor readings, must be configured to the actual scenario. Short-lifetime security systems require fast and cost effective deployment but the safety requirements are still high. In this paper an architecture and a corresponding model-based code generation scheme is proposed, which provides easy and fast deployment for security applications with various sensory needs.

1 Introduction

Wireless security solutions are gaining more and more popularity because of their flexibility, convenience and clean installation, especially in home security and alarm systems. Sensor networks, however, may have an important role in complex, integrated security systems as well, where protection is provided against a wide range of threats, like physical attacks (e.g. robbery, flood), logical attacks (e.g. unauthorized access or data theft), or human/organizational threats (e.g. suspicious behavior of an employee or negligence of safety regulations). The application of wireless sensor networks is especially desirable in fast deployment scenarios.

In this paper a security architecture and a corresponding model-based code generation scheme is proposed, which provides easy and fast deployment for applications with various sensory needs.

1.1 Fast Deployment Scenarios

In certain applications the lifetime of the system is comparable with the time the setup of the system requires. Such application scenarios can be concerts, expos, conferences or exhibitions, requiring security services for a limited time only. Building the system from predefined elements and templates fastens the construction of the system. Using easily configurable elements, which are easy to deploy fastens the setup of the system.

A very expensive and time consuming part of system setup is the deployment of sensors. Special security needs require special hardware; especially the wiring of the sensors can be costly and difficult. E.g. fire alarm sensors require special fireproof

wiring not easily fitted into the given environment (e.g. in a protected 500-year old Town Hall drilling will not be welcome). There are more and more security solutions based on wireless sensors, where only the power supply must be provided (to charge the batteries on the sensors), or even this can be omitted if batteries with sufficiently long lifetime are used. These sensors communicate on various wireless channels, e.g. using WiFi/802.11, Bluetooth/802.15.1, or Zigbee/802.15.4 standards. The emerging Zigbee, especially tailored for communication between sensors provides an ideal architecture for mobile security applications.

Security applications require special security measures on the communication channels as well. While in the case of wired sensors usually a few wires are used and their intactness can easily be checked with proper design of the electronic components, systems with wireless sensors must be protected against new types of attack, e.g. jamming (detectable and a countermeasure alarm can be set), or message forgery (message authentication can provide strong protection).

1.2 Related work

In the field of intelligent sensors much research has been done because of the large industry demand. As a result, the *IEEE 1451 Standard for Smart Transducer Interface for Sensors and Actuators* is under preparation [11, 9]. Smart sensors not only measure a physical quantity (e.g. voltage or resistance) but convert the measured value to the desired unit of the physical attribute being measured. IEEE 1451 compliant smart sensors and instruments can work together with ease in a plug and play fashion. A future section of the standard (1451.5) will cover wireless communication issues as well (WiFi, Bluetooth, and Zigbee).

Smart sensors can identify themselves via the built-in Transducer Electronic Datasheet (TEDS). The TEDS contains basic data that identifies the sensor (e.g. manufacturer ID, serial number), extended data to store all the electrical and physical properties (e.g. min/max range, sensitivity), user data (e.g. sensor location) and sensor specific template data.

Model-based code generation has many appealing properties vs. handcrafted coding: the systems can be formally specified, formal verification procedures are available, and through automatic code generation coding errors can be eliminated. Based on the used initial model and the transformation tools different approaches have been used. Graph transformation is a general tool to convert the initial model into the desired format [3]. State machines [1] and hybrid models [2] proved to be rich and efficient modeling environments for automatic code generation. Programming with design patterns also has been successfully automated using application specific models [4]. Template-based code generation is widely used due to its simplicity and feasibility [10].

2 Security System Architecture

The architecture of a sensor network based security system can be seen in Fig. 1. The wireless sensors sense their environment and send measurement data through the ad-hoc wireless network formed by the sensors (and possibly actuators) to the sensor

gateway(s). The gateway provides interface between the wireless network and the Security Center. The actuators can also be operated through the wireless network: they receive their control signals through the sensor gateway and the ad hoc network. Note that intelligent wireless sensors and actuators are much more alike than their wired counterparts: sensors in addition to sending measurement values also receive information (e.g. control signals from the central system) while the actuators also send messages (e.g. status info) to the central station, besides receiving commands. In the following we handle sensors and actuators in the same way and for simplicity we refer to them as sensors.

The sensors can be configured to send measurement data in different ways: a simple solution is polling, when the central station asks the sensors to send their measurements whenever they are needed. Autonomous periodic data sending can also be provided (e.g. send data in every second). The event-based mechanism sends data only if "interesting" events occur (e.g. door opened, smoke sensed, temperature too high, etc.) In addition to measurement messages control/monitor messages are also traveling through the network: the central station can configure sensors, and they can send status information (the most important being *heartbeat* messages reporting that the sensors are still functional).

Most traditional sensors used in security systems measure physical quantities, e.g. thermometers, door controllers, proximity readers, fingerprint readers, smart card sensors, cameras; but much more sophisticated "sensors" can also be used in integrated security environments, e.g. picture recognition subsystems or log-based attack detection/prediction systems, just to name a few. To provide the ease of wireless deployment, the sensors (simple and sophisticated sensors as well) are connected to the standard Smart Sensor Interface (see Fig.1), which provides various middleware services on networking and application-specific levels:

- ad-hoc network management,
- message routing/authentication,
- interfacing the sensor to the network,
- providing intelligent sensor services (e.g. periodic data sending),
- intelligent sensor configuration.

The smart sensor interfaces are inexpensive components containing a simple (8-bit) microcontroller and a radio chip (e.g. Chipcon2420 [5], Xbee [12], em250 [6]). This simple inexpensive architecture (similar to smart dust approach) enables the use of many, possibly redundant sensors providing a rich sensory field.

The wireless network is connected to the Security Center (which in fact can be centralized or distributed) through the Sensor Data Gateways. Through these gateways the Security Center unit can receive data from and send data to the network. Redundant gateways can be provided to increase fault tolerance. Naturally, different network types can be connected through different gateways.

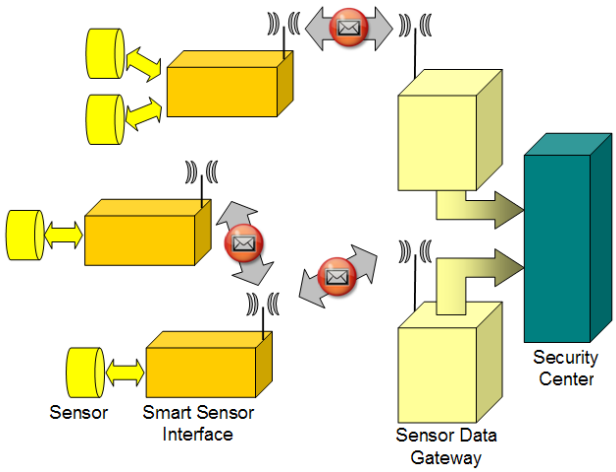


Fig. 1. Structure of the wireless-enabled security system. The Smart Sensor Interfaces provide interface to the sensors and through the ad-hoc wireless network forward messages to the Sensor Data Gateways. The gateways translate sensor data to the Security Center and also forward control signals back to the network

Various applications require different amount and types of sensors. To facilitate fast deployment and fast system setup smart sensor interfaces are standard units: the hardware interface provides connection possibilities to various types of sensors (e.g. AD/DA, digital inputs, serial port, I^2C, etc.), and most software components are also standard (e.g. network-specific components, generic sensor handling). Based on the actual requirements, sensor-specific software components will be generated and configured automatically based on sensor descriptions, which will discussed in Section 3.

The Sensor Data Gateway provides a standard interface to the Security Center. The interface is built from generic components and sensor-specific parts. The generic components are not changed; only sensor specific information is added compile-time or run-time to the gateway to enable handling of various types of sensors. The sensor specific information may be added to the gateway in two ways:

- "electronic data sheet" approach: each sensor stores its own attributes and sends this information during network setup phase to the smart sensor gateway and thus to the central processing unit. With this solution all the sensor-specific information is stored in the sensors and can be accessed when the sensor is connected to the network.
- "handbook" approach: the sensor's attributes are written in a "book" (implemented as a file). Large amount of information can be stored in the handbook; no network communication is required and the limited resources on the sensors are not wasted. The book can be accessed by various components, e.g. smart sensor interfaces can be generated based on the sensor description, or the sensor data gateway can be configured using relevant data stored in the handbook.

Although both solutions are technically possible, in the proposed security system architecture there is no specific need for the more sophisticated "electronic data sheet" approach: various system components can easily access sensor-specific

information from the handbook and this approach suits better the limited resources of the sensor interfaces. In the following examples the handbook approach will be used.

3. Sensor Description

The description of the sensor contains all the necessary information to handle the sensor's hardware as well as rules to convert and use the sensor's measurement values. The *functional description* contains high level information for the gateway and the central processing unit, while the smart sensor interface is generated from the *platform specific* hardware and software description.

To ensure uniform handling and portability of the data, XML file format is used, shown in an XML editor in Fig. 2. In the example a simple temperature sensor is defined for a MicaZ mote [8] using TinyOS operating system and nesC [7]. The description contains high-level (functional) and low-level (platform specific) information as well.

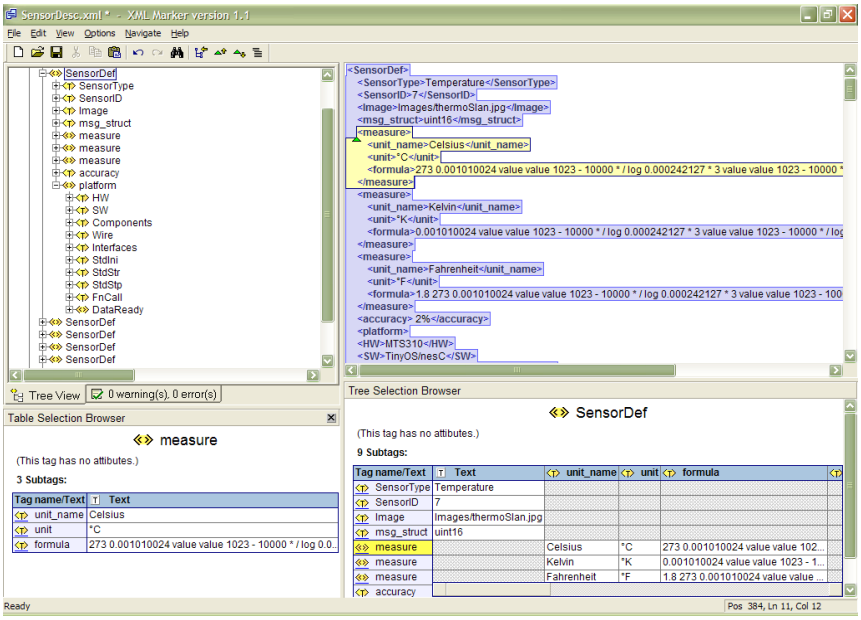


Fig. 2. The description of a temperature sensor in xml format.

3.1 Functional Description

The final consumer of the measurement data is the central processing unit. The sensor data gateway converts raw data to standard formats understood by the central unit, based on the functional description of the sensor. These data fields identify the sensor type, the data format, accuracy, etc. This field also contains conversion information (how to interpret the bit series of the sensor) and display options for graphical representation as well. The tags identify the sensor type (Temperature) with code "7"

to identify this type of measurements, measurement units (Celsius, Kelvin, and Fahrenheit), the accuracy of the sensor, and the conversion formulae to generate the required units from the raw data. The optional image fields can define various graphical representations for the sensor.

3.2 Platform Specific Description

The platform specific sensor description defines low level interfacing to the sensor. It includes hardware specific information (e.g. at which port the sensor can be accessed), usage rules (e.g. how the measurement values can be read), architectural information (e.g. module descriptions, wiring), and any other relevant platform-specific information (e.g. constants, functions, any necessary glue code). Fig. 3 illustrates two fields in the temperature sensor's description for the Mica mote platform using TinyOS operating system and nesC language: wiring and interface definition.

The wiring information inserts the module to the smart sensor interface, using the defined interfaces. The init, read, and stop fields describe how to start, operate, and stop the sensor, respectively (shown in Fig. 2). Naturally these fields must be platform specific and have to contain information for the code generator

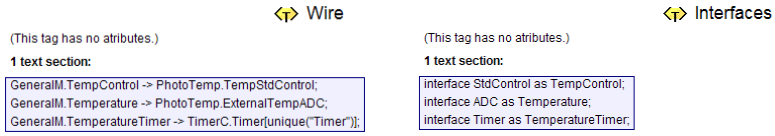


Fig. 3. Interface description of the temperature sensor on Mica mote HW and TinyOS/nesC environment

4. Code generation

The code generation is based on the sensor descriptions and code templates, as shown in Fig. 4. The Smart Sensor Interfaces and the Sensor Data Gateway are implemented on their specific platforms, using meta tags in the code to identify sensor specific code segments to be generated. The translator, using the Sensor Description generates and inserts the appropriate code into the templates.

The translator is platform independent and can understand generic commands like *replace* code segment and *make new instance* of a code segment. Using these meta-commands (recursively, if necessary) the translator can generate the required Smart Sensor Interface and Sensor Gateway Code. The codes then can be compiled and downloaded to the desired target using standard make utilities and the system is ready to run.

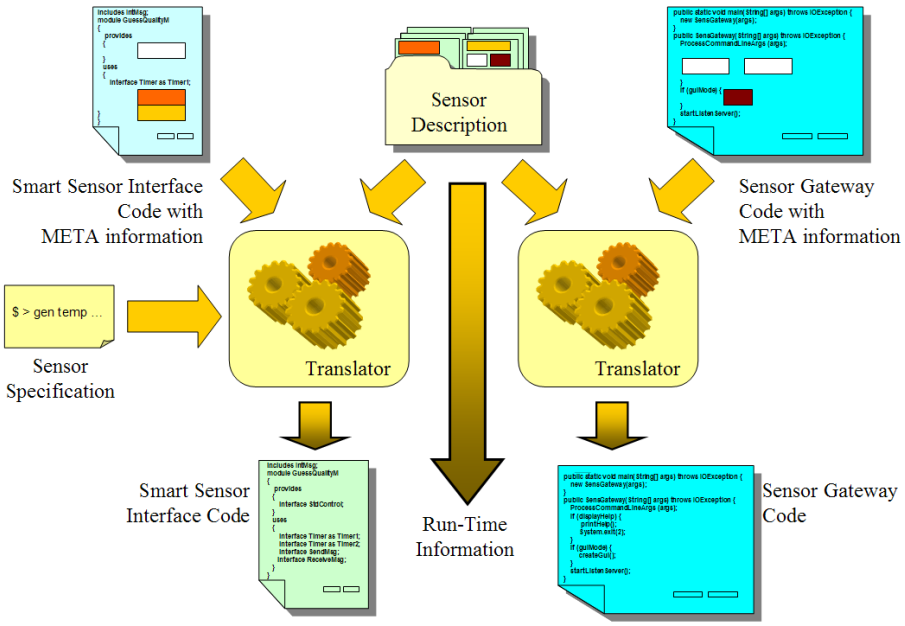


Fig. 4. Code generation scheme

The following example shows a Smart Sensor Interface template code segment with meta tags. The <MakeTimer> and </MakeTimer> tags instruct the translator to generate a code segment for all sensors, while the <$...$> tags are replacement instructions:

```
<MakeTimers>
event result_t <$SType$>Timer.fired(){
    <$FnCall$>
    return SUCCESS;
}
</MakeTimers>
```

The code is generated for two embedded sensors, a light and a temperature sensor. The generated code segment includes two instances of the event handler, one for each sensor, with the correct function calls and references:

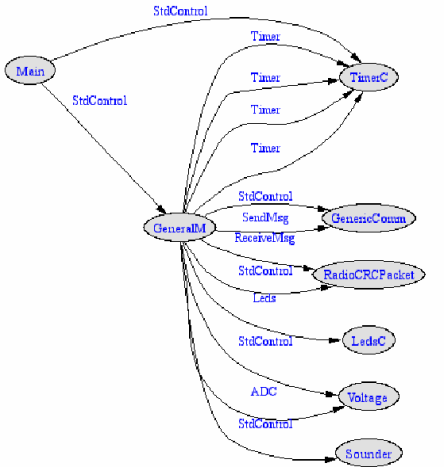
```
event result_t TemperatureTimer.fired(){
call TempControl.start();
call Temperature.getData();
return SUCCESS;
}

event result_t LightTimer.fired(){
call PhotoControl.start();
```

```
call Light.getData();
return SUCCESS;
}
```

5 Results

We implemented a wireless sensor network based security surveillance system as a case study. The system contains MicaZ motes as sensors [8] running TinyOS/nesC [7], and a PC serves as gateway. The motes can control several sensors, e.g. temperature and light sensors, accelerometers, magnetometers, etc. The Smart Sensor Interface code running on the motes was generated automatically for each unit, implementing various sensors. The gateway in the example does not require compile-time code generation; it uses the functional sensor description run-time (e.g. for unit conversion).



(a) (b)

Fig. 5. Software components and their connections in the TinyOS representation of the Smart Sensor Interface. (a) shows the framework, while (b) is a generated application containing a photo sensor and a two-axis accelerometer. Internal changes in the components are not shown

The Smart Sensor Interface code is illustrated in Fig. 5 showing its component structure, automatically generated by the TinyOS documentation tool. The initial template code is shown in Fig. 5(a). The figure shows the standard components (e.g. Main) and their connections to each other through interfaces (e.g. StdControl). In Fig.5(b) the structure of the generated code is shown when a light sensor and a two-axis accelerometer are used. New components handling the sensors were automatically added to the code (PhotoTemp, Accel, and three new instances of TimerC), and were connected to standard components via appropriate interfaces. Note that Fig. 5 shows the structural modification of the code, while the change of the internal content of component GeneralM was illustrated by the examples in Section 4.

The GUI of the surveillance application is shown in Fig. 6. The Tree Manager window displays the topology of the ad-hoc network, along with the pictograms of the available sensors; e.g. Mote 5 contains a microphone, a light sensor, and a thermometer. The GUI allows the configuration, control, and query of individual sensors. The system can automatically identify new sensors in the network and can detect sensor failures as well. The gateway also generates and forwards events (in the example to the surveillance GUI, in a real security system to the Security Center) when a sensor appears in or disappears from the network. The active motes and their installed sensors can be accessed and controlled through the gateway, as shown in the control window of Mote 5.

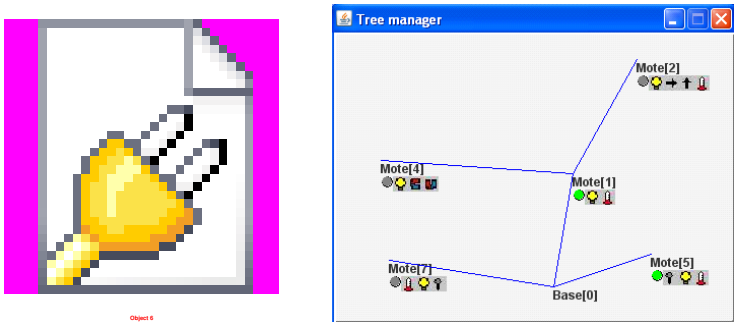


Fig. 6. Elements from the GUI of the surveillance application. Smart sensor interfaces providing ad-hoc wireless access to sensors were generated from platform-specific sensor description. The gateway uses functional-level sensor information to display temperature values (see the defined units and conversion formulae in Fig. 2)

Summary

In integrated security systems wireless solutions will be increasingly important because of their flexible, fast and convenient installation. Such systems can provide protection against various threats and attacks, with safety requirements similarly high to that of the wired solutions. In fast deployment systems, where fast and cost effective deployment is crucial, the application of wireless sensors can radically shorten the deployment time of the system.

In this paper a wireless-enabled security architecture and a corresponding model-based code generation scheme was proposed. The architecture handles various types of sensors in a similar manner through the generic Smart Sensor Interface. The interface provides the necessary middleware services to enable the access of sensors via the ad-hoc network, including routing, state monitoring, message authentication, and naturally, sensor interfacing. Wireless Gateways connect the Security Center (not discussed in this paper) to the ad-hoc network with generic sensor interfaces.

The fast as error-free development and deployment of security applications is eased by the proposed automatic code generation of various sensor-related components. The Smart Sensor Interfaces and the Wireless Gateways can automatically be generated from sensor description information. The sensor descriptions can also be used run-time by higher level components.

The architecture and the code generation scheme were illustrated in a case study, where an ad-hoc sensor network provided measurements. The code for the Smart Sensor Interfaces was generated automatically based on platform-specific sensor description. In the test application the usage of functional sensor description information was also illustrated.

Acknowledgment

This research was partially supported by the Hungarian Government under contract NKFP2-00018/2005.

References

1 Agrawal A., Simon G., Karsai G.: *Semantic Translation of Simulink/Stateflow models to Hybrid Automata using Graph Transformations,* Electronic Notes in Theoretical Computer Science, Vol. 109 (2004) 43–56.
2 Alur R., Ivancic F., Kim J., Lee I., Sokolsky O.: *Generating embedded software from hierarchical hybrid models,* SIGPLAN Not. 38, 7 (2003) 171–182.
3 Bredenfeld A., Camposano R.: *Tool integration and construction using generated graph-based design representations,* Proceedings of the 32nd ACM/IEEE conference on Design automation conference, San Francisco, CA (1995) 94–99.
4 Budinsky F. J., Finnie M. A., Vlissides J. M., Yu P. S.: *Automatic Code Generation from Design Patterns* IBM Systems Journal archive, Vol. 35, (1996) 151–171.
5 *Chipcon Zigbee RF Tranceivers,* `http://www.chipcon.com`
6 *Ember Zigbee Chips,* `http://ember.com`
7 Gay D., Levis P., von Behren R., Welsh M., Brewer E., Culler D.: *The nesC language: A holistic approach to networked embedded systems,* In *Proceedings of the ACM SIGPLAN 2003 Conference on Programming Language Design and Implementation* (2003). 1-11.
8 Hill J. L., Culler D. E.: *Mica: A Wireless Platform for Deeply Embedded Networks,* IEEE Micro 22, 6 (2002) 12-24.
9 *IEEE 1451.0, Standard for a Smart Transducer Interface for Sensors and Actuators– Common Functions, Communication Protocols, and Transducer Electronic Data Sheet (TEDS) Formats,* IEEE Instrumentation and Measurement Society, TC-9, The Institute of Electrical and Electronics Engineers, Inc., New York, N. Y. 10016.
10 iMatix Corp: Template-based Code Generation. *Technical White Paper,* `http://legacy.imatix.com/pub/info/codegen.pdf`
11 Lee K.: *IEEE 1451: A Standard in Support of Smart Transducer Networking,* Proceedings of the 17th IEEE Instrumentation and Measurement Technology Conference, Baltimore, MD, May 1-4, 2000, Vol. 2, pp. 525-528.
12 *Maxstream Xbee Modules,* `http://www.maxstream.net`

Proceedings of the International Multiconference on
Computer Science and Information Technology, pp. 661 – 670
ISSN 1896-7094
© 2007 PIPS

Dealing With Network Security in Academic Institutions – a Case Study

Ivan Doležal[1], Jiří Grygárek[1], Ondřej Jakl[2], Karel Krečmer[2]

[1] VSB-TU Ostrava, IT Services, 17.listopadu 15,
708 33 Ostrava-Poruba, Czech Republic
{ivan.dolezal, jiri.grygarek}@vsb.cz
[2] Institute of Geonics of the Czech Academy of Sci.,
Studentská 1758, Ostrava – Poruba, Czech Republic
{jakl, krecmer}@ugn.cas.cz

Abstract. The paper deals with a real-life experience of the authors with their efforts at a radical security improvement of the academic computer networks that they administer at a large university and a medium-sized research institute. The solution, which started in 2004 and is still going on, has been based on hardware multi-threat security appliances with high throughput. The requirements on them included a combination of general purpose Intrusion Prevention System, HTTP/FTP antivirus capabilities and firewall functions. In particular, we describe our experience with the appliances of the FortiGate series, which have been deployed as the best solution available.

1 Introduction

VŠB-Technical University (VSB) and Institute of Geonics of the Czech Academy of Sciences (IG) are adjoining, but independent academic institutions located in Ostrava, Czech Republic. Although incomparable in size (VSB is a large university having about 25 thousand students, whereas IG is a medium-sized research institute, the staff of which is about 100), they are capable to cooperate and in 2004 responded to a call for projects declared by the Czech Ministry of Education and proposed a joint infrastructural project concerning network security in the environment of research and development (R&D) organizations. The proposal has been accepted for funding. This contribution presents some intermediate experience obtained by the authors during the more than three years of the work on the project, namely with managing security in academic environment with help of specialized network protection systems.

1.1 Computer and Network Users in Academia

In general, when dealing with security in academic organizations, we have to consider their specific characteristics. Most of them can be derived from the diversity of users, who can be roughly split into three categories, each of which deserves quite different approach:

- By standard users we mean typically management and technical staff ensuring the everyday operation of the institution as such. For their routine work, most of them need regular computing environment, i.e. an office computer and corresponding software (usually text processor, spreadsheet, mail/groupware client, (restricted) Internet access and interface to some local economic information system), which are now highly standardized. The administration of this environment can be extensively automated. While this type of users is prevalent in most non-academic organizations, they are in minority in academia.
- Research workers are typical computer and network users in R&D institutions, but represent the most diversified group. Depending on their field of study, they may need completely different computing environments, individual approach and specific services, e.g. software development, handling of large data sets or computer support for special laboratory equipment. Those users appreciate full freedom for their research and may be annoyed by security measurements posed by network administrations. As a rule, they are involved in inter-institutional collaboration and take advantage of remote access, data transfers, recently also Grid computing and other forms of network services. It may be difficult to enforce some regulations on them. On the other hand, the researchers are generally very competent and disciplined computer users.
- In this respect, the students, the third category of academic computer and network users, are the very opposite. They are probably the most problematic group to handle. From nature, they like to experiment and explore. Malicious activities must be anticipated, since some of them, such as the students of computer science, may have good background for efficient hacking. But also “normal” students can disable networks through legal or illegal activities generating large traffic. The great number and permanently changing classes make the handling extraordinary difficult.

It is the two latter categories of users that pose the greatest demands on network administration. VSB is an institution where all three categories of users meet in a heterogeneous collection, i.e. the network and security administration is particularly challenging, whereas at IG the missing students make the situation considerably easier.

1.2 Local Situation

In general, the starting point of our security-improving efforts was quite similar on VSB's and IG's networks, since their state was result of the former rather spontaneous and chaotic development, when the security was not adequately considered.

At VSB, the local network was protected by a set of stateless filters on the border router. Antivirus and/or firewall was installed on some of the employees' computers. Student machines were entirely out of our control.

At IG, all networked personal computers were protected by antivirus software and the e-mail traffic was checked on the mail server for viruses and spams. The network as a whole was protected by a Linux machine instrumented as firewall (*iptables* with *fwbuilder*, later *shorewall* user interface), which we found rather awkward for occasional administration. More importantly, Fast Ethernet network interfaces of the firewall machine limited the traffic throughput to 100 Mbit/s, wasting the capacity of the institute's Gigabit uplink to the Internet.

In both environments, the network operation had to count on conscientiousness of users who were urged to protect their workstation themselves by local antivirus and firewall software. Especially at VSB this was very difficult to enforce since its organizational units (faculties, departments, etc.) are quite independent.

1.3 Objectives

Our general aim was to improve the security of the local network environment of VSB and IG without seriously affecting the users, especially in the R&D sphere.

In particular, we intended to solve the long-standing issues of the network perimeter, i.e. at the boundary of the local and public network (Internet). The idea was to implement some security apparatus, that would

- be aware of as many application layer protocols as possible (e.g. H.323, SIP)
- block viruses, worms and spyware from HTTP, FTP and mail traffic,
- implement IPS – not only to protect the local network from Internet attacks, but possibly also to quickly identify potentially harmful computers inside LAN

The great difference between the VSB's and IG's requirements on the solution was related to the throughput: Based on long-term observations and knowledge of the traffic characteristics, for VSB we had to find an appliance that would be able to handle simultaneously about 400 000 connections, 800 Mbits/s firewall throughput and 70 Mbits/s antivirus throughput, i.e. performance, that was on the cutting edge of the actual offerings. For IG, firewall's speed in network traffic control was not an urgent point. Here, the network administration would appreciate the ease of use and as far as possible maintenance-free operation. Nevertheless, both organizations preferred to build their solutions on similar and compatible devices of the same vendor.

2 Towards Efficient Network Security

The realization of our plans was not straightforward. Especially at VSB, we considered simultaneously several possible conceptions of the solution and tried to match them with products available on the market.

2.1 Kick-off Solution

We addressed the vendors of security systems to offer an appropriate appliance that would be positioned on the border between the Internet and the local area network and that would fulfill our requirements specified in section 1.3 at a very high speed.

At the end of 2004, we could find just some three vendors in the Czech Republic who claimed to meet the requirements. In the testing phase, it was difficult to verify and compare the capabilities since those complex devices need some study to be set up. Already in this point we appreciated the FortiGate specimen which surpassed the other products in the ease of deployment. More importantly, the family of Fortinet appliances fitted best our functional needs at an acceptable price. In fact, we could not find any serious competitor at that time.

FortiGate [1] is the brand name of the multi-threat security appliances for real-time network protection made by Fortinet Inc. [2]. These are ASIC accelerated network devices, the capabilities of which include statefull firewalling, shaping, Intrusion Detection/Prevention System (IDS/IPS), antivirus (AV) and antispam functionality. FortiGate units can work on L2 (link layer) of OSI model and allow both command line and web-based management. Some more characteristics can be found in Table 1.

Table 1 . Performance of selected FortiGate products (data of 2004)

Model	Interfaces [# Fast, Giga E.]	Firewall [Mbits/s]	Antivirus [Mbits/s]	Sessions [#]
FG60	4 FE	70	15	50k
FG200A	8 FE	150	30	400k
FG400A	2 GE, 4FE	500	100	400k
FG800	4 GE, 4 FE	1000	150	400k
FG1000	2GE, 4 FE	1000	~200	600k
FG3600	6 GE, 1 FE	4000	~400	1000k

The estimated life expectancy of the deployment was five years – that's why we inclined to buy the most powerful models affordable. Especially at VSB, to get the best performance and reliability, we decided to build a cluster of two FortiGate FG3600 (cf. Table 1) and configured them to run in the *active - active* high availability mode. This mode truly load balances the TCP sessions (antivirus, IPS) between the units in the cluster – other kinds of the IP traffic flow over the primary unit only. So far, we have verified that this configuration can handle throughputs of more than 500 Mbit/s. To prevent changes in network addressing, we decided to explore a unique *L2 transparent mode* feature, where the device acts as a network bridge.

IG similarly needed a scalable and robust multi-threat firewall solution (i.e. a Gigabit router-firewall) to protect its LAN against attacks, which would be dimensioned at least for a five-year service. Respecting the compatibility with VSB, we chose two FG800 units from the FortiGate product family (Table 1). Its characteristics perfectly suited our demands on capabilities and performance and we quickly got accustomed to its friendly user interface. Whereas for VSB, the pair of FG3600 units above was just the starting point for further development to meet its needs (cf. next section), IG's network, from the performance point of view, can be served just by one unit until now. That is why the two-box cluster runs in the *active - passive* high availability mode (the primary unit handles the traffic and the secondary unit is in the hot-standby state).

2.2 Response to Network Pervasiveness

Since 2005, wireless networking (WiFi) has been deployed at VSB, especially for student needs. Unfortunately, private notebooks, although becoming part of the campus network, are out of our control. Their users are just required to prove the identity either with IEEE 802.1x EAP-TLS (“with a user certificate”) or by connecting through VPN. Students' notebooks can also get connected through Ethernet with 802.1x authentication, but this technology has never reached the expected popularity. The number of VPN connections is growing fast. To protect

these new network entry points in a topologically clean way, we purchased and deployed two pieces of Fortinet's FG400A and FG1000 units.

The number of mobile devices, however, has been continually growing among regular staff, too. Many such devices are not being connected only through the well-defined set of network sockets and/or WiFi – the same entry points are often shared by "safe" desktops and "promiscuous" notebooks. Consequently, the whole concept of the fortified perimeter is deprecated. We considered the following advancements:

- 802.1x authentication of all connected devices – not only this is not a security solution *per se*, furthermore this feature would have to be supported by all of the network switches which in turn would require massive upgrade.
- Cisco NAC (*Network Admission Control*) – high TCO plus overcomplicated manageability would be the indisputable cons of this option.
- Leaving the end-user devices without special supervision and minimizing the impact of potential incidents by a strict network segmentation.

We decided for radical changes in the spirit of the last alternative: We sliced the network and created "internal perimeters" that correspond to more than 120 smaller subnets. Many of those VLANs are already connected to the network backbone through a FortiGate security appliance, each of them representing one security domain; yet more are to be connected.

This slicing approach was applied at every site, where VSB (its branch) is situated. These sites are interconnected by redundant L3 links,[1] that's why every locality was equipped with its own FortiGate unit, operating in L2, as depicted in Fig. 1. At present, VSB employs 13 FortiGate units in total, what might make it the greatest Fortinet customer in the Czech Republic.

The size of IG is comparable to the size of a smaller branch of VSB, so the issues discussed above could be handled in analogy with any VSB branch here. In fact, the IG network is divided into two subnets considered as separate security domains. We try to minimize the risk involved in accessing IG's network from machines out of our control in various ways. For example, the most urgent service, remote mail access, has been satisfactorily solved through the IMAP server and users' needs to remotely access IG's servers or their local machines is possible through the SSH protocol and tunneling. In general, we still refuse to connect foreign machines directly to our network. However, moderate and stable number of users makes it possible to have even their notebooks under reasonable control, i.e. at least to ensure that they are protected by local antivirus and firewall software. Similarly, computers making use of the IG's wireless connect, which has been designed primarily for visitors and conference participants, are kept separated and the firewall handles them in the same way as ordinary machines on the Internet. Nevertheless, VPN, Eduroam and other modern services that could be beneficial for R&D are certainly under consideration. Our experience shows that FortiGate appliances are very valuable for the development of our LAN.

1 Not shown in Fig. 1

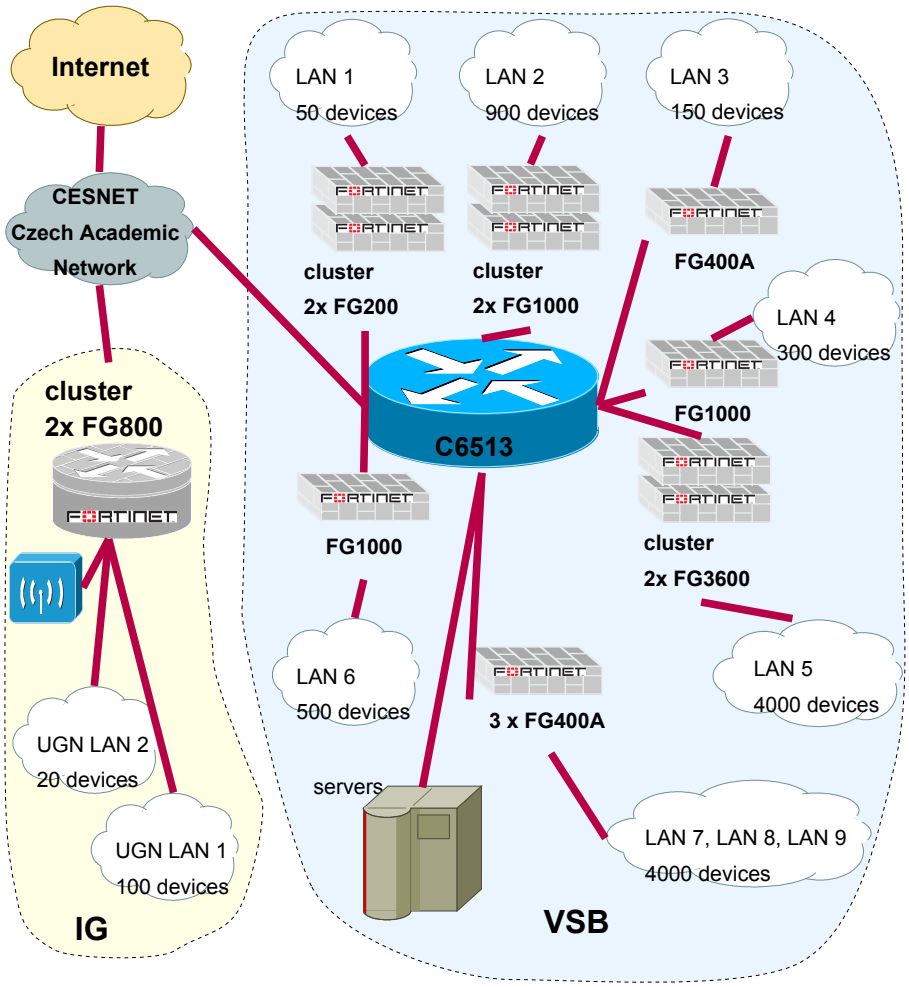


Fig. 1. VSB and IG network topology

4 Real Life Experience

In this section, we present some concrete experience with the FortiGate deployment.

4.1 Is Less More?

Before the decision was made to acquire FortiGate multi-treat security appliances in 2004, *Snort* [4] among other alternatives was also evaluated. Its pro was the public availability of open source signatures. Its con was lack of them. Commercial solutions

are the true opposite: plenty of them, but no one (including Fortinet support) knows what they exactly do. If an event is triggered, it is often very difficult to investigate if it is true positive or false positive. We decided to use about three quarters of the signatures only for detection (the actual ratio changes over time).

4.2 Tuning Up IDS/IPS

The Fortinet company sells a rather expensive auxiliary appliance called FortiAnalyzer for logging, analyzing and reporting information from FortiGate devices. However we learned to get raw information from the devices through Unix legacy *syslog*. Here is an example of a logged attack:

```
Jun  6 09:17:52 pixla1 date=2007-06-06 time=09:17:52
devname=Pixla1 device_id=FG36002805033272 log_id=0419070000
type=ips subtype=signature pri=alert vd=zamestnanci
serial=1237491673 attack_id=107610115 severity=high
src=158.196.154.46 dst=83.10.110.7 src_port=4163 dst_port=111
src_int=vl-206 dst_int=vl-2206 status=detected proto=6
service=111/tcp user=N/A group=N/A
ref="http://www.fortinet.com/ids/ID107610115" msg="rpc_decoder:
RPC.Fragment.Overflow"
```

In the VSB environment, we log over 300 000 lines a day on a regular Linux PC. Since it would be hardly possible to analyze all this information manually[2], the authors wrote a simple Perl analyzer *CORELLA* ("correlation"). This tool allows us to

- aggregate records with the same source IP address,
- match information on leased addresses from other systems involved (RADIUS server, DHCP server, Active Directory in the future),

all that in the heterogeneous university network infrastructure consisting of Cisco devices, Fortigate appliances and various open source software.

An example of information processed by *CORELLA* follows. In this example we can see an IPS aggregation that clearly points out a computer infected by a virus:

```
158.196.49.27        kolc408c.vsb.cz     55 x netbios:
MS.Windows.ASN.1.Bitstring.Overflow drop_session
prnnk222a.vsb.cz  gps.vsb.cz  pcj331n.vsb.cz pcj331d.vsb.cz
... (~20 more destinations follow)
```

It is not only the signature itself that is significant, but also the number of computers attacked from the source IP as well as the fact that some of them cannot be resolved by reverse DNS. A very useful strategy is that (a) whenever possible, the computer should have a static IP address that can be resolved and that (b) unused IP addresses cannot be resolved. The "drop_session" string indicates that suspicious packets were dropped and the connection was violently terminated.

A few signatures can themselves reliably indicate an infected machine:

2 In small organizations like IG there will be no problem with manual analysis of logs through the standard FortiGate web interface.

```
158.196.158.46       kocour.vsb.cz     25 x netbios: LSASS.445
[Reference: http://www.fortinet.com/ids/ID102039611]
drop_session  ts1-l06.vsb.cz  ts1-l12.vsb.cz  ts1-l16.vsb.cz
vpns157.vsb.cz  ts1-l03.vsb.cz  vpns252.vsb.cz  vpns248.vsb.cz
```

Traffic anomaly detector logs can show us suspicious behavior of a PC. However, to determine the difference between zero-day attacks and a curious student playing with *nmap*, personal investigation is needed:

```
158.196.229.36       kolc411c.vsb.cz    7249 x anomaly:
portscan, 1001 > threshold 1000 [Reference:
http://www.fortinet.com/ids/ID100663398] clear_session
vpns241.vsb.cz  vpns368.vsb.cz  158.196.198.1  158.196.198.157
158.196.198.164  ... (~30 more destinations follow)
```

Peer-to-peer (P2P) networks are "special case". Although Fortinet did produce some signatures to positively identify and possibly block this traffic, it is hide-and-seek game. The P2P clients try to mask the traffic. A totally obscure way to identify P2P network activity is by finding a different-than-expected signature. In our opinion it is quite unlikely that all of the downloaded files in the following example contained the BMP attack – we guess the traffic originated from P2P[3]:

```
158.196.xxx.xxx      xxx.vsb.cz    525 x misc:
MS.Windows.Media.Player.BMP.Buffer.Overflow [Reference:
http://www.fortinet.com/ids/ID101974234] dropped  1olo-
172.dialup.vol.cz  ap-veseli-hutnik.netopen.cz  gw-
contactel.teltech.cz  ppp141.brno.tiscali.cz  ovj-84-242-94-
58.nat.karneval.cz  ip-85-160-88-194.eurotel.cz  wov-1-
46.802.cz  85.132.180.86  ...  (~20 more destinations follow)
```

In the following log fragment, *CORRELA* identified users of systems that were assigned dynamic IP addresses (although we aim at avoiding dynamic address assignment, we were forced to use it in favor of WiFi and VPN):

```
158.196.68.42        vpnp042.vsb.cz      1 x misc:
IRC.clientToServer.communication.JOIN detected  ost069
_kolb410b.vsb.cz_09:13
```

False positive alarm that we cannot really handle: A typical example caused by a network printer:

```
158.196.154.191      prnj303a.vsb.cz      1 x snmp_decoder:
SNMP.Restricted.OID detected  pcn304d.vsb.cz
```

3 Btw: This signature was later disabled by the manufacturer himself.

4.3 FUD[4] Unconfirmed (Not Really)

There was a nightmare for the project team that some IPS rule would totally paralyze traffic from some key application. Fortunately we encountered this only in a few cases of minor importance – Debian update or IBM Software Update. Both cases were promptly identified and resolved by an IP address exception.

Low latency multicast support was one of our important requirements. During the selection four years ago we could not find any other vendor than Fortinet that would meet our demands in this respect. Most appliances did not let the multicast pass through at all, whereas FortiGate units operated smoothly with latency under 1 ms.

4.5 Latest Update

In spring 2006 new FortiGate firmware version 3 with major improvements was released. One of them was the P2P filtering capability. After applying the new filter set we could recognize some 80% decrease of the P2P traffic, but we cannot exterminate it completely (yet). Instant messaging antivirus is another highly remarkable feature and URL blocking is a "cherry on the pie": It is suitable for preventing users from accessing phishing web pages.

With the new firmware, the licensing policy was also modularized. Now, different features and various kinds of support must be purchased separately. Devices offer many more features (antispam, web pages classification, etc.), but in this paper we referred only to the features we have personal experience with.

5 Conclusion

And humility is an important quality. Especially if you're wrong a lot...
of course, when you're right, self-doubt doesn't help anybody, does it?
-- House M.D.

When we look back after nearly four years, we can find that the FortiGate suite employed in our networks did improve manageability of security incidents both by reducing their number with AV/IPS and improving their detection with traffic analysis. This has been achieved without negative impact on the diversity of academic users in our institutions. Here are some additional comments.

The fear of the "bottleneck effect" was not confirmed.

The wise choice of an unknown appliance and ignoring Barnumesque advertisement has paid off. By the way, one of the most famous competitive products – Symantec SGS [6] – has been abruptly discontinued recently.

Even though prices for support represent significant annual expense (and are slightly growing), you can always give up buying the prefabricated signatures (support renewal) and add new signatures by hand using FortiGate's simple configuration language. This is a tremendous difference from other vendors.

However, we also ran into a few problems:

4 Fear, Uncertainty and Doubt

- We experienced and reported three specific major bugs in the appliance (their explication is behind the scope of the paper). The time necessary for their resolution was in our opinion unacceptably long for a commercial customer (the toughest issue is being solved by Fortinet for seven months already and is continuing without outcome).
- Although Fortinet has been promising IPv6 support from the very beginning, neither IPS nor AV is functional for IPv6 traffic up to this day.
- Even though the appliance can provide its administrator with some log data, an advanced administrator will always want more data – be it more information on signatures or detailed statistics of accepted/rejected packets/sessions. Just by reading *FortiAnalyzer Administration Guide* we got an impression that purchasing this expensive device would not help either.
- If running more than two FortiGate appliances, you may want to buy FortiManager – another rather expensive appliance, which keeps the FortiGate configurations "in sync". This is enormously important for setting up IPS rules. Although we could work our some "do-it-yourself" tools for that purpose, too[5].

Acknowledgment: This work was supported by the project No. 1N04035 of the Ministry of Education of the Czech Republic.

References

1. FortiGate - Installation and Configuration Guide. Fortinet Inc., 2004-7
2. Fortinet. .`http://www.fortinet.com/`
3. Juniper Networks IDP 50/200/600/1100. `http://www.juniper.net/`
4. Snort - the de facto standard for intrusion detection/prevention. `http://www.snort.org/`
5. Strebe, M., Perkins, C.: Firewalls and Proxy-Servers. Comp. Press, Brno (2003) (in Czech)
6. Symantec SGS Security Appliance. `http://www.symantec.com/`
7. Wikipedia. `http://en.wikipedia.org/`

5 The boxes can be controlled over SSH.

Proceedings of the International Multiconference on
Computer Science and Information Technology, pp. 671 – 680
ISSN 1896-7094
© 2007 PIPS

A Joint Meta-Linguistic Taxonomy of Intrusion Detection and Testing/Verification

Krzysztof Brzezinski

Institute of Telecommunications, Warsaw Univ. of Technology
Nowowiejska 15/19, Warsaw, Poland
kb@tele.pw.edu.pl

Abstract. The current research into intrusion detection makes only the marginal use of results obtained by the community concerned with formal verification and testing. To harmonize the ideas and methods used by these two separate disciplines, we develop a discourse space (a taxonomy), in which the linguistic problems common to testing (in particular passive testing) and intrusion detection are captured. It is shown that the currently accepted main intrusion detection paradigms can be described and explained by this taxonomy.

1 Introduction

We deal with distributed reactive systems, such as telecommunications networks. In this area, particular research communities develop their own methodology. The community rooted in traditional telecommunications networks with their standardized signalling protocols has strongly contributed to the development of formal methods and industry-strength tools, e.g., for rigorous testing [1]. The Internet community is guided by "rough consensus and working code", and is generally more inclined to adopt *ad-hoc* solutions. Each community tackles common problems in their own way, without seeking to adopt the results of the other one, and even without being fully aware that these *are* common problems. This leads to chaos and waste of resources.

A step towards raising mutual awareness and harmonizing the approaches followed by both communities was made in [2], where *intrusion detection* (ID), an important issue in "new" telecommunications, was re-defined as a verification problem. For a real system under examination (an *implementation*), the suitable verification technique is *testing*. Due to the pragmatics of intrusion detection, in [2] a particular testing concept and technique - *passive testing* was considered. It was shown how a standardized test language TTCN-3 (*Testing and Test Control Notation* [3]) can be effectively used in this doubly unconventional application: for passive testing (and not for *active* testing, for which it was originally developed), and applied to ID (which was not originally envisaged as one of its possible application areas). The experiments showed the viability of the approach, but failed to provide the clear understanding of *what* was actually being checked (tested), and *how* the applied technique related to the ideas and elements present in other intrusion detection systems. A suitable taxonomy, or (more modestly) a *discourse space*, was thus needed. Superficially, the obvious candidate could be the popular classification of ID techniques into *misuse detection*, *anomaly detection*, and the *specification-based* approach. However, in our opinion it is too arbitrary to be convincing. Instead, we generalize work on testing and intrusion detection, by defining a more detailed discourse space that *joins* and *harmonizes* the concepts of both these research areas.

The taxonomy is centered around the linguistic issues: the ways of expressing and reasoning about "interesting" (or malicious) behaviours. It may thus be described as being placed at the *meta-linguistic* level.

The rest of the paper is organized as follows. First we briefly survey the two disciplines that are to be joined and harmonized: formalized verification/validation and testing in chapter 2, and intrusion detection in chapter 3. Chapter 4 forms the core of the paper – it presents the structure and salient features of our discourse space. In chapter 5 the discourse space is used to position other research approaches, including the experiment reported in [2]. Chapter 6 concludes the paper with a short summary.

2 A framework for reasoning about behaviours

In the late (cut-in and operation) phases of its life-cycle, a system is no longer a formal object – it is an *implementation* (a hardware / software artefact). Its qualitative and quantitative behavioural properties can only be measured empirically. *Testing* and *monitoring* are two distinct [4] techniques of such measurements. The aim is to evaluate the system behaviour with respect to its design specification (*verification*) or to any set of properties that represents the expectations of a particular party (*validation*). For the purposes of formalized verification and validation, in telecommunications an implementation is usually considered to be a *black box*, or a *grey box* (a set of black boxes connected via visible interfaces), that exhibits a certain *behaviour*. The actual internal design of an implementation is assumed to be unknown, and thus any access to particular elements of this internal structure is, by definition, not presupposed. To the ID community this may seem an unnecessary and adverse self-restriction, as, apparently, the details of many implementations appear to be well known. However, this belief is also only an assumption, and the black box approach is central to the powerful methodology that we intend to re-use.

To reason about the behavioural properties, including those related to an intrusion, a suitable *linguistic support* (a language / formalism) is required. In this paper we are generally concerned with "high-level programming languages" for Intrusion Detection Systems (IDS). A language is needed primarily to express a *model* (a reference), against which the observed behaviour will be assessed. The existence of this model is generally presupposed both for intrusion detection [5] and for testing. The kind of testing, with which we seek similarities, is *model-based testing* [6], and in particular *passive testing* (a.k.a. *model-based monitoring* [7]).

A distributed system S is composed of a set of N>1 active, non-distributed (at a given abstraction level) entities, communicating over passive channels. An *instance* of behaviour results from an *execution* (a *run*) of a system. It yields a *trace* - a finite sequence (a total order) of events that are *observable* in the system: $\sigma_S \in E_S^*$. *System behaviour* $B_S \subset E_S^*$ is the set of behaviour instances that S can ever manifest. A behavioural *specification* P is any set of traces that, for some reason, is considered "interesting". Let us take P as a *reference model* of system behaviour. A generic verification problem is to check whether a system is in a certain relation (correspondence) to its reference model, which we denote by $B_S \sim P$.

In *testing-based verification*, a system is considered to be an implementation I whose detailed system behaviour B_S^I is *a priori* unknown (if it were known, testing would not be necessary at all). Whether $B_S^I \sim P$ holds, now has to be established solely

by inspecting the *observed instances* of behaviour $B^I_{obs} \subseteq B^I_S$. In the testing parlance, I is referred to as an *Implementation Under Test* (IUT). It is usually embedded in a *context* [8] - an environment that is not intended to be verified, but from which, for various reasons, the IUT cannot be separated (*operational* context: e.g., "the rest" of a network; *testing* context: the means to access the IUT for testing purposes, which usually obstructs its controllability and observability). IUT in its testing context is called *System Under Test* (SUT). Testing consists in evaluating the behaviour of an IUT in particular, purposely chosen circumstances (conditions). The usual paradigm of *active* testing considers this activity as consisting of three elements: (a) stimulating an IUT (through a testing context) to place it in one of the pre-defined conditions; (b) observing the behaviour of IUT in *this* situation (again, through the testing context); and (c) comparing the observation with a model, in order to issue a *verdict*. The "pre-defined conditions" are related to *test purposes*. In *passive* testing, the test system does not force an IUT towards test purposes - instead, it waits for an IUT to "place itself" in one of the conditions, in which its behaviour would be *symptomatic*.

Among many reasons for considering a particular set of traces as a specification, one is to establish a *design specification* P_D for a prospective implementation I. A typical P_D can be found, e.g., in standards for protocols (as developed by ETSI, ITU-T, IETF, etc.). P_D is normally expected to define *all* the intended (allowed) behaviour manifestations. $B^I_S \sim P_D$ thus means: I *implements*, or *conforms to*, its design specification P_D. If $\neg(B^I_S \sim P_D)$ then I is said to be *faulty.* A specification P that is *not* a design specification may be intentionally incomplete. In this case the semantics of the ~ relation may be better characterized by the notion of *satisfaction*.

3 Intrusion detection

The approaches to the construction of an IDS are usually divided into [9, 10, 11, 12]:

- *misuse detection* (also known as *signature-based*), in which an IDS looks for patterns of behaviour that correspond to pre-specified attack signatures;
- *anomaly detection*, in which an IDS looks for deviations from the "normal" behaviour (or traffic profile), described by statistical measures; what is taken as "normal" is established autonomously in the learning phase, and the decision on flagging an intrusion is taken basing on pre-defined similarity coefficients / metrics;
- *specification-based* approach, in which an IDS detects deviations from a pre-defined correct / legitimate logical behaviour.

With regard to the topological relation to a system under observation and the source of processed data, intrusion detection systems have been broadly classified as:

- *host-based* (HIDS), in which events that constitute the observed behaviour come directly from the active component(s) of a system;
- *network-based* (NIDS), in which the behaviour of a system is inferred from the messages observed in the network.

To bring some order to numerous linguistic approaches and solutions used for intrusion detection, [13] classifies the languages used in this field. In the sequel we deal only with the *event* languages (used for constructing the elementary "events" from the observed data) and *detection* languages (a.k.a. *attack* languages) that provide

mechanisms and abstractions for identifying the manifestation of an attack. The following languages of this kind are characteristic of the Internet community: Regular Expressions for Events (REE), Behavioural Monitoring Specification Language (BMSL) [11], self-constructed Finite State Automata, Parallel Environment (PE) grammars to specify trace policies [10], Attack Language for State-based Intrusion Detection (STATL) [13], Auditing Specification Language (ASL), Trace / Filter Description Language (TDL/FDL) [7], the Bro language, Network Event Recognition Language (NERL) [14]. All these languages are being assessed and compared agaist each other using arbitrary arguments and subjective experiences, without a deeper reflection on *what*, and *how*, these languages really express. The taxonomy presented in the following chapter is meant to serve as a basis for more organized comparisons.

4 The joint discourse space for testing and intrusion detection

As a starting point, we adopt the following assumptions and conventions:

- intrusions are specific behaviour manifestations of implementation I ;
- the (non-intrusion-related) behaviour of I is specified in a, normally standardized, design specification P_D, from which I is supposed to have been derived;
- intrusion manifestations can be (directly *or indirectly*) captured and specified in a reference specification P, which is developed specifically for the purpose of ID;
- ID is a testing-based verification problem of establishing the truth of a hypothesis: $B_S^I \sim P$, where ~ is an "intrusion present" (or, dually, "absent") relation.

4.1 Primary Dimensions

A. ***The semantics of P***

A reference specification P may express (specify):

(1) *abnormal* (disallowed, irregular, hostile) behaviour;

(2) *normal* (allowed, intended, correct) behaviour.

A RS that enumerates the correct behaviour(s) may be understood to also indirectly specify incorrect behaviour (any behaviour that is not directly specified). However, the consequences for a detection mechanism are in both cases quite different.

B. ***The logical aim of testing***

The direction of reasoning, (a verification / testing paradigm) may be [8]:

(1) to *falsify* a hypothesis: $B_S^I \sim P$;

(2) to *confirm* (verify) the hypothesis.

In testing, a working hypothesis is that I (and thus B_S^I) is *correct* w.r.t. P. Note that, in the tradition of the model-checking community, this statement could be dually re-formulated as: " P is a *correct* model of I ". This alternative formulation fits the setting of natural sciences (where a model is constructed so as to explain natural phenomena, and can serve as a *theory* until empirically falsified). However, it is counter-intuitive in the area of engineering, where artefacts are developed *from*

models, and at some point in the development process a model is assumed to be correct "by definition".

The hypothesis can be falsified or confirmed (verified). The conceptual validity of these operations has been discussed by philosophers (including, e.g., Karl Popper). These discussions have seeped into the testing theory: it is said that testing aims at revealing faults (falsifying a hypothesis), and not at proving correctness of an implementation. In the theory of temporal logic of behavioural properties, division is made into *safety* properties (those that are "finitely refutable"), and *liveness* properties that hold *iff*, despite any possible finite failure, there is always a (possibly infinite) continuation of behaviour such that the formula defining a property is satisfied. Intuitively, only the safety properties are "testable" [15].

Similarly, in the context of enforcing a *security policy* by means of execution monitoring (EM) mechanisms that stop a misbehaving program, it was shown that only a subset of safety properties is EM-enforceable [16]. This result was further extended by allowing a misbehaving execution to be modified on-line, which allows for some non-safety properties to be handled as well. Note that stopping or modifying an execution considered as "insecure" corresponds to the idea of an Intrusion Prevention System (IPS); apparently, this observation has not been exploited yet.

C. ***Class (aspect) of behaviour expressed by P***

A reference specification *P* may express:

(1) *qualitative* aspects of behaviour (functional, logical, i.e., based on the order and presence/absence of events in a trace);

(2) *quantitative* aspects behaviour (performance-related, i.e., involved with counting, timing, integrating, statistical measures, etc.).

These two aspects call for different methods and approaches. TTCN was initially designed for qualitative conformance tests. There are efforts at extending its scope to performance testing [17], but with its current semantics this requires adopting certain pragmatics of the use of language constructs [18].

D. ***Locality - the kind of entity to which P refers***

P may refer to:

(1) the *local* behaviour of a particular entity - a "protocol side" (e.g., "entity A receives message x and *then* sends message y");

(2) the *joint* (global) behaviour of a subset of implementation entities (e.g., "entity A sends message x, and *then* entity B sends message y").

Recall that the assumed model of a distributed system consists of a number of entities, each of which is non-distributed (local, and sequential) at a given abstraction level. The joint (compound) behaviour of multiple system entities strongly depends on the properties of communication channels. Although in principle it can also be transcribed as a sequential process (implied by a global reachability graph), for testing purposes this is normally impractical due to the phenomenon of *state space explosion*.

E. ***Observability, or an IUT - TS (Test System) relation***

The elementary events of behaviour that are expressed by *P* will have to be matched against the real events visible through a context. These real events may be:

(1) *communicated* to a test system (unrestricted, or perfect, observability); this is possible if a TS is co-located with *I*, and/or *I* is suitably *instrumented* ;

(2) *inferred* by a test system (restricted observability); a TS observes the *effects* of events executed by IUT (e.g., the messages present in a physical communication channel constituting a part of the context).

This aspect closely corresponds to the division into HIDS (1) and NIDS (2). In a HIDS, the causal dependencies between the events of the observed behaviour are naturally preserved, while the inferences made by a NIDS may be inaccurate or incorrect - this is a specific research problem due to *observation infidelity*, e.g., resulting from the possible *message cross-over* [19]. An incorrect inference of a trace can result in false negatives and false positives being issued, in addition to, and independently from, such verdicts issued due to the inadequacy of criteria used for detecting an intrusion-related behaviour.

F. *P-P$_D$ relation*

The two specifications may be related to each other as follows:

(1) *same level* : P and P_D are of the same "kind" - they both define behaviour (on its syntactic and semantic level), and they can be directly related to each other; as a special case, we may have $P = P_D$;

(2) *meta-level* : P is placed at a different - "higher" - level than P_D : it includes additional information, expectations and constraints pertaining not to the behaviour itself, but rather to *meta* -behaviour (the "behaviour of behaviours"), or the *pragmatics* of system operation.

Define an *(execution) policy* as a predicate W on sets of traces [16]. A set of traces $\Sigma \subset E^*$ *follows* a policy $\hat{W}$ if $\hat{W}(\Sigma)$ is true. Given a system *S*, if every set of traces Σ_S resulting from the execution(s) of this system follows a policy $\hat{W}$, then we say that *S satisfies* this policy. A policy is a *property* if policy satisfaction can be decided by evaluating a single predicate W separately for each individual trace. For a policy to be a property, such predicate (defined on traces, not sets of traces) must exist: $\exists W : E^* \to \{true, false\} : \forall \Sigma (\hat{W}(\Sigma) = true \Leftrightarrow \forall \sigma \in \Sigma : W(\sigma) = true)$.

In protocol engineering, as applied to practical telecommunications, a design specification typically refers to a *single instance* of infinitely *recurring* behaviour. This instance corresponds to a single realization of a system's *mission*: the handling of a single call, session, transaction, flow, etc. A design specification P_D defines, directly or indirectly, *all* (and *only*) the valid traces for *any single* instance. It is also normal for P_D to be nondeterministic; e.g., a connection may be set up or rejected due to the momentary lack of internal resources. It is tempting to assign modal qualifiers to those different traces: accepting a call is "good", while rejecting it is "bad". However, for a non-deterministic protocol specification, all the allowed traces are equally "correct".

Let us now return to the level of P vs. a P_D. In case (1), P is used to capture the essence of an attack-related behaviour as a *property* of each single sub-execution. The case of $P = P_D$ is quite common in research on passive testing [20, 4], where any trace not accepted by a design specification indicates a failure. In general, P may define as attack-symptomatic (and thus unadmissible) selected traces that are correct w.r.t. a P_D. In case (2), P defines a *policy*. This policy may state, e.g., that "too many" half-open connections are invalid, even if each one is formally correct.

Yet another, alternative explanation would be to treat case (1) as a *verification* problem, and (2) as a *validation* problem. An intrusion-related behaviour may be correct (successful verification against P of the first kind), but invalid (failed

validation against the expectations expressed by a *P* of the second kind). The division is not absolute and "stationary", as it is quite possible that experiences from intrusion detection activities could lead to reconsidering the contents of a P_D - over time, behaviours that were considered harmful may be marked in a P_D as *incorrect*.

G. *Dynamics*

P may be:

(1) *static* - unchanged throughout the operation of a TS;
(2) *evolving* - autonomously modified by a TS during its operation.

A typical example of an evolving reference specification can be found in the self-learning anomaly detection approach. Superficially, the behaviour of any program can be perceived as "evolving" in time, according to input data. However, for a given program its behaviour does *not* change, unless a program itself "mutates".

4.2. Secondary Dimensions

The secondary, but important dimensions of the discourse space are revealed when a generic passive testing / ID system is decomposed into three conceptual layers:

- a *problem-oriented domain*, in which a reference specification *P* is defined with a specific aim, e.g., intrusion detection;
- a generic *abstract behaviour testing domain*, in which a test program TP for the evaluation of behaviour is programmed in some "high layer testing language" TL, such as TTCN-3 [2];
- a run-time *detection engine domain* DE.

H. *P vs. test program (TP) / test language (TL)*

(1) *direct* - *P* is expressed by a test program written in a test language (TL);
(2) *multi-tier* - *P* is expressed in some suitable domain-specific language DL and then transformed into a TL program.

In the direct approach, the specification of reference behaviour is embedded in a test program written "manually", e.g., in TTCN-3. In the multi-tier approach, it is prepared in another formalism, e.g., as a finite state automaton (FSM, EFSM). This form is then translated automatically into the TTCN-3 program, using pre-defined transformation patterns. Note that the general need for a multi-tier approach has been acknowledged by the creators of TTCN-3: a very suitable alternative notation for intrusion detection is the MSC language, and its slightly redesigned form is used as an alternative, semantically equivalent, *Graphical Presentation Format* (GFT) of TTCN.

I. *Test program (TP) execution vs. detection engine (DE)* :

(1) *uniform* - a test program TP is executed according to the standard understanding of the execution mechanisms of programs;
(2) *hybrid* - a detection engine plays more than a technical role of a "processor"; a test program written in a TL may be transformed into a different form / paradigm (usually for the sake of run-time performance), or some functionality of a test program may be delegated to a detection engine, and thus must be expressed using the mechanisms of this engine.

This division may seem obscure, as it is based on the "standard understanding" of the program execution mechanism. However, in practice it is a distinguishable conceptual and practical problem. For example, [21] reports on a hybrid approach in which parts of the program are delegated to field-programmable hardware.

5 Discussion

The structure of the proposed discourse space is summarized in Table 1, along with the explanation of the place of the experiment reported in [2], and its relation to the general paradigms of intrusion detection.

One of the consequences of taking the passive testing perspective is that currently there is only a placeholder for the "active/passive" dimension in our taxonomy. According to [9], a passive IDS merely generates an alarm (in testing - a verdict), and an active IDS also reacts by taking a corrective or proactive actions. Apparently, "active" refers to what happens *after* detection, and not to any characteristics of the detection process itself (a reaction is an intrinsic feature of intrusion *prevention* systems - IPS). Any measurement is always a part of a loop of activities that aim at achieving or preserving agreed properties [4]. The dynamics of this loop vary, from an open loop (e.g., for documentation purposes), through a long-term development (re-design) strategy, scheduled corrective actions after failed tests, up to on-line control (reactive monitoring, measurement-based admission control, *intrusion detection*, enforcing the safety policy, and *real-time intrusion prevention*). However, whether the testing process itself is active or passive, appears to be a "free variable". In principle, a "truly active" intrusion detection method is also conceivable. A test system might actively stimulate a SUT so that an intrusion-related behaviour, *if present*, becomes "more symptomatic" and amenable to comparison with a suitable reference. We thus reserve a place for this dimension in our discourse space.

Table 1. The structure of the discourse space for intrusion detection. Classified are: misuse detection (M), anomaly detection (A), specification-based approach (S), the experiment reported in [2] (E), and generic passive testing (GP)

		M	A	S	E	GP
A: Semantics of reference	1: abnormal behaviour	■			■	
	2: normal behaviour		■	■		■
B: Aim of testing	1: to falsify		■	■		■
	2: to confirm (verify)	■			■	
C: Class of behaviour	1: qualitative (logical)	■		■		■
	2: quantitative	■	■		■	
D: Locality	1: local behaviour	■	■	■		■
	2: joint behaviour	■	■	■	■	
E: Observability	1: communicated	■	■	■		■
	2: inferred	■	■	■	■	■
F: Relation to design spec	1: same level			■		■
	2: meta-level	■	■		■	
G: Dynamics of reference	1: static	■		■	■	■
	2: dynamic		■			
H: Reference vs test program	1: direct approach	■	■		■	■
	2: multi-tier approach		■	■		
I: Execution of test program	1: uniform				■	■
	2: hybrid					
J: (Passive / active ?)						

6 Conclusions

Attempts at defining a taxonomy of ID concepts and techniques have been numerous [9, 22]. Similar publications come from the research community concerned with safety-critical industrial control systems [23]. Our discourse space is specific in that it treats intrusion detection as a *passive testing problem* and tries to capture the dimensions of *this* problem. Apart from vague remarks [14], we have not come across any work that would adopt a similar perspective. On the contrary, ID and testing have been presented as fundamentally different, or alternative [10] activities.

The developed taxonomy is not only *descriptive* (allowing to better understand the existing IDS approaches and techniques), but it also seems to have a *generative* potential [22] - many combinations of its elements are apparently valid and define new classes of intrusion detection systems that could be worth a closer inspection.

Acknowledgment. This work was supported by the Polish Government Research Grant N517 008 31/1429.

References

1. ITU-T Rec. Z.500, *Framework on formal methods in conformance testing,* ITU-T (1997).
2. Brzezinski K. M.: *Intrusion Detection as Passive Testing: Linguistic Support with TTCN-3,* Proc. DIMVA'2007, Lucerne, LNCS (2007).
3. ETSI ES 201 873. *Methods of Testing and Specification; The Testing and Test Control Notation version 3,* Parts 1-6.
4. Brzezinski K. M.: *Towards Practical Passive Testing,* Proc. PDCN'05, Innsbruck (2005).
5. Labib K., Vemuri V. R.: *Detecting And Visualizing Denial-of-Service and Network Probe Attacks Using Principal Component Analysis,* Proc. SAR'04, La Londe (2004).
6. Utting M., Pretschner A., Legeard B.: *A Taxonomy of Model-Based Testing,* Working Paper 04/2006, Dept. of Computer Science, Univ. of Waikato, Hamilton, New Zealand.
7. Hofmann R. et. al.: *Distributed Performance Monitoring: Methods, Tools, and Applications,* IEEE Trans. on Parallel and Distributed Systems, vol.5, no.6, June 1994.
8. Heerink L., Brinksma E.: *Validation in Context,* Proc. PSTV'95, Warsaw (1995).
9. Debar H., Dacier M., Wespi A.: *Towards a Taxonomy of Intrusion-Detection Systems,* Computer Networks. Int. J. Comp. and Telecomm. Networking 31 (9) Apr. 1999.
10. Ko C., Ruschitzka M., Levitt K.: *Execution Monitoring of Security-critical Programs in Distributed Systems: A Specification-based Approach,* Proc.IEEE Symposium on Security and Privacy (1997)
11. Uppuluri P., Sekar R.: *Experiences with Specification-based Intrusion Detection,* Proc. RAID'2001, 2001
12. Sekar R. et. al.: *Specification-based Anomaly Detection: a New Approach for Detecting Network Intrusions,* Proc. ACM CCS'02, 2002.
13. Eckmann S. T., Vigna G., Kemmerer R. A.: *STATL: An Attack Language for State-based Intrusion Detection,* JCS'02 (2002).
14. Bhargavan K., Gunter C.: *Requirements for a Practical Network Event Recognition Language,* Electronic Notes in Theoretical Computer Science 70 (4) (2002).
15. Netravali A. N., Sabnani K. K., Viswanathan R.: *Correct Passive Testing Algorithms and Complete Fault Coverage,* Proc. FORTE'2003, Berlin (2003).
16. Schneider F. B.: *Enforceable Security Policies,* ACM Trans. Information and System Security 3 (1), Feb.2000.
17. Dai Z.: *TimedTTCN-3, a Real-time Extension for TTCN-3,* Proc. TestCom'02 (2002).
18. Ziegler G., Rethy G.: *Performance Testing with TTCN-3,* TTCN-3 User Conf. (2006).
19. Bhargavan K., Chandra S., McCann P .J., Gunter C. A.: *What Packets May Come: Automata for Network Monitoring.* Proc. Symp. Principles of Prog. Lang. (2001).
20. Lee D., Netravali A. N., Sabnani K. K., Sugla B., John A.: *Passive testing and applications to network management,* Proc. ICNP'97 (1997).
21. Deri L.: *Passively Monitoring Networks at Gigabit Speeds Using Commodity Hardware and Open Source Software,* Proc. PAM'03 (2003).
22. Axelsson S.: *Intrusion Detection Systems: A Survey and Taxonomy,* Dept. of Computer Engineering, Chalmers Univ. of Technology, TR:99-15 (2000).
23. Papadopoulos Y., McDermid J.: *Automated Safety Monitoring: A Review and Classification of Methods,* Int. J. of Condition Monitoring and Diagnostic Engineering Management 4 (4), 14/32, Oct.2001.

Proceedings of the International Multiconference on
Computer Science and Information Technology, pp. 681 – 690
ISSN 1896-7094
© 2007 PIPS

Picture Passwords Superiority and Picture Passwords Dictionary Attacks

Krzysztof Gołofit

Warsaw University of Technology, Faculty of Electronics and Information Technology,
Nowowiejska 15/19, 00-665 Warsaw, Poland
University of Warsaw, Faculty of Psychology, Stawki 5/7, 00-183 Warsaw, Poland
K.Golofit@elka.pw.edu.pl

Abstract. This paper explores authentication techniques based on pictures as a possible solution to the most important problems concerning traditional passwords. The aim of this work is to bring together the technical (cryptological) and non-technical (psychological) awareness into the research on passwords. Security issues of any authentication mechanism (relying on knowledge) should not be considered without analysis of the human factor − since the users' human nature was identified as a source of major weaknesses of conventional authentication. Several issues of security and possibility of practical application were discussed in the paper. In the first place the statistically significant superiority of picture passwords over alphanumerical ones was presented. Then, it was shown that the crucial weaknesses of the picture passwords follow less from pictures locations and more from their meaning and ability to be recognized. Moreover, the techniques leading to resistance to 'key logging' and 'mouse tracking' were discussed. Finally, methods guaranteeing that users choose dissimilar, personalized and cryptographically strong graphical passwords were proposed.

Key words: graphical authentication, picture passwords (p.p.), p.p. superiority, p.p. cryptanalysis, p.p. dictionary attacks, strong passwords, human factor

1. Introduction

According to recent reports, many researches and statistics from real systems, there are many vulnerabilities and threats typical of alphanumerical passwords. As usual, the users are 'the weakest link in the security chain'. One of the major problems is the difficulty of remembering passwords, the other, ignoring security requirements. Users tend to create either too short passwords or passwords that though long enough are easy to guess. There is an informal rule stating that passwords easy to remember, are mostly also easy to break. According to Schneier [1], passwords' length distribution based on 34,000 users shows that 65% of passwords have only up to 8 characters and

almost 95% up to 10 characters. Other research [2] shows that only 17% of the inquired IT professionals use complex passwords (including letters, numbers and symbols) and 72% stated that they almost never or never change their access codes. 52% of professional users tend to share their passwords and 65% of them have only one or two passwords to access the majority of services. A study of information contained within the passwords [3] shows that 66% of users' passwords are designed making use of personal characteristics thereof, where 32% contained names of people, places or things. The common constructions involve full information in the passwords – 75%, partial – 13.5%, or combined – 7.5%. Almost all respondents reuse passwords – on average 4.45 passwords are used in 8.18 systems. On the other hand, strong passwords imposed on users bring no solution as well – people cannot and/or would not remember strong passwords and write them down instead. According to [4] we can say: there will be either about 80% remembered but weak passwords (created by users) or 80% strong passwords (generated by the system) but written down.

Many alternative authentication solutions have been invented and developed to ensure the proper security level – in order to avoid weaknesses of traditional methods. One group of techniques (involving a physical factor in the authentication process – 'something you have') focuses on utilizing all kinds of tokens, one-time passwords, magnetic stripes and proximity cards, iButtons, cryptographic cards, etc. The other kind of research makes use of methods (called 'something you are') based on biometric information like fingerprints, voiceprints, the patterns of blood vessels on the eye retina, the topography of the eye iris, the geometry of the hand, facial patterns, DNA codes or even thoughts (cerebral waves). However, all of the aforementioned solutions have two significant disadvantages. First, they may become unacceptably expensive when a large number of users are involved. Second, access to the system is strongly dependent on the suitable interfaces – which makes such methods comparatively less universal (in the context of mobility) and in some cases, impossible to use. Additionally, biometrics is extremely vulnerable to a replay attack – the personal information is hard to hide (face, fingerprints) and cannot be changed.

In the past few years we have been observing a growing interest in graphical authentication techniques – those of knowledge-based methods, which include graphical aspect(s) in the authentication process. There are a few distinct grounds, which aroused the interest in graphical techniques. There are methods particularly useful for mobile devices and/or systems that have no keyboards [5], methods resistant to *shoulder-surfing* attacks (enabling to log in 'in the crowd') [6], there are also advantages coming from resistance to malicious software (malware). Notwithstanding, the leading inclination is still to construct a system, which will prevent from choosing trivial passwords and which will allow to remember passwords with the cryptographically proper length.

2. Picture Passwords Superiority

There is no psychological research which can directly justify the superiority of graphical authentication – although there are publications that points at the *picture superiority effect* as a proof of picture passwords superiority. This phenomenon (introduced by Nelson et al. in 1976, [7]) states that pictures are much more likely to be remembered than words. However, in the experiments both pictures and words

characterize the same concepts, e.g. word 'cat' versus 'a drawing of the cat' – there is no knowledge about various sequences of pictures versus various sequences of keyboard characters. At the same time, there exists Paivio's *dual-coding theory* [8], saying that both visual and verbal information is processed differently and along distinct channels. Concrete concepts presented as pictures are encoded into both systems; however, abstract concepts are recorded only verbally. On the basis that separate information representations are processed in each channel there exists a part of Pavio's theory called *coding redundancy hypothesis*. It states that "memory increases directly with the number of alternative memory codes available for an item". It seams to be the most important argument in favour of picture passwords when the users both name and picture their passwords – but we also do not know how single characters or their sequences will be coded in comparison with picture. Each symbol can arbitrarily represent something else than itself and unique composition of the symbols can trigger off distinct and various associations for each person.

Moreover, there are series of interesting researches, which can help to understand how to create efficient methods based on graphical materials. For example, study of what kind of information will be remembered over relatively long periods of time [9]; how people remember nonsensical material and what are the conditions to improve memory for pictures [10]; what is the role of prior knowledge in recognition [11]; what is the influence on memory when words and pictures are or are not recognized [12]; how we recognize pictures with or without additional details [13]; what is the influence of symmetry in the remembered material [14] and how can we manipulate with the colour (vs. b&w), high versus low information, semantic versus sensory processing [15]. These (among others omitted) can be only a clue to understand which factors should be taken into consideration when constructing the authentication mechanism, but none of them answers the questions about superiority or strong and weak points of picture passwords.

In order to confirm *picture password superiority* three experiments were conducted (two preliminary). The first one was constructed to be possibly close to the real authentication systems – subjects created the passwords themselves. Unfortunately, instead of the thesis, the tendency in password selection was perfectly confirmed. In spite of instruction (to take particular care over creating non-trivial passwords) participants chose passwords like '*duckonduck*' or '*0a1b2c3d4*' – in the case of alphanumerical passwords; likewise ch ose row, column or diagonal of the graphical keypad – in the case of picture passwords (the more sophisticated ideas of graphical passwords were based e.g. on moves of the chess knight). Once again the adage that 'system fails when users pick the passwords on their own' founds confirmation. The second of the pilot study examined the proper length of the sequences (of pictures and characters) to be remembered – the sequence was established as 25 elements.

Hypothesis. Because there was no prospect of subjects choosing passwords for themselves, the hypothesis was established as follows "Random sequences of the pictures will be recalled from the long term memory better than random sequences of the alphanumerical (keyboard) characters".

Method.

According to the hypothesis and the aforementioned preliminary experiments participants were asked to memorize the sequence consisting of 25 elements – pictures in the one case, characters in the other.

Subjects. There were 41 participants – male students in the third year at the Warsaw University of Technology. 21 of them dealt with pictures and 20 with characters.
Apparatus. There were prepared four randomly generated sequences of 25 elements. Characters were generated from the 90 alphanumerical (keyboard) signs. Pictures were chosen from the set of 90 cliparts. The set consists of colour pictures, rather simple than complex and presented recognizable objects.
Procedure. 'A coin flip' decided which of the tests was given to the participant (in practice, tests were mixed before the experiment). In the first part of the experiment subjects had 300 seconds to memorize assigned sequences. The second part came after 60 minutes (which indicates long term memory) – participants recalled the sequences without time restriction. After that they were asked about strategies used for memorizing the material.

Results. The responding variable was amount of the correctly recalled elements from the beginning of the sequence (up to the first mistake). There were only several participants necessary to affirm *picture password superiority* hypothesis with the $p<.05$. Significance level (with the normal distribution assumption) based on all participants was $t(39)=3,16$, $p<.0016$ or based on Wilcoxon's rank test: $p<.0024$.

3. Origins of Picture Passwords Cryptanalysis

As long as the discipline is still new and there is no wide range of practical implementations, we have no direct observations on the graphical passwords and we don't know much about possible attacks on the graphical authentication systems. Four hypotheses were constructed in order to disclose vulnerabilities of picture passwords. One the one hand, locations of the elements (independent and relative) were taken into consideration. On the other hand, mutual meanings of the pictures as well as influence of recognizing possibility were investigated.

Method.
With regard to the following four hypotheses there was prepared an interface consisting of 60 pictures. Twenty of them (10 pairs) were chosen to have obvious connection (like '*Christmas tree*' and '*Santa Claus*' – Tab.2), twenty were supposed to be difficult to identify or recognize (Tab.3) and the last one-third of the elements should be neither related to another nor hard to be named.
Subjects. In the main research there were 300 students at the Warsaw University of Technology form third up to tenth semester of study – groups: I-VI of Tab.1.

Table 1. Characteristics of the experiments' participants.

Group:	**I**	**II**	**III**	**IV**	**V**	**VI**	**VII**	**VIII**	**General:**
Number of Subjects:	69	35	46	32	60	58	61	24	**385**
Females / Males:	3/66	1/34	0/46	3/29	3/57	4/54	3/58	10/14	**27/358**
Semester of study:	4	5	3	6 -10	4	6	1	(18-35)yo	**1-10**

Apparatus. 60 colour pictures (*CorelDraw Cliparts*) were used to construct the interface – matrix 10x6 randomly placed elements (illustrated on Fig.1).

Procedure . The participants from groups I – VI (Tab.1) were asked to choose and remember passwords consisting of 7 elements based on graphical keypad (Fig.1). The participants were also informed that pictures should be remembered in the right order and only regarding to the meaning of the elements (not to the location), because in the second part of experiment the set of pictures will be mixed. There was no time restriction. There was no second part (based on recall).

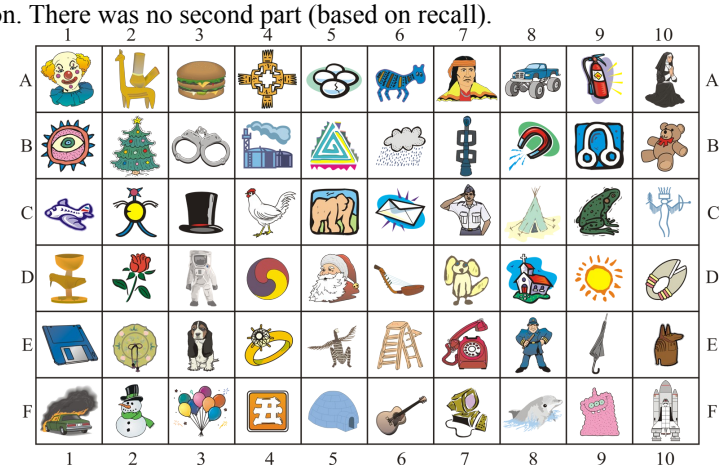


Fig. 1. Picture passwords keypad – matrix10x6 randomly placed *CorelDraw Cliparts*, [16].

Hypothesis H1. *The first elements of passwords will be placed closer to the top left corner of the keypad then the randomly chosen.* Justification: as far as the pictures are new for the participants there may arise a tendency to choose the first element where we usual start reading (papers, books, etc.) – the top left corner.

Results. There were 300 passwords and therefore 300 first elements. The distribution of the distance of the first element according to an element chosen by random is illustrated in Fig.2 (left). In the first quarter of the distribution of distance we can observe 2.3x more choices in respect to the random choices – 31% of selections instead of 13.(3)% expected by random. Based on Wilcoxon's test the level of significance is $p<.013$ – the tendency can be observed, but it seems to be too weak to taking into consideration by planning a dictionary attack.

Hypothesis H2. *Two neighbour elements in the sequence of picture password will be placed closer than two elements chosen by random*. Justification: it is suspected that searching for the elements and paring those in order to better memorizing can result in closer location between two neighbour password elements.

Results. There were 1800 pairs (seven elements in each of 300 passwords) intended and unintended by the participants. The histogram of distance distribution of two neighbour elements in the passwords according to the two chosen by random is illustrated on Fig.2 (right). In the first quarter there is even smaller tendency than H1 (33.9% instead of 25.8%) – with the Wilcoxon's test significance level is $p< .33$.

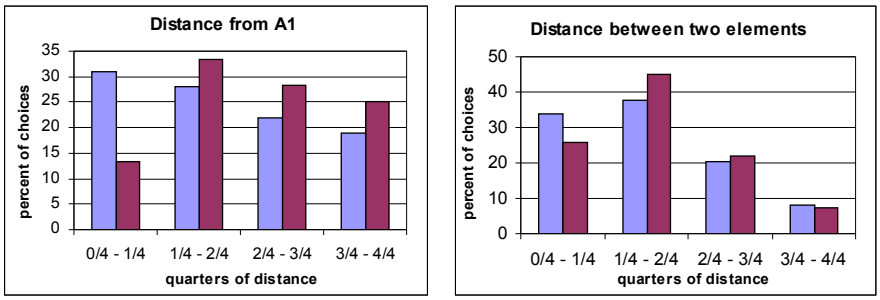


Fig. 2. Distribution histograms: the distance from the fist element – A1– *clown* (on the left); distance between two neighbour elements (on the right).

Hypothesis H3. *Elements that are more difficult to recognize (and to be called) will be chosen rarely than other elements***.** Justification: there is an assumption that users will not choose and memorize elements that cannot be described or are difficult to. The twenty 'hard to describe' elements were listed in Tab.2 (compare Fig.1). The difficulty of an element was measured by the time of verbal response to the element and confirmed by research on group VIII (Tab.1).

Table 2. List of the difficult to identify or recognize elements (compare Fig.1).

	A2		A4		A6				
B1				B5		B7		B9	
	C2			C5					C10
D1			D4		D6	D7			D10
	E2			E5					E10
			F4					F9	

Results. Statistics for 60 pictures based on 2100 choices (on average 35 choices per one element) are illustrated in Fig.3. The 20 elements of hypothesis H3 (Tab.2) marked with brown/dark colour were chosen almost 4x more seldom than the others (excluding H4 intended pairs marked as white). The evidence for hypothesis H3 is quite strong – null hypothesis was rejected with: $p<10^{-9}$ ($p<10^{-12}$ with all elements).

Hypothesis H4. *The pairs of pictures that have meaning connection will be chosen more frequently than the elements chosen by random*. Justification: When there is obvious relationship between two pictures, memorizing will be easier (through stronger associations) and what follows those elements will be chosen willingly. The 10 pairs prepared to have obvious connection are listed in Table 3.

Because there is no such thing as obviousness in the meaning, 61 first year students (group VII, Tab.1) were asked to select 10 most connected pairs from the keypad and order them from the most meaningful to the least (assigned weights from 10 to 1). Both frequency of elements choices and weights statistics confirmed all chosen pairs.

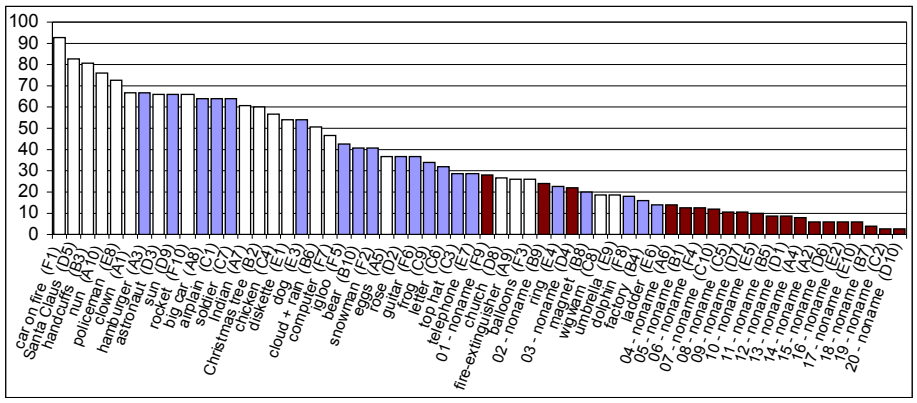


Fig. 3. Numbers of choices: H3 difficult to recognize elements (brown/dark), H4 pairs (white).

Table 3. List of the intended pairs in the keypad.

diskette – computer	E1–F7	cloud+rain – umbrella	B6–E9
astronaut – rocket	D3–F10	handcuffs – policeman	B3–E8
Indian – wigwam	A7–C8	eggs – chicken	A5–C4
Christmas tree – Santa Claus	B2–D5	clown – balloons	A1–F3
nun – church	A10–D8	fire-extinguisher – car on fire	A9–F1

Results. Among 1830 possible pairs (including pairs consisting of the same element), between 1800 pairs being analyzed, there occur 714 different pairs. The list of the top 50 pairs presents Figure 4 (pairs listed in Table 3 were marked with the brown/dark colour). One pair (*clown – balloons* A1–F3) was chosen only 4 times – what put this pair between 93 rd and 131 st place (the other nine pairs are situated in the top 50 list). The 10 H4 pairs collected 143 choices, what by the expected value 0.984 of random choices per pair, makes the evidence for hypothesis H4 very strong ($p<10^{-16}$). However, there appeared a lot of non-intended pairs like A8–F1 (*big car – car on fire*), C7–E8 (*soldier – policeman*) and F2–F5 (*snowman–igloo*), which have as great popularity as the intended pairs. It shows that we cannot perfectly foresee the efficacy of the picture authentication mechanism without an expensive research on users.

4. Conclusions

One of the main requirements when exploring human tendencies is to test a large number of participants. In the described research the verification of the memorized passwords was given up. On the one hand we have no screening of malicious behaviour of the participants, but on the other hand, it made putting more subjects to the test feasible. It was ensured that the participants were convinced that there would be a second part of the experiment. Moreover, in respect to knowledge about tendentiousness in picture password selections (rows, columns, etc.) the subjects were informed that the pictures will be randomly mixed. The outcome was very good

because there were only a few weak sequences (making a graphical arrangement) – and being only parts of the passwords.

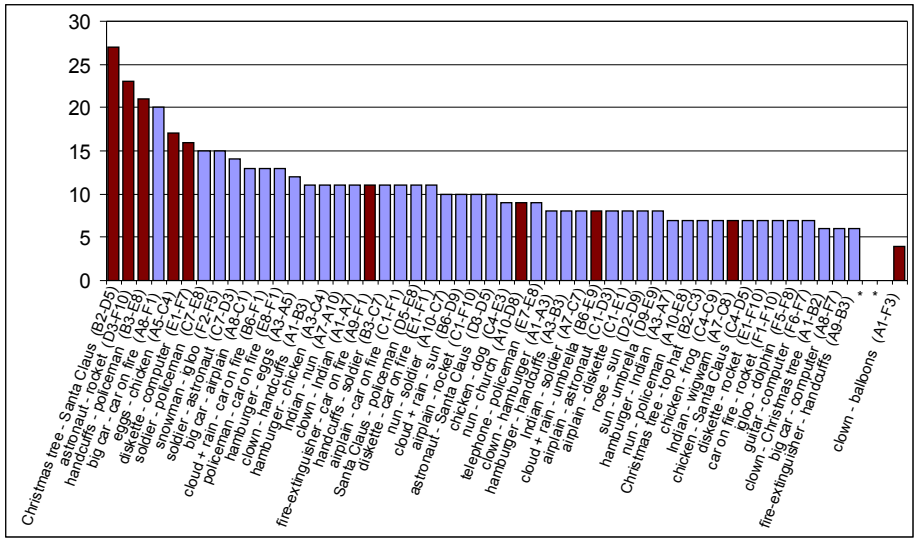


Fig. 4. Numbers of pair's choices – the top 50 list (2.73% of all pairs & 29.8% of all choices).

However H1 and H2 hypotheses did not reveal very significant vulnerabilities in respect to either independent location of the first elements or pictures with reference to other pictures. H3 and H4 show quite important correlations in respect to recognition possibilities and meaning of the elements. Figure 3 shows that the frequency of choosing one element with reference to another can be in the ratio of 20 to 1 (or even more). Such differences in popularity can dramatically decrease the passwords entropy. In order to choose a proper set of pictures there will always be a need for reliable investigation on human material concerning the contents of elements – techniques leading to minimize differences in choices without expensive research on real interfaces, large group of people and long time observations are thus crucial.

5. Discussion of *Pros* and *Cons*

There are much more issues (apart from hypotheses) that should be taken into consideration before practical use of picture passwords (and related methods based on pictures). This section is a compilation of conclusions with so far presented as well as yet not mentioned issues – which correspond with problems of traditional passwords.

Advantages of Picture Passwords

1. There are possibilities of significant enlargement of the picture passwords space. In the traditional passwords, the base does not exceed one hundred (b^n – password space, b – base, n – password length). Picture passwords offer the password base up to ten times larger – technically, the only limitation is the resolution of the screen.

2. There is technical possibility to prevent users from choosing trivial passwords. Through the permutation of the set (only once – after the password creation) the users are forced to remember their passwords regarding only the meaning of elements and not the topological placement of elements (like row or column).

3. It's possible to personalize the passwords, which can prevent from (common) dictionary attacks. After removing irregular elements and pairs from the set (according to H3 and H4 hypotheses respectively), chosen passwords will depend only on individual thoughts, associations with one's past and personal feelings – in contrast to the common language dictionaries and attacks based on universal strings.

4. There is a possibility to make the authentication process resistant to 'key logging' and 'mouse tracking'. The first resistance is obvious (due to not using the keyboard). The second one is achieved (again) due to the permutation of the set. But in this case, the permutation table should be individual for each login name (and/or workstation). As a matter of fact the sequence of the screen coordinates (users' clicks) will be meaningless without the view of the screen.

5. Sharing passwords with other users as well as noting them down or giving them away as a result of a social engineering attack are much more difficult. While traditional passwords can be easily written down or spelled out, giving a picture password out is very troublesome. The effect can be especially intensive when the keypad consists of ambiguous symbols or symbols having similar names.

6. Passwords *incrementation effect* (i.e. recurrent and obvious changes made to the passwords – e.g. change from "2nd*SIS#07" to "3rd*SIS#08") can be neutralized. It is because there are no sequences or numbers when the keypad pictures were properly chosen (e.g. avoiding signs). Even when user changes only one element of the password − it is still hard to guess which one and which is the new one. Moreover, there is always a way of forcing users to change their passwords significantly by modifying (entirely or partially) the set of pictures.

7. Likewise, the rule "one system – one password" can be asserted by different sets of the pictures in those systems where unique password codes are required.

8. There is a possibility for undemanding and inexpensive implementation (in comparison to biometrics and cryptographic hardware).

9. There was revealed the *picture password superiority* over alphanumeric codes resulting (as presumed) from better memory for picture passwords, (e.g. in [17]).

10. There are known a few techniques preventing *shoulder-surfing attacks*, [6].

11. There are many prospective possibilities (manipulation of the pictures colours, meanings of the elements, personalizing passwords sets) of achieving even better memory abilities and/or constructing much stronger mnemonics.

Disadvantages of Picture Passwords

1. There is a threat based upon tendentiousness of choices resulting in decrease of passwords entropy. The most important effort is to choose the proper graphical material for an interface, but only statistical analysis based on large (enough) group of people can confirm the right choices of the pictures.

2. More time needed to enter the password. Although we can expect shortening of the passwords (compared with alphanumerical ones) we should not expect better time performance in entering picture passwords (even after the users have learnt the order and placement of elements).

3. Graphical interface is required − which means that the graphical authentication methods will be less universal (in terms of mobility) and troublesome in implementation (in comparison to traditional passwords).

4. Conventional picture password method is not resistant to *shoulder-surfing attacks* − there is a better chance to observe the passwords entered on the screen than the alphanumerical passwords typed with traditional keyboard.

5. There are many people who dislike: changes, giving up their habits, technical novelties, graphical interfaces (i.e. unix or linux users) and "compulsion to better life" (as in "we know what is better for you").

6. We should not forget about visually impaired and blind people (who are using common internet services as well), for whom something like graphical authentication interface will be impassable or hard to get through.

7. The discipline is still new and there may be many attacks, of which we cannot be aware at the present time.

References

1. Schneier B. *Real-World Passwords*. Crypto-Gram Newsletter, December 15, 2006.
2. Magalhaes S. T., Revett, K., Santos, H.D. *Generation of Authentication Strings From Graphic Keys*. IJCSNS, Vol. 6 No. 3 pp. 240-246, 2006.
3. Brown A. S., Bracken E., Zoccoli S. and Douglas K. *Generating and remembering passwords*. Applied Cognitive Psychology 18, 641-651, (2004).
4. Zviran, M., Haga, W. J., *User authentication by cognitive passwords: an empirical assessment*, 5 JCIT, 137-144, 1990.
5. Jansen W., Gavrila S., Korolev V., Ayers R., Swanstrom R. Picture Password: *A Visual Login Technique for Mobile Devices*. NIST, NISTIR 7030.
6. Zhi L., Qibin S., Yong L., Giusto D. D. *An Association-Based Graphical Password Design Resistant To Shoulder-Surfing Attack*. IEEE, ICME, 2005.
7. Nelson D. L., Reed U. S., Walling J. R. *Pictorial superiority effect*. Journal of Experimental Psychology: Human Learning & Memory, 2(5), 523-528, Sep. 1976.
8. Paivio A. *Mental representations: A dual-coding approach*. NY: Oxford Univ. Press, 1986.
9. Mandler J., Ritchey G. *Long-term memory for pictures*. Journal of Experimental Psychology: Human Learning and Memory, pages 386-396, 1977.
10. Bower G. H., Karlin M. B., Dueck A. *Comprehension and Memory For Pictures*. Memory and Cognition, 3:216-220, 1975.
11. Long D. L., Prat C. S. *Memory for Star Trek: The Role of Prior Knowledge Recognition Revisited*. J. of Experimental Psychology: Learning, M. & C., 28(6), pp.1073-1082, 2002.
12. Kroll J. F., Potter M. C. *Recognizing words, Pictures, and Concepts - A Comparison of Lexical, Object, and Reality Decisions*. J.of Verbal Learning & V.Behavior 23, 39-66, 1984.
13. Pezdek K., Maki R., Valencia-Laver D., Whetstone T., Stoeckert J., Dougherty T. *Picture Memory: Recognizing Added and Deleted Details*. Journal of experimental psychology. Learning, memory, and cognition, 14(3), pp.468-76, 1988.
14. Attneave F. *Symmetry, Information and Memory Patterns*. American Journal of Psychology 68, 209-222, 1955.
15. Childers T. L., Houston M. J. *Conditions for a Picture-Superiority Effect on Consumer Memory*. Journal of Consumer Research 11: 643-54, 1984.
16. CorelDRAW, *Clipart and Photos*. Graphical material used for research comes from CorelDRAW distribution (license for ISE PW). All rights reserved.
17. Gołofit K. *Metody uwierzytelniania graficznego jako alternatywa dla haseł tradycyjnych*, XI ENIGMA, Warsaw, 23-25 May 2007, pp. 389-405. [in Polish].

Proceedings of the International Multiconference on
Computer Science and Information Technology, pp. 691 – 700
ISSN 1896-7094
© 2007 PIPS

Design and Implementation of a Portable ID Management Framework for a Secure Virtual Machine Monitor

Manabu Hirano [1, 2] , Takeshi Okuda [2] , Eiji Kawai [2] , Suguru Yamaguchi [2]
[1] Department of Computer and Information Engineering,
Toyota National College of Technology , Aichi, 471-8525, Japan
hirano@toyota-ct.ac.jp
[2] Graduate School of Information Science,
Nara Institute of Science and Technology, Nara, 630-0192, Japan
{okuda,eiji-ka,suguru}@is.naist.jp

Abstract. A commonly used virtual machine monitor (VMM) allows multiple operating systems to share physical hardware resources as virtual resources in a safe manner. It provides a strong isolation mechanism between virtual machines (VMs). In this paper, we state the importance of ID management for a security-purpose VMM system to enforce security policy on an end-user environment. We present a design of a portable ID management framework for a security-purpose VMM. Our proposal employs a smart card (ID card) for user authentication. The proposed ID management framework can provide generic programming interfaces to existing VMM software. Our ID management framework realizes an authentication between a VMM and its users, an authorization for a VM boot operation, storage of cryptographic keys for VMM-layer's disk encryption/decryption, and access control for virtual/physical resources based on a user identity. In this paper, we show the prototype implementation of our ID management framework and its integration into the proven VMM software, QEMU.

Keywords: ID Management, Authentication, Virtual Machine Monitor, VMM, Hypervisor, Smart card, ID card

1 Introduction

A virtual machine monitor (VMM), also called a hypervisor, allows multiple operating systems to share physical hardware resources as virtual resources in a safe manner. An ideal VMM technology provides complete isolation of virtual machines (VMs) [1]. Especially, a layer of a trusted VMM can, without modifying a guest operating system, provide useful and strong security functions transparently for each VM, such as storage encryption, traffic confidentiality using a virtual private network (VPN) and other access control mechanisms for physical resources. We call this kind of VMM system "a secure VMM". We assume that a secure VMM system for an end-

user environment is used as a foundation of security policy enforcement in an organization. In recent research, some security-purpose VMM systems have been developed. However, these systems do not consider ID management on the VMM layer. In this paper, we show an ID management framework for these security-purpose VMM systems.

At first, in section 2, we summarize related work on security-purpose VMM systems such as sHype, NetTop, Terra and their security mechanism. In section 3, we show a model for a typical secure VMM system and a problem we have found with it. Section 4 shows the design of our portable ID management framework for a secure VMM system. In section 5, we present our prototype implementation using a smart card and middleware systems. We describe our future work in section 6. We conclude with a summary in section 7.

2 Related Work on a Secure VMM system

A VMM is a technology to encapsulate an operating system, a technology which was originally designed and developed for mainframe computers such as IBM VM/370 [2]. It was intended to support legacy software in a new hardware environment. In recent research, some VMM systems designed for security-purposes have been developed. In this section, we summarize the related work on secure VMM systems.

sHype. sHype (Secure Hypervisor) [3][4] is developed by IBM Research. The subject of the sHype project is to provide a secure foundation for server platforms. sHype provides an access control and an isolation mechanism of virtual resources in hypervisor software. sHype is intend ed to provide a mandatory access control (MAC) for inter-VM communication in *VM coalitions* . sHype is currently implemented on the Xen hypervisor [5] , rHype (Research Hypervisor) [6], etc . sHype implementation supports some major security policy mechanism s such as the C hinese W all policy and the s imple Type Enforcement policy. sHype mainly consists of: an access control module for virtual resources and inter-VM communication , hypervisor mediation hooks , callback functions and a policy management VM. sHype also supports a TPM (Trusted Platform Module)-based attestation mechanism [7, 8] .

NetTop. NetTop [9] is designed by NSA. NetTop is constructed with commercial off-the-shelf technology, the VMware product [10] , which started in DARPA-sponsored research at Stanford University, and the SELinux as a host operating system. NetTop provide s security mechanism s based on isolation of VMs, reliability of a host OS and VMM software. NetTop project also states the importance of trusted BIOS because all host platform security is depend ent on an initial boot-up process. NetTop mainly provides a transparent *VPN VM* and a *Filtering VM* that work with an end-user OS like Microsoft Windows . NetTop is intended to be deployed in a governmental environment . NetTop also provides a secure data transfer mechanism, called the “Regrade” server protocol , between two VMs with different security level s . The r egrade server achieves data transfer based on a token-based user identity and a

regrade polic y. It also check s and sanitiz es malicious content , and record s audit log s. NetTop also provides transparent storage encrypt ion and coalition support.

Terra. Terra architecture provides a trusted virtual machine monitor (*TVMM*) that isolates and protects independent virtual machines [11] . Ter r a architecture is designed to build a trusted computing platform using VM technology . Terra architecture supports a remote attestation mechanism to establish a trusted path. A t rusted path is achieved by attestation certificate chain s for BIOS, a boot loader and a VM image. Terra also supports an encrypted disk or an integrity-checked disk in the VMM layer . A prototype implementation of Terra architecture employs The *VMware ESX server* product and some python scripts.

2.1 Security Mechanisms

Most security frameworks using VMM technology are designed carefully to maintain performance and security. Therefore, it is preferable that security codes in a VMM are as small and simple as possible. In addition, security frameworks described above are dependent on the security and reliability of the VMM layer, the underlying BIOS, and a boot-loader. So security frameworks based on VMM technology should guarantee tamper-proof BIOS firmware, a boot-loader and secure VMM codes. For examples, sHype supports TPM-based attestation to check the integrity of hypervisor software and VM images. NetTop and Terra also state the importance of a trusted BIOS and a reliable execution mechanism for BIOS, a boot-loader, VMM software and VM images. Terra states the importance of a trusted path by a remote attestation mechanism. A secure VMM should support a secure and reliable bootstrap mechanism [12].

Security frameworks using VMM technology are based on the isolation mechanism of that technology. Each VM on same VMM can not influence the other VMs. For this mechanism, we can separate the purpose of each isolated VM. For example, one VM can connect to the Internet, but another protected VM can connect to an intranet only. NetTop takes this approach to reduce the number of physical machines for different purposes and different security levels in a governmental environment.

NetTop and Terra support a transparent storage encryption mechanism. The advantage of this mechanism is that it does not require modifying the guest operating system. The secure VMM layer can provide storage encryption and decryption automatically. Even a guest operating system is attacked; an encryption key is not leaked – on the assumption that a secure VMM layer is strongly protected from untrusted guest operating systems and other malicious actions. NetTop also provides a transparent VPN function for guest operating systems using a *VPN gateway VM* . This approach achieves simple and practical security. It has the same effect as two physical separated machines, an end-user machine and an administrator-controlled VPN gateway server.

We summarize the security mechanism of a secure VMM system as follows: (1) simple design and minimum overhead for security processing with high performance and high reliability; (2) trusted BIOS, trusted bootstrap architecture and an attestation mechanism for core components (a boot-loader, VMM software and VM images). So

we need a software measurement mechanism such as TPM to check software integrity; (3) a strong isolation mechanism for each VM; (4) a mandatory access control (MAC) function based on an access control list (ACL) or a security policy (e.g. the Chinese Wall policy, the Type Enforcement policy). The MAC function basically consists of a policy decision point, a policy enforcement point (i.e. hooks for virtual/physical resources), and a policy management function to update and validate distributed policy files; (5) transparent security functions such as automatic storage encryption and traffic confidentiality by a VPN. These security functions will provide useful components for a secure VMM system.

3 Practical Model for a Typical Secure VMM System

Fig.1 depicts a model for a typical secure VMM system described in section 2.1. A key component of a secure VMM system is mandatory access control (MAC) based on a trusted VMM layer. A VMM layer controls the virtual/physical resources from each VM. In this paper, the word *resource* refers to devices such as a physical NIC which handles network packets, mass storage, a USB device like a thumb drive which holds user data, and other peripherals such as a printer, and so on. A *Policy Decision Point (PDP)* in Fig.1 processes an authorization decision based on VM requests for virtual/physical resources and VM identity (VM #1, #2 or #3). *Policy management* updates and validates a security policy which is distributed by a central policy server in an organization. A *hook point* or A *Policy Enforcement Point (PEP)* are used to enforce the security policy on VM requests. PEP interacts with PDP to retrieve an authorization result. *Transparent security functions* add useful and practical security functions to a secure VMM (storage encryption/decryption, VPN service, etc).

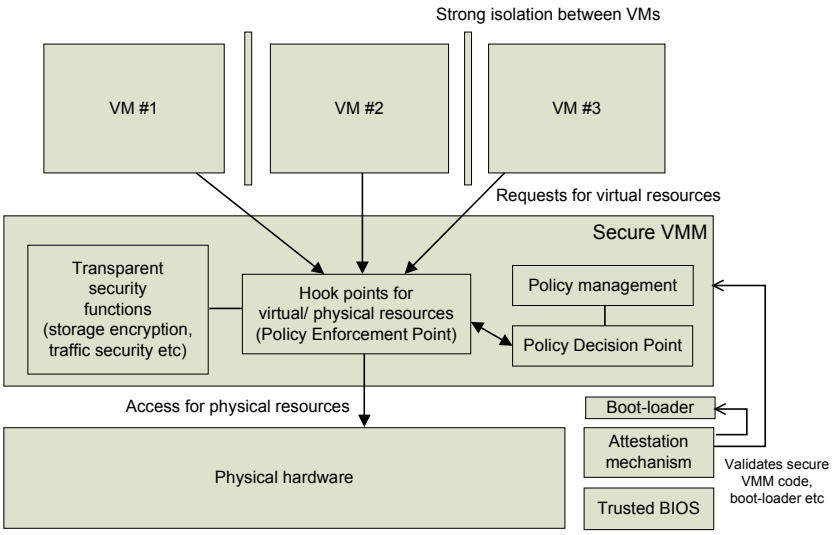


Fig. 1. A model for a typical secure VMM system: This model consists of *secure VMM software*, *trusted BIOS*, a *boot-loader* and an *attestation mechanism* to construct a whole trusted VMM system.

To check reliability of a secure VMM platform, an *attestation mechanism* validates boot-loader software, secure VMM execution codes, and so on. The bootstrap process should also be executed securely. Attestation and trusted bootstrap are key technologies for a secure VMM system.

A secure VMM (Fig.1) provides a MAC function. The foundation of a MAC function is dependent on there being only one administrator that can access a secure VMM system. Therefore, an end-user cannot bypass the administrator-controlled secure VMM layer. Thus, a secure VMM can provide a policy enforcement mechanism for an end-user environment.

3.1 Problem Statement

In existing secure VMM systems, a VM entity is identified by VMM software, and a secure VMM enforces a security policy based on an each VM identity (VM ID). However, to deploy a secure VMM for a practical operational environment, we need to manage a user-identity in a secure VMM. Fig.2 shows access control models for a secure VMM system. (A) in Fig.2 shows access control based on a VM ID only. (B) in Fig.2 shows access control based on both a VM ID and a user identity (USER ID). In this case, secure VMM software authenticates a user. Therefore, secure VMM software can manage a USER ID, and can apply the USER ID to the access control mechanism. In a conventional system ((A) in Fig.2), secure VMM software does not authenticate a user; a user authentication mechanism is normally dependent on a guest operating system (e.g. Windows Logon, Unix user/password). A conventional system cannot handle the mapping between a VM ID and a USER ID in a VMM layer.

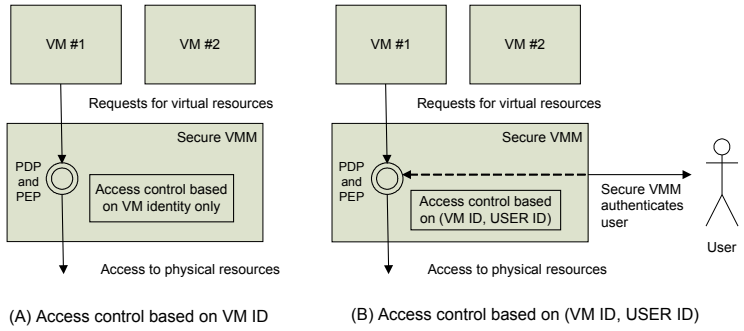


Fig. 2. Access control models for a secure VMM system: (A) is *VM ID*-based access control, (B) is access control based on both a *USER ID* and a *VM ID*.

From the perspective of a security administrator, ID management on multiple guest operating systems is not efficient, because the administrator have to maintain multiple user IDs on multiple guest operating systems. Our proposal assumes that a trusted VMM layer can provide many useful security functions transparently to administrators. Introducing user identification and authentication in a secure VMM system has many advantages. First, a secure VMM will be able to control many VMM functions based on a USER ID, such as a VM boot operation and disk encryption. Furthermore, access control for virtual/physical resources will be more effective using

an authenticated USER ID because a security administrator in most organizations prefers a USER ID-based authorization, instead of a VM ID-based authorization only. The USER ID-based authorization also enables a secure VMM to record audit logs based on a USER ID. An audit log is an essential component of security systems. An audit log also achieves non-repudiation of user actions on an administrator-controlled secure VMM system. In this paper, we describe a user authentication and an ID management framework using a smart card (ID card) for a secure VMM system.

4 Design of a Portable ID Management Framework for a Secure VMM system

4.1 User Authentication for End-point Security Enforcement

Fig.3 shows the basic design of the portable ID management framework for a secure VMM. This example employs a user identity to control a VM boot process. We assume that each employees of an organization has their ID card, and that the ID card is a multi-application smart card. Our proposed smart card stores a *Secure VMM application* which supports PKCS#11 compatible functions. So our proposed ID management framework can be used with many smart card products. A *secure VMM ID management API* in a secure VMM is a high level API using PKCS#11 middleware and underlying libraries such as a PC/SC (Personal Computer/Smart Card) library and a device driver of a smart card reader.

A secure VMM ID management API provides user authentication functions based on Public Key Infrastructure (PKI) for secure VMM systems. Our proposal also provides a standard ID/password authentication API without a smart card. We mainly employ PKI technology to manage an employee's identity of an organization. Governments in Belgium, France and many countries are considering a national ID card system based on PKI to authenticate a citizen's identity. ID management based on PKI technology is growing in the public sector world wide.

Fig. 4 shows the details of user authentication in a secure VMM system. This process is a simple challenge and response authentication procedure. A user inputs a PIN number, and a smart card authenticates the user. An authentication handshake is executed between the smart card and an ID management system of the secure VMM. The secure VMM sends a challenge as R , and the smart card then returns the user's Public Key Certificate (PKC_{USER}) and a signature of R ($\{R\}_{USER}$) generated by the user's private key ($PrivateKey_{USER}$). The ID management system validates the certificate chain using a trust anchor certificate for a VMM ($PKC_{TRUST\ ANCHOR\ (VMM)}$). In addition, an ID management system checks certificate revocation status by a Certificate Revocation List (*CRL*) or some other mechanism like an Online Certificate Status Protocol (OCSP). A secure VMM and the ID management software should be protected from malicious attacks by an attestation mechanism and an isolation mechanism of VMM. If authentication is successful, then a secure VMM will be able to enforce security policy based on a USER ID in the protected VMM layer. Thus, an administrator-controlled secure VMM and the ID management

framework provide the fundamental part of an end-point policy enforcement mechanism in an organization.

Fig. 4 shows unilateral authentication. However, our proposal can provide mutual authentication using a user's trust anchor certificate ($PKC_{TRUST\ ANCHOR\ (USER)}$) in the smart card. A secure VMM system may have a transparent traffic security function like a VPN (VM's outbound traffic is automatically encrypted and its integrity is automatically checked). The ID management framework can provide mutual authentication for the VPN services by PKC_{USER} and $PKC_{TRUST\ ANCHOR\ (USER)}$.

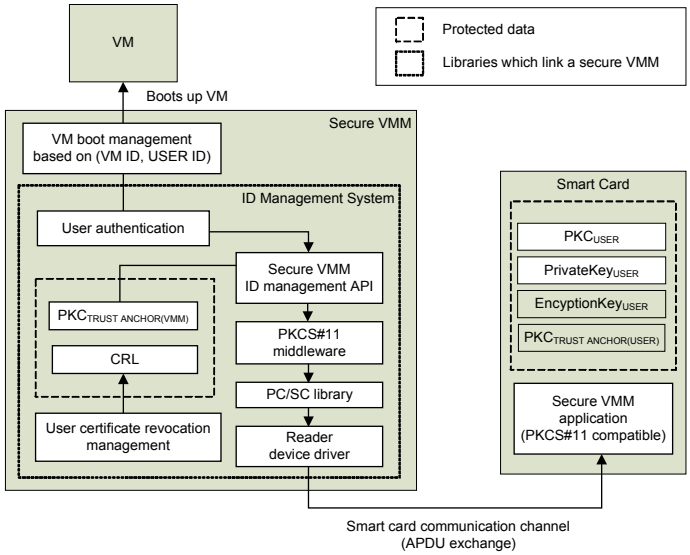


Fig. 3. Basic design of a portable ID management framework for a secure VMM using a smart card (ID card): An example of *VM boot management* based on a *VM ID* and an authenticated *USER ID*.

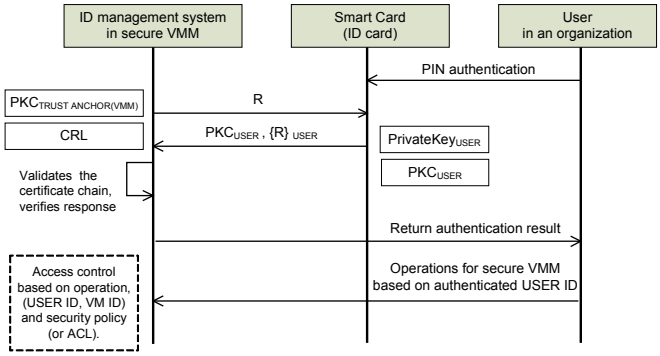


Fig. 4. Sequence of user authentication: A user logs in to an administrator-controlled secure VMM system.

4.2 Example: Transparent VM image Encryption/Decryption

Fig.5 shows an example of encryption/decryption for VM image files. In this case, an ID management framework handles a user's disk encryption key. An encryption key (*EncryptionKey* $_{USER}$) is stored in the user's smart card. A secure VMM decrypts the VM images by retrieving the key from an authenticated smart card. Each encryption key is associated with the user's identity. So an attacker without a valid and user-specific smart card can not decrypt the VM image. The encryption/decryption of a VM image file is automatically and transparently executed by a hook point in a secure VMM layer. This is a good example of a transparent security function in a VMM layer using our ID management framework.

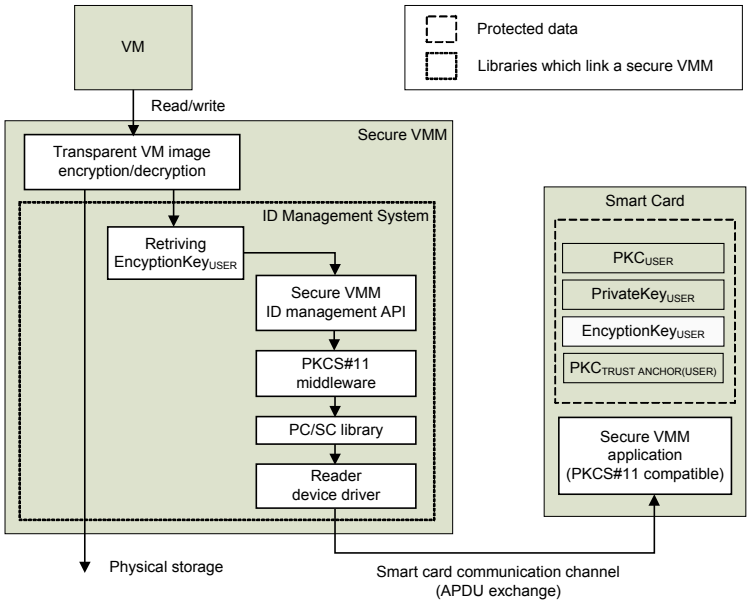


Fig. 5. Example usage of the ID management framework (Transparent VM image encryption): A smart card (ID card) can store a user-specific key (e.g. a 256 bit AES encryption key) for storage encryption or for other user-specific binary data.

5 Prototype Implementation

We have implemented the portable ID management framework for a secure VMM system shown in Fig.3. In our prototype implementation, we used open source software to develop the portable ID management software. Table 1 shows the hardware and software environment for prototype implementation. We have implemented the ID management software on both FreeBSD and Linux operating systems on x86 architecture. We employed the *eLWISE* smart card manufactured by NTT Communications. The eLWISE is also used as the Japanese national ID card (i.e. the *Basic Resident Register* system in Japan). We have implemented original

PKCS#11 middleware and its high level API and our portable *secure VMM ID management API* . We have merged our portable ID management software with proven open source VMM software, *QEMU* [13]. In the current implementation, we have realized APIs for VM boot management based on a USER ID and VM image encryption/decryption by a user's encryption key (Fig.3 and Fig.5). Our portable ID management software is an independent module, so it will be applicable to other VMM software.

Table 1 . Hardware and software environment for prototype implementation .

Smart card	eLWISE (NTT Communications)
Smart card reader	ASE drive IIIe (Athena Smartcard Solutions)
OS	FreeBSD 6.1 / Linux Fedora Core 6
Cryptographic libraries	OpenSSL 0.9.8d [14]
PC/SC library	PCSC-Lite 1.3.2 [15]
USB CCID driver	Athena CCID driver [16]

6 Future Work

Our prototype implementation is constructed on standard middleware APIs (PKCS#11 and PC/SC), so our software is portable and can utilize many consumer off-the-shelf smart card products. We are currently refining our prototype implementation to reduce some library and system call dependencies, so as to integrate our software into thin and lightweight VMM software which runs directly on hardware (i.e. *Type I VMM*). We are working towards improved ID management software to achieve OS and library independence, and more portable and higher performance.

In this paper, we show a design and implementation of a secure VMM component only. However, for the operational phase, we need to consider the following systems: an issuance server for an ID card, a revocation management server for a PKC (CRL, OCSP or some other mechanism), a security policy distribution server, and an update mechanism for the ID card's contents. If a user stores his or her disk encryption key, we must consider a key escrow and recovery mechanism for a lost or stolen ID card. Our framework has a digital signature API for authentication purposes. If a digital signature function is used to guarantee validation and non-repudiation of data, we will need a mechanism to manage key and certificate history.

7 Conclusion

Conventional VMM software does not consider ID management and its standard interface. Our ID management framework defines standard APIs for secure VMM software, such as a smart card-based user authentication API based on PKI, a simple ID/password authentication API without a smart card, a VM boot management API, a storage API for an encryption key, and so on. We have implemented the ID management framework using a standard PKCS#11 compatible smart card and open source software. We are currently refining the ID management software to reduce

library and system call dependencies, so as to integrate our software into thin and lightweight VMM software which runs directly on hardware. Our ID management framework can be used to construct a security enforcement mechanism on end-user terminals.

Acknowledgments. This work was supported by Special Coordination Funds for Promoting Science and Technology of the Ministry of Education, Culture, Sports, Science and Technology of Japan.

References

1. Stuart E. Madnick and John J. Donovan.: *Application and analysis of the virtual machine approach to information system security and isolation,* Proceedings of the workshop on virtual computer systems, ACM Press (1973) 210-224.
2. L. Seawright and R. MacKinnon: *VM/370 – a study of multiplicity and usefulness,* IBM Systems Journal (1979) 4–17.
3. Reiner Sailer, Trent Jaeger, Enriquillo Valdez, Ramon Caceres, Ronald Perez, Stefan Berger, John Linwood Griffin, Leendert van Doorn: *Building a MAC-Based Security Architecture for the Xen Open-Source Hypervisor,* ACSAC2005, IEEE CS (2005) 276-285.
4. Paul A. Karger: *Multi-Level Security Requirements for Hypervisors,* ACSAC2005, IEEE CS (2005) 267-275.
5. Dragovic B., Fraser K., Hand S., Harris T., Ho A., Pratt I., Warfield A., Barham P., and Neugebauer R.: *Xen and the Art of Virtualization,* Proceedings of the ACM Symposium on Operating Systems Principles (2003).
6. IBM Research: *The Research Hypervisor – A Multi-Platform, Multi-Purpose Research Hypervisor,* `http://www.research.ibm.com/hypervisor/`
7. TCG TPM Specification Version 1.2, `http://www.trustedcomputinggroup.org/`
8. R. Sailer, X. Zhang, T. Jaeger, and L. van Doorn: *Design and Implementation of a TCG-based Integrity Measurement Architecture,* In Thirteenth USENIX Security Symposium (2004) 223-238.
9. Meushaw R. and D. Simard: *NetTop: Commercial Technology in High Assurance Applications,* National Security Agency Tech Trend Notes (2000) 3-9
10. VMware: `http://www.vmware.com/`
11. Tal Garfinkel, Ben Pfaff, Jim Chow, Mendel Rosenblum, Dan Boneh: *Terra: a virtual machine-based platform for trusted computing,* Proceedings of the ACM Symposium on Operating Systems Principles (2003) 193 – 206.
12. W. A. Arbaugh, D. J. Farber and J. M. Smith: *Secure and Reliable Bootstrap Architecture,* in IEEE Computer Society Conference on Security and Privacy (1997) 65-71.
13. QEMU open source processor emulator: `http://fabrice.bellard.free.fr/qemu/`
14. PCSC-Lite: `http://pcsclite.alioth.debian.org/`
15. OpenSSL: `http://www.openssl.org/`
16. Athena CCID driver: `http://www.athena-scs.com/downloads.asp`

Proceedings of the International Multiconference on Computer Science and Information Technology pp. 701–711
ISSN 1896-7094
© 2007 PIPS

Access Control for Cross-Organisational Web Service Composition

Michael Menzel[1], Christian Wolter[2], Christoph Meinel[1]

[1] Hasso-Plattner-Institute for IT Systems Engineering
University of Potsdam, Germany
{michael.menzel,meinel}@hpi.uni-potsdam.de
[2] SAP Research
Vincenz-Priessnitz-Str. 1, 76131 Karlsruhe, Germany
{christian.wolter}@sap.com

Abstract. Service Oriented Architectures (SOA) promise a flexible approach to utilize distributed capabilities that may be located in independent trust domains. These capabilities can be exposed using Web Service technologies, which provide functionality to describe, discover, and invoke exposed services across organisational boundaries. A broad range of SOA-platforms and toolkits are available focusing on Web Service enabling and orchestration within an organisation. This paper addresses an evaluation and classification of different SOA-platforms and security frameworks regarding secure cross-organisational service invocation. To overcome the revealed limitations of existing frameworks, a two layered security architecture is introduced that satisfies the identified security requirements and abstracts from local access control models to enable secure federated cross-organisational services compositions.

1 Introduction

The standard set of Web Service technologies, such as WSDL, UDDI, and SOAP provides the means to describe, locate, and invoke a Web Service as part of a Service Oriented Architecture. These standards, based on XML, remove the dependencies of operating systems and system architectures, allowing a composition of different services. Service composition is one of the success factors of Service Oriented Architectures in order to enable the flexible integration of services provided by independent business partners. The term of service federation reflects that services are composed across organisational boundaries to create additional business value for all members of the federation.

However, the seamless and straightforward integration of cross-organisational federated services conflicts with the need to secure and control the access to provided services. Traditional access control is based on a dedicated model, such as role-based access control, access control lists, or task-based access control [1]. In a service federation, however, there might exist multiple models with different semantics or expressiveness for each federation member. In addition, access control decisions in classical models are based on identities and permissions assigned to them. In a federated service the identities and their assigned permissions cannot be exposed to all participants

of the federation in order to prevent information leakage of security policies or due to model inconsistencies (i.e. different role concepts, unknown attributes). Also, system evolution, such as new employees are hired or delegation of permissions takes place, must be mediated to all partners. Thus, classical access control architectures provide a non-feasible approach in terms of the general independence of services in a Service Oriented Architecture.

In this paper we propose a Two-Level Access Control (2LAC) architecture for cross-organisational federated service composition independent from local access control models—thus, preventing information leakage and allowing authorization-based cross-organisational service invocation [2]. Therefore, we provide:

- A list of access control and authorization requirements for federated composite web service frameworks.
- An evaluation and categorization of existing SOA security frameworks and their capabilities to support cross-organisational federated composite services.
- A proposal of a two layered security architecture that addresses the identified security requirements and abstracts from local access control models in place.

The rest of the paper is organized as follows. The next section provides a list of security requirements for cross-organisational federated service compositions. In Section 3 we will briefly mention related SOA security architectures, such as Amberpoint, Crossvision, PERMIS, or the WS-Federation architecture. Each framework will be evaluated and categorized based on our identified security requirements. In Section 4, we introduce the concepts of our proposed 2LAC architecture for secure federated cross-organisational web service composition based on the authorization-based access control paradigm [2]. The last section concludes this paper and gives an overview about the ongoing implementation of 2LAC and outlines future work.

2 Case Study for Federated Access Control

To illustrate the requirements for cross-organisational federated service invocation, consider the example presented in Figure 1, involving three business partners (a manufacturer and two suppliers). The manufacturer produce goods using components delivered by the suppliers. A service that composes the suppliers order services is used by the manufacture's staff to place orders.

Six basic requirements can be derived from the case study to facilitate secure, compliant, and flexible cross-organisational service invocation:

- Independent Access Control Models
  The dynamic nature of SOA demands a flexible and straightforward approach to establish and administrate service federations. Services provided by new business partners have to be integrated seamlessly without severe changes in the security infrastructure. Especially, the security and access control models used by the respective service providers must be independent from each other. Dictating a common access control model is a non-suitable approach in a federation, hence domain specific access control models might be necessary.

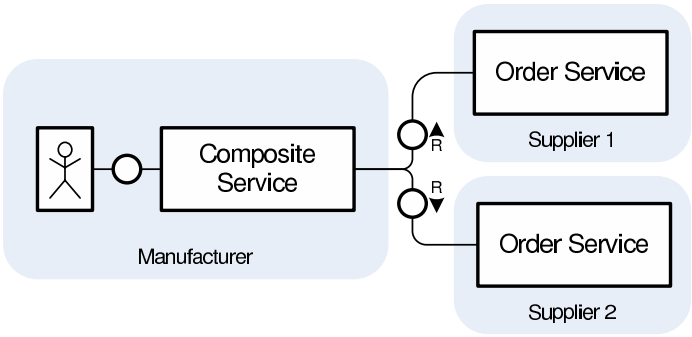


Fig. 1. Use Case Service Composition

– Service Level Agreements
  To establish a federation with a new business partner, the manufacturer and the new supplier have to agree on IT-level and business process-level service parameters. This agreement affects the access control decision that is also based on environment attributes, such as time and access statistics.

– Information Leakage
  Business roles, information about staff, and the service provider's security infrastructure are sensitive information which should be kept confidential to avoid information leakage. Especially, if the service provider deals with frequently changing business partners, continuously exchanging security policies is inapplicable.

– Dynamic Adaptation of Security Policies
  Once a federation is established, continuous changes relating to permissions assigned within a partner's domain might be necessary. For example changing personal, system evolution, and administrative, or ad-hoc delegation of access permissions [3] might be a reason, but this information must not be mediated to federation members in order to prevent flooding and revocation of authorization information.

– Trusted Service Invocation
  The trustworthy interaction with exposed services is essential for the federated business operations. To avoid business process disruption the access to services must be limited exclusively to authorized staff defined by security policies.

– Compliance
  As part of the Public Company Accounting Reform and Investor Protection Act that is known as SOX [4], auditing and reporting has to facilitate the verifiability that all actions in the federation comply with policy constraints, service level agreements, and law regulations. This is an important aspect regarding the federated delegation of authorizations in order to further guarantee compliance to corporate governance, internal control assessments, and public agencies.

3 Evaluation of Security Framework Architectures

In this section we analyze existing security frameworks with respect to access control for cross-organisational composite web services. Although, these frameworks and platforms focus on entirely different security and SOA aspects, they can be categorized in different groups based on their application to protect a service in a federated environment. We propose a classification depending on the distribution of authentication and authorization information. For each category we present existing example frameworks along with a short description.

3.1 Frameworks for Service Managed Policies

Frameworks based on *Service Managed Policies* facilitate the service to store and handle all information for access control. The identity of the service requester and its role is usually the most important aspect to grant access. Since all this information needs to be maintained for each user who is allowed to access the service, an initial registration of users is required (cf. Figure 2). Beside the exposure of identity related information, the inexistent possibility to delegate authorizations in the user domain is adverse. Furthermore, this approach requires the user to adopt the authentication method specified by the service provider. The interaction between user and service provider will fail if different security infrastructures are used, probably supporting incompatible ways for authentication.

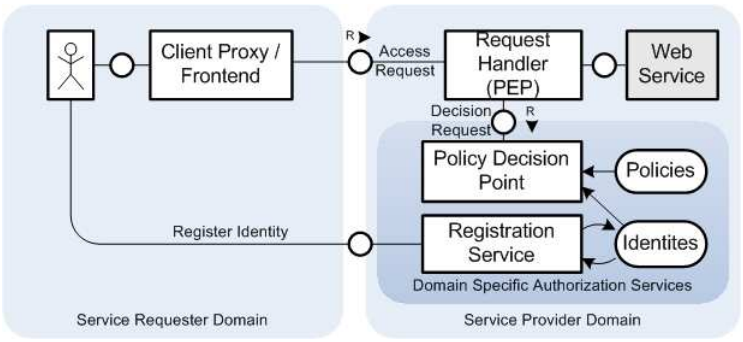


Fig. 2. Service Managed Policies

A representative example for *Service Managed Policies* is the Software AG Crossvision Suite [5], which provides a set of tools and components for service enabling, orchestration, and integration in an enterprise environment. One of the central components of Crossvision is the Service Orchestrator that can be used to compose services to a process defined with BPEL [3]. The Service Orchestrator supports message-level security based on WS-Security, and a public key infrastructure (PKI). However, since

[3] Business Process Execution Language

these components are linked to a predefined java keystore containing the certificates of either authorized users or the issuing certificate authority, only an identity-based access control can be realized.

All aforementioned disadvantages related to service managed policies apply here. The secure invocation of Web Services across domain boundaries requires all participating organisations to deploy a PKI. Furthermore, each external user has to pass his certificate over to the service provider in an initial registration step to enable the verification and authorization of signed requests. Since the authorization decision is tied to the user's identity, access control in a federation is hard to manage in this way. Although a basic secure cross-domain invocation of Web Services is enabled by using a PKI, the general problem remains that such a trust domain cannot interact with a domain that is based on another security solution, such as Kerberos.

3.2 Frameworks for Equal Sharing of Policy Information

Equal sharing means the policy information is maintained by the client along with the service provider. This can be realized based on direct policy exchange, a central federation policy repository, or a dedicated authentication/authorization services (cf. Figure 3). This approach simplifies administrative aspects, although information has to be exposed to the service provider. The delegation of authorization is enabled since the user domain has the possibility to adapt security policies. Nevertheless, the establishment of a federation is complicated, due to the necessity to adopt the central security settings for each local infrastructure and domain-specific individual security requirements are hard to be supported by this approach.

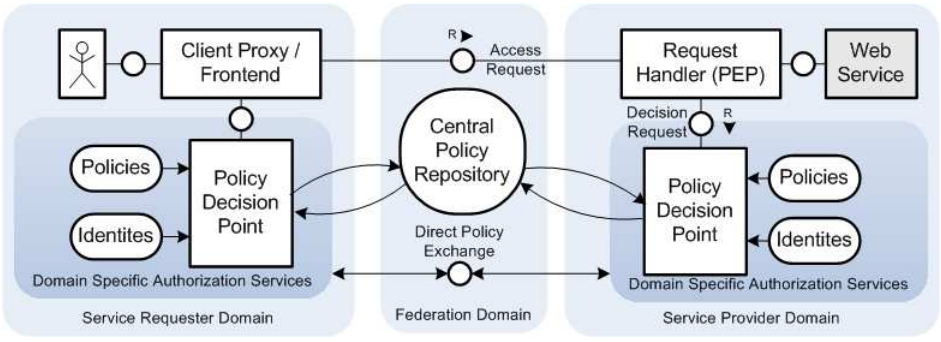


Fig. 3. Equal Shared Policies

The Privilege and Role Management Infrastructure Standards Validation (PERMIS) framework is a policy-based authorization infrastructure implementing a hierarchical Role Based Access Control (RBAC) model [6]. It is based on a privilege management infrastructure to maintain the user attributes in X.509 attribute certificates (AC), which are published in LDAP directories. These certificates describe the relationship between

identities and their privilege attributes and are signed by an Attribute Authority. A central authority defines the access control rules that are translated into an authorization policy and stored as an attribute certificate by an LDAP directory service.

PERMIS has been designed to enable access control within a single organisation. Therefore, it provides an enhanced security functionality, such as an hierarchical RBAC model and delegation of authorities. However, an extension is described in [6] to enable PERMIS to issue SAML authorization tokens. The approach uses PERMIS as a central authorization service in a distributed environment. Although, a centralized solution avoids information leakage and supports security principles, such as Separation of Duties, some issues still remain. All organisations in a federation must use PERMIS as a centralized security solution and have to agree on a common role model.

Another approach is the proposed multipolicy authorization framework for Grid infrastructures by Lang *et al.* in [7]. This framework is based on Web Service security specifications, such as XACML and SAML. The authorization mechanisms of the Grid computing platform supports multiple security policies and dynamic policy changes. The access decisions are based on a requester's attributes, such as the service, the resource, or the environment. Therefore, whenever a new identity is added to the service requester's side this identity information must be mediated to the service domain. The general concept of centralized resource and authorization management does not fit into the concept of Service Oriented Architectures. Nevertheless, the GT4 framework utilizes a promising set of security specifications that will be adapted by our proposed architecture.

3.3 Frameworks for User Managed Policies

In the context of *User Managed Policies* the service provider may store some local policies necessary on the provider's side, while the policy information stored at the service requester's domain is independent from the provider's side. This means that the identity of the service users—and therefore the authentication policies—are solely managed and known in the user domain. There are no cross domain policies used—the policies of all organisations in the federation are restricted to the respective security domain. This enables each domain to utilize an own security model independently from others. Asserted user attributes have to be conveyed along with each request to the service (cf. Figure 4). The service will grant access based on these asserted user information if the asserting authority in the user domain is trustworthy.

Although, this approach decouples the security infrastructure used in the different trust domains, a common understanding of the exchanged attributes is still required. For example, the involved organisations may have a different understanding of roles and identities. This would require complicated mapping mechanisms to translate these attributes. Information leakage is still an issue, since identity and role information must be provided to other organisations anyway.

The *Amberpoint SOA Management System (ASMS)* [8] is a solution for SOA runtime governance. The ASMS is based on SOAP message interception and forwarding of service requests and responses. It is policy-driven to enforce the requirements concerning authentication, authorization, and integrity. Authentication can be performed by ASMS based on a local identity management system, WS-Security signed messages, or

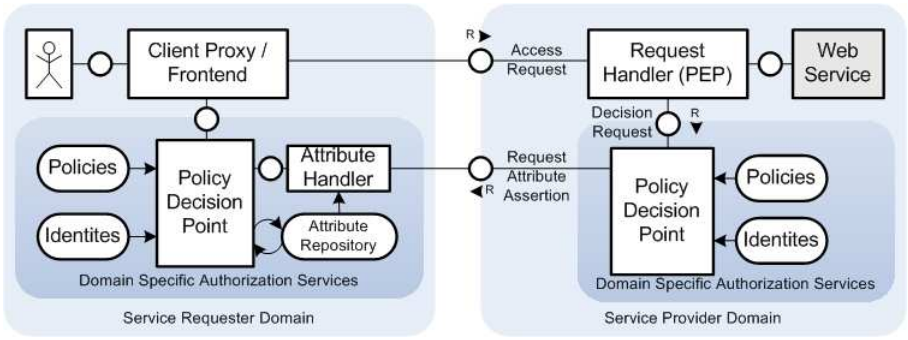


Fig. 4. Client Managed policies

a SAML Authentication Assertion. Based on this information the authorization is performed using basically a simple RBAC model.

Since the supported identity management systems are not designed to be used in a federated environment, SAML is supported to enable secure service invocations across trust domains. An *User Managed Policy* approach can be realized in this way by requiring the authorization to rely on the user's role that is provided by the SAML-token. However, all organisations in a federation must have a common understanding of roles.

A standardized foundation for federation security is provided by the *Web Service Federation language (WS-Federation)* [9] and the *Liberty Alliance* [10] defining frameworks to federate independent trust domains. These specifications provide models for security token exchange to enable the brokering of identity, the discovery and retrieval of attributes, and the provision of security claims. A meta-data model to describe and establish a federation is introduced as well [11]. Altogether, WS-Federation as well as Liberty Alliance are designed to enable the use of identity attributes across trust domains to facilitate authorization decisions. Since common problems such as the need to map attributes or the provision of pseudonyms are addressed, these specifications provide a standardized model for federations based on user managed policies.

4 2-Level Access Control Architecture

Our classification reveals that each security solution has its own strength, but also specific weaknesses. Even WS-Federation as well as Liberty Alliance presume a common understanding of particular attributes and claims. Since this standard is focused on the brokering of attributes, it does not fully meet the requirements defined in Section 2. Based on the WS-Trust standard, we propose an architecture that meets these requirements.

4.1 Cross-Organisational Access Control

In the domain of federated services, Kang *et al.* [12] describe a security architecture that proposed decoupled security infrastructures by introducing a shared role domain

containing organisation independent roles and their relationships within the federation. Each federation partner maintains its own specific policies and translates it to the predefined shared roles of the federation. Although this approach does not fit into the concept of short living service federations, the basic idea to decouple security infrastructures by separating the local access control model from the cross-organisational one constitutes the foundation for our solution to enable secure cross-domain service invocation.

While the local access control model should depend solely on the local security infrastructure and requirements, the authorization-based access control model [2] is used for cross-organisational access control. The authorization decision is made and asserted in the service requester's domain and attached to the request that invokes the service in a service provider domain. Requesting domain and service domain form a federation which is characterized by a trust relationship. Therefore, requests are only accepted by the service provider if a valid security token from a trusted domain is provided along with each service request. Since the server-side authorization component handles the relation between the domains, service level agreements are enforced by this component according to parameters, such as access time or access statistics.

The user-to-service relation is handled by the user-side authorization component using the local security infrastructure to perform the authentication and the authorization. Since each organisation has its own sphere of local responsibility, we have a two-fold Policy Enforcement Point (PEP) hierarchy. In addition, the identity related policies reside in the domain of the service requester and no policy exchange takes place.

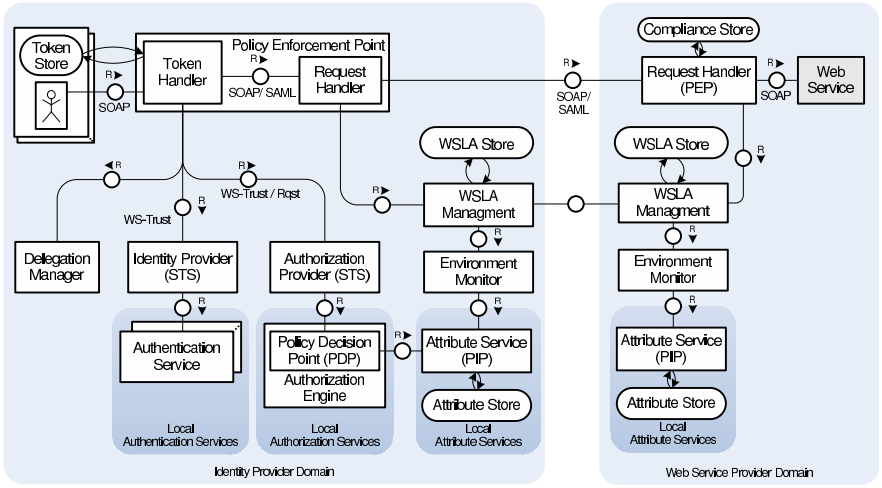


Fig. 5. The 2LAC Security Architecture

4.2 Service Invocation

A complete abstract of the 2LAC architecture is given in Figure 5. The *Token Handler* is a local component in the user domain that initiates the client-side authentication process

and manages the authentication and authorization tokens for the user. These tokens are requested by contacting the Security Token Services (Identity Provider and Authorization Provider), which serve as an interface to the local authentication and authorization mechanisms. Once a valid authorization token is received, the token is conveyed with the request to the *Request Handler*. The *Request Handler* verifies that the request complies with the service level agreements, encrypts the message and invokes the service.

In the service domain the *Request Handler* acts as a second PEP and intercepts incoming requests. Similar to the *Request Handler* in the user domain, requests are decrypted and verified according to service level agreements. It is ensured that a trusted party has authorized this request by validating the security token provided by the requesting domain. Subsequently, the authorization decision is enforced by the *Request Handler*. If access is granted, information about the request such as the authorization token will be stored in the compliance store and the service will be invoked.

4.3 Trust Brokering

Authorization tokens are conveyed with a request and enable an user to access a particular service. These tokens reflect the user's permission to perform an operation on this service and are valid for a predefined period of time. Issued tokens are stored in the user's *Token Store* and are maintained by the *Token Handler*.

For each incoming request, the *Token Handler* tries to obtain a token from the *Token Store*. If a token for a request is not present in the store or has to be validated, the handler will commit an authorization token acquisition for a requested service by contacting the security token services in his domain.

The usage of security tokens enables the delegation of authorizations independent from the local security infrastructure. User can instruct the *Token Handler* to delegate authorizations to other users. This is realized by transferring the authorization token to the other user's *Token Store*. The handler utilize the *Delegation Manager* to enable revocations later on and the PDP to authorize the delegation, since it can be restricted by policy constraints, such as separation of duties.

Each authorization token is created with a unique number to manage and audit the delegation of these tokens. Since the service provider records all incoming authorization tokens in the compliance store, this information will be provided to the user domain if an abuse of authorizations is suspected.

5 Conclusion

The 2LAC Security Architecture enables the secure interaction of independent trust domains in a federation. In contrast to other security frameworks 2LAC decouples the organisational and the cross-organisational access control to allow domain-specific security solutions. All evaluated security solutions can be used in the 2LAC Architecture as local security infrastructure and are encapsulated using WS-Trust interfaces.

Since each organisation has its own sphere of responsibility—one domain handles the trust relationship user-to-organisation, while the other is responsible maintain the

trust relationship organisation-to-service—there is no need to exchange user information or to perform an error-prone attribute mapping. Only the support of SAML as the used token type to express authorization claims is presumed. In addition, a flexible delegation of authorizations is enabled independently from the local security infrastructure. Exchanged service level agreements are also enforced. Therefore, 2LAC meets all the requirements specified in Section 2.

Nevertheless, these requirements depend on the trust relationship and the use case. In some cases, the provision of identity information may be required, e.g. identity information has to be provided to a credit card service. If the exposure and the exchange of identity attributes is an essential requirement in an use case than WS-Federation will provide a suitable framework for attribute based security. However, if any kind of information leakage has to be avoided and a clear separation of local and cross-organisational access control is needed, than 2LAC is an appropriate solution.

5.1 Future Work

There is an ongoing implementation effort for the proposed 2LAC architecture. Parts of the security infrastructure, such as the token exchange mechanisms, are already in place. We further plan to evaluate the performance implications of our security infrastructure to the overall service response time. Another aspect is the integration of existing SOA security frameworks, such as Amberpoint or Crossvision, by defining extension points to attach our proposed authorization-based components to the local authorization infrastructure. A third option would be to translate and apply our architecture into the context of virtual organisations, such as described by the European research projects TrustCom [13] and Serenity [14].

References

1. Christian Liesegang, Mathias Kohler, and Andreas Schaad. Classification Model for Access Control Constraints. In *WIA—Proceedings of the 1st International Workshop on Information Assurance*, New Orleans, USA, 2007.
2. Alan H. Karp. Authorization-based access control for the services oriented architecture. In *C5 '06: Proceedings of the Fourth International Conference on Creating, Connecting and Collaborating through Computing (C5'06)*, pages 160–167, Washington, DC, USA, 2006. IEEE Computer Society.
3. Andreas Schaad and Jonathan D. Moffett. A Framework for Organisational Control Principles. In *ACSAC*, pages 229–238, 2002.
4. Hugh Taylor. *The Joy of SOX: Why Sarbanes-Oxley and Services Oriented Architecture May Be the Best Thing That Ever Happened to You*. John Wiley & Sons, Inc., New York, NY, USA, 2006.
5. Sandra Rogers. CentraSite: An Integrated SOA Registry and Repository. `http://www.idc.com`, 2006.
6. David Chadwick, Sassa Otenko, and Von Welch. Using SAML to Link the GLOBUS Toolkit to the PERMIS Authorisation Infrastructure. In *Proceedings of Eighth Annual IFIP TC-6 TC-11 Conference on Communications and Multimedia Security*, Windermere, UK, September 2004.

7. Bo Lang, Ian Foster, Frank Siebenlist, Rachana Ananthakrishnan, and Tim Freeman. A multipolicy authorization framework for grid security. In *NCA '06: Proceedings of the Fifth IEEE International Symposium on Network Computing and Applications*, pages 269–272, Washington, DC, USA, 2006. IEEE Computer Society.
8. AmberPoint Software. Introduction to AmberPoint SOA Management System. AmberPoint Software, June 2006.
9. Web Services Federation Language Specification 1.1. `http://www.ibm.com/developerworks/library/specification/ws-fed/` 2007.
10. Liberty Alliance Specification. `http://www.projectliberty.org/liberty/specifications_ _1`, 2007.
11. Nadalin Michael McIntosh Marc Goodner, Maryann Hondo and Don Schmidt. White Paper: Understanding WS-Federation. `http://www.ibm.com/developerworks/library/specification/ws-fed/`, 2007.
12. Joon S. Park Mzong H. Kang and Judith N. Froscher. Access Control Mechanisms for Inter-Organizational Workflow. *Proc. SACMAT'01*, pages 66–74, 2001.
13. Michael D. Wilson, Alvaro Arenas, and Lutz Schubert. TrustCom Framework V2. `http://www.eu-trustcom.com`, 2006.
14. Paraskevi Kostaki, Spyros Kokolakis, and C. Pandolfo. Serenity—System Engineering for Security & Dependability WP A2.D4.1. `http://www.serenity-project.org` 2006.

International Conference on Principles of Information Technology and Applications (PITA'07)

International Conference on Principles of Information Technology and Applications (PITA07) is the most general Event of the IMCSIT. PITA invites contributions presenting research results in any area of computer science or information technology, which are not covered by other Events of the IMCSIT.
TOPICS include but are not limited to:

- Computer systems development and maintenance
- Foundations of computer science
- System dependability
- Modeling and simulation
- Formal methods
- Numerical computing
- Parallel and distributed processing
- IT applications
- E-country, E-learning
- IT Management

We are especially interested in any novel, may be controversial solutions. The interdisciplinary contributions and papers that hardly classify to typical topics of conferences are invited as well.
We would like to extend a particularly warm welcome to young researchers whose research is supported by EU, national and local grants. We are interested in papers concerning theory and applications as well as case studies of successful transfer of technology from academia to industry.

International Programme Committee

- Witold Abramowicz, The Poznan University of Economics, Poland
- Wojciech Cellary , The Poznan University of Economics, Poland
- Zbigniew Czech, Silesian University of Technology, Poland
- Marek Druzdzel, University of Pittsburgh, USA
- Dariusz Dziuba, Warsaw University, Poland
- Eugene Eberbach, University of Massachusetts Darmouth, USA
- Jozef Goetz, University of La Verne, USA
- Andrzej Gościński, Deakin University, Australia
- Jarek Gryz, York University, Canada
- Anna Hać, University of Hawaii, USA
- Olgierd Hryniewicz, Systems Research Institute, Polish Academy of Science, Poland
- Michal Iglewski, Université du Québec en Outaouais, Canada
- Ryszard Janicki, McMaster University, Canada
- Stanisław Jarząbek, National University of Singapore, Singapore
- Lech Józwiak, Eindhoven University of Technology

- Jerzy Kisielnicki, Warsaw University, Poland
- Mieczysław Kłopotek, Institute of Computer Science, Polish Academy of Sciences, Poland
- Krzysztof Kochut, University of Georgia, USA
- Janusz Konrad, Boston University, USA
- Jerzy Korczak, University Louis Pasteur, France
- Michael L. Korwin-Pawlowski, Université du Québec en Outaouais, Canada
- Maciej Koutny, University of Newcastle upon Tyne, UK
- Jacek Kryt, Ryerson University, Canada
- Andrzej Lingas, Lund University, Sweden
- Leszek Maciaszek, Macquarie University, MACOMP Software, Australia
- Zygmunt Mazur, Wroclaw Uniwersity of Technology, Poland
- Boleslaw Mikolajczak, University of Massachusetts, USA
- Mieczyslaw Muraszkiewicz, Warsaw University of Technology, Poland
- Tomasz Müldner, Acadia University, Canada
- Marek Ogiela, AGH University of Science and Technology, Poland
- Wojciech Olejniczak, Technical University of Szczecin, Poland
- Witold Pedrycz, University of Alberta, USA
- Pawel Pietrzak, Technical University of Madrid, Spain
- Zbigniew Raś, University of North Carolina, USA
- Michael Sobolewski, Texas Tech University, USA
- Janusz Sosnowski, Warsaw University of Technology, Poland
- Wojciech Szpankowski, Purdue University, USA
- Andrzej Śluzek, Nanyang Technological University, Singapore
- Andrzej Stanisław Targowski, Western Michigan University, USA
- Wlodek M. Zuberek, Memorial University, Canada
- Jozef Zurada, University of Louisville, USA

Organizing Committee

Anna Derezinska, Warsaw University of Technology, Poland
Tomasz Pelech-Pilichowski, AGH University of Science and Technology, Poland
Krzysztof Wecel, Poznan University of Economics, Poland

Proceedings of the International Multiconference on
Computer Science and Information Technology, pp. 715 – 724
ISSN 1896-7094
© 2007 PIPS

Application of Model Transformation in the Generic Framework for Traceability

Anna Derezińska, Jacek Zawłocki

Institute of Computer Science, Warsaw University of Technology, Nowowiejska 15/19,
00-665 Warsaw, Poland
A.Derezinska @ii.pw.edu.pl

Abstract. Model transformation approach allows us to develop automatic and flexible solutions for the software evolution. Application of model transformation concepts was shown for the generic framework for traceability in object-oriented designs. Three transformations within the framework were considered: the input transformation of any model to the internal format, the traceability analysis generating a dependency area for a given model, and the output transformation of the resulting dependency area. They can be realized as model-to-model transformations with respect to their metamodels and in accordance to the independently specified transformation rules. In the input and output transformations the language and tools of the QVT standard proposed by the OMG were applied. In the remaining transformation, traceability rules were defined as automata with transitions labeled with conditions and actions.

1 Introduction

Model Driven Development (or Model Driven Engineering) is an idea promoting the shift from the code-oriented to model-oriented software production techniques. An important role is played by models and their transformations.

Model Driven Architecture™ (MDA) [1,2] was intended as the realization of the model engineering principles around the set of the Object Management Group (OMG) standards. One of them is QVT (Query/View/Transformation) [3] devoted to model transformation.

Many approaches to model-to-model transformation have been proposed [4], but there is a lack of sufficient experience in their practical applications. Transformation capabilities of modeling tools available on the market are growing up, but are still premature. An important question is how well QVT and other MDA-based solutions fit for different kinds of transformations [5].

In this paper, we show a practical application of a model transformation in the development of a generic software system. The system is a framework for traceability of object-oriented projects [6,7]. A distinguishing feature of the generic framework is its flexibility. One of the basic concepts is separation of application logic and its implementation. The same postulate is true for transformation i. e., separation of transformation rules encapsulating the internal logic and the transformation execution. Advantages but also drawbacks of the transformation-based solutions are presented.

Traceability refers to relations, which are defined among different artifacts of a software development process. It is beneficial in requirements verification and

evolution, models and code development and maintenance, tests generation and management [8-10].

Traceability issues addressed in our framework are aimed at object-oriented projects. However, those projects can be described at different abstraction levels e.g., consisting of analytical or implementation classes, but also general concepts from a given domain. The framework is based on the original ideas of dependency areas developed by one of the authors [11,12], but it is not limited to the traceability strategy proposed for UML. The framework can help solving different problems related to:

- identification of dependencies and inconsistencies in a project,
- impact analysis of changes within a project,
- support for model understanding, including reverse engineered models and legacy systems,
- creating documentation of a project,
- support for model and code instrumentation.

The rest of the paper is organized as follows. Section 2 explains basic concepts of model transformation. Next section summarizes briefly main features of the generic framework for traceability in object-oriented designs. Section 4 describes selected transformation issues within the framework. Final remarks conclude the paper.

2 Model transformation

Model transformation deals with manipulation of different system abstractions expressed by models. A variety of software development artifacts can be a subject of transformations. Survey and classification of model transformation approaches can be found in [4].

Model transformation is a key part of MDA that serves as a conceptual framework for an approach to model-driven development. For creating transformations we can apply the OMG specification MOF QVT [3]. The QVT Relations language is a declarative language for model-to-model transformation. It can be used for writing own transformations or adopting some existing transformations to our needs.

In order to make a transformation between different notations of a model, or between models of different abstraction, we need a common core of basic concepts. In the QVT approach this was achieved using the meta-modeling layers developed by the OMG specifications (Fig. 1). Layer M1 includes models describing systems in different application domains. Model specification languages (e.g., UML) are described by their metamodels (M2 layer). It was provided a language for defining metamodels i.e. a meta-metamodel level, called MOF (Meta Object Facility). QVT supports a transformation of the MOF metamodel and, therefore, a transformation of any models defined using MOF.

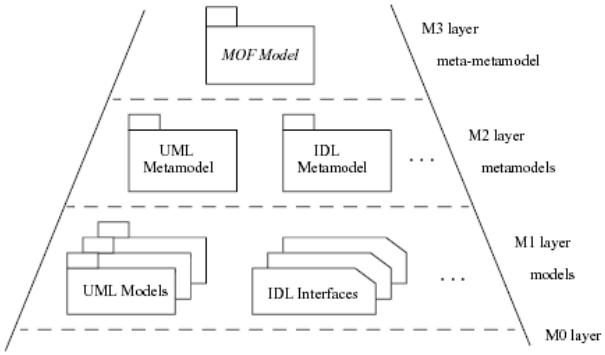


Fig. 1. The role of the MOF in the metamodel hierarchy

A concept of model transformation realized in QVT is illustrated in Fig. 2. There are two groups of models: source models and target models. These models are instances of given metamodels. Metamodels can be in general different but both conform to the meta-metamodel MOF, as they are instances of this meta-metamodel. A transformation is defined with respect to the metamodels. There is a set of rules defining the transformation process. The transformation engine converts a source model to the appropriate target model according to the transformation rules.

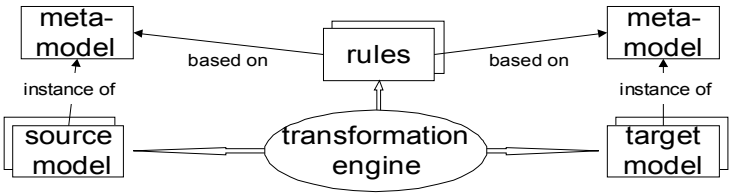


Fig. 2. Idea of model transformation in QVT

Although QVT was developed by OMG as one of specifications within the MDA approach, there are many other solutions used for this purposes [4,5,13]. Transformation rules were successfully described in OCL, but also in other formal notations (e.g., Object-Z, B, Maude).

3 Overview of the Generic Framework for Traceability in Object-Oriented Projects

The framework is devoted to traceability in software designs. Its goal is discovering traceability relationships in a given model according to a given traceability strategy in an automatic way. The framework is based on the traceability concepts of *Dependency Areas* [11,12]. In general, it considers an object-oriented project described at different levels of abstraction. For a given project and an initial element it extracts a subset of the mostly related elements and identifies relations between them. This extraction process can be ruled by different traceability strategies. They determine, for example, kinds of relations between elements that are considered as "the mostly related". A traceability strategy is defined by a set of traceability rules.

The framework was intended to be generic and highly flexible. The configurability of the framework is based on multi-tier architecture, state-machine theory, scripting languages provided to end-users and plug-in mechanisms. An object-oriented model in any notation can be accepted by the framework. It requires only a pre-processing realized by an appropriate input plug-in. It converts any model to an internal form (so-called *Project* notation).

Traceability analysis is performed on a Project model according to a given subset of traceability rules. Each rule is defined by a finite state automaton. Automata are interpreted by an engine of the framework, so-called traceability analyzer. More details about the framework architecture can be found in [6,7].

Application of the generic framework to practical purposes requires its adaptation. It is a role of a *plug-in developer* and a *traceability process modeler* that prepare the platform to be used by a *model designer*. Several analytical and technical tasks should be realized:

- problem recognition and its analysis concerning possibility of solutions with the traceability analyzer,
- selection of model notation for description of the problem,
- preparation of conversion from the given model notation to the notation accepted by the traceability analyzer (Project format),
- design of a traceability logic in dependence of the considered problem (selection among prepared traceability strategies),
- preparation of conversion of a dependency area from the form given by the traceability analyzer to a suitable output form (if required).

An exemplary instance of the framework was implemented. It handles selected parts of the UML metamodel and its traceability. Models can be serialized as Extensible Markup Language (XML) using the XML Metadata Interchange (XMI) [14]. The general process realized within the framework is shown in Fig. 3.

In order to satisfy the flexibility requirements of the framework we made use of model transformation approaches. In the process supported by the framework we can distinguish three main points, where model transformation can be applied:

1. *input transformation*: from an object-oriented model to the model in the internal (Project) notation,
2. *traceability transformation*: from a Project model with a given initial element to the resulting Dependency Area,
3. *output transformation*: from a Dependency Area to a resulting model in a desired notation.

Input transformation was realized using QVT approach and existing, supporting it tools (Sec. 4.2). Output transformation is analogous to the input one. The main,

traceability transformation follows the described concepts, but it was performed by the dedicated traceability analyzer (Sec. 4.1). It transformed models according to the traceability rules defined as the specialized automata. Therefore, the principle of separation between traceability logic and its execution was preserved.

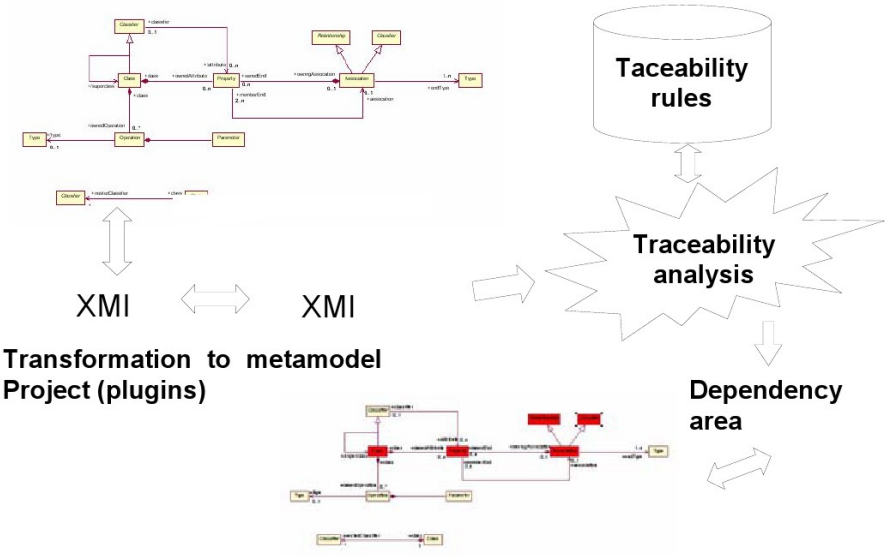


Fig. 3. Realization of traceability process

4 Application of Model Transformations in the Framework

This section describes transformation solutions used in the framework. All of them are consistent with the concepts illustrated in Fig. 2.

4.1 Transformation Process of Traceability Analyzer

Internal form of an object-oriented project is described by the metamodel Project. Traceability rules are designed to operate on any model consistent with the Project metamodel. It is a general form to which models of specific types can be transformed.

A project (class *Project*) consists of many elements (class *ProjectElement*) (Fig. 4). Any element of a project can have any number of annotations (class *LinkAnnotation*) referring to traceability process. Annotations define a set of other elements related in the project, specifying their identifiers, types or names.

Such a project can be understood by the executor of traceability rules. According to the interpretation of MDA transformation, any instance of metamodel Project should be transformed to a dependency model. This model is an instance of the metamodel of Dependency Area (Fig.5). It is a resulting metamodel of the traceability process.

A dependency area aggregates a set of area members. Each member is a specialization of an element of the Project. An area member can have a number of links (class *Link*) to other elements of the dependency area. Links can have their priorities and types, defined according to performed traceability rules.

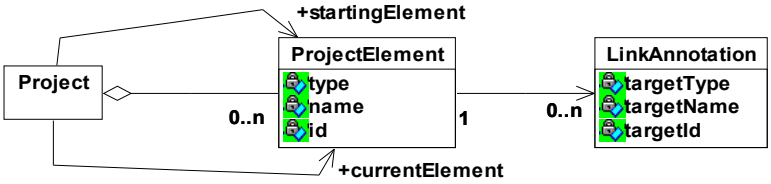


Fig. 4. The core of the Project metamodel

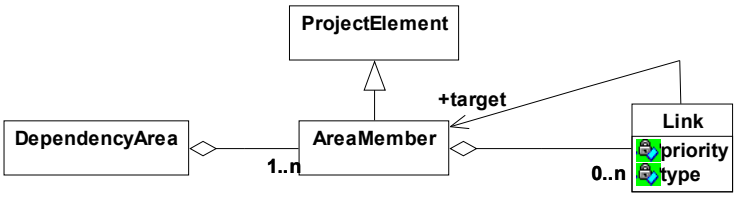


Fig. 5. The core of the Dependency Area metamodel

Traceability rules are specified using automata approach. Each rule is a separate, finite state automaton. A status of a rule execution is stored in the nodes of automata. Transitions between nodes are annotated with actions and conditions. An action can be performed in a given state only if the appropriate condition is satisfied. Actions and conditions are specified in a scripting language. In the implementation we used JavaScript language. The detailed description of the syntax of traceability rules is beyond the scope of this paper [7].

The automata approach is very flexible allowing specification of different transformation rules, including different traceability policies. On the other hand, the transformation can be realized by a simple rule executor, because the entire logic is stored in the rules. The rule executor executes a set of rules for any project element that was assigned to a dependency area. All rules are ordered according to their priorities and executed in the defined order. If two or more rules have the same priority the order of their execution is random. Elements assigned to a dependency area are considered by the executor in the order of their addition. The first one is the initial element indicated in the input project.

Before executing a single rule, its precondition is checked. If it is satisfied, the initial node of the rule is considered. In case a node has more than one outgoing transitions, they are ordered according to edge priorities. If a condition of a selected transition is satisfied, the transition is followed and its action performed. All nodes of the rule accessible from its initial node are visited during the rule execution.

In the result of the transformation process, the whole set of rules is executed for any project element assigned to the dependency area. Any element is added only once to the resulting dependency area. If an action specifies assignment of an already

existing element, only additional references between elements are added in the output dependency model.

4.2 Transformation Process of Input Models

According to the assumptions of the framework, an input model can be specified in any form that can be transformed to an internal form of the Project metamodel. The transformation of the input model can be realized by an input plug-in. The idea will be explained on an example of a subset of UML.

General approach corresponds to that shown in Fig. 2. Transformation rules are defined as a set of declarations satisfying the requirements of the QVT specification. The set of rules describes mapping of elements of a project defined according to one metamodel, to elements of a project from another metamodel. Transformation rules from UML to Project are straightforward. Figure 6 illustrates the idea of exemplary rules. On the left hand side input elements of UML are given. The set of rules takes these elements as their inputs. Results of the rules are instances of classes *ProjectElement* and *LinkAnnotation* shown on the right hand side. Theses instances are connected with appropriate references. The detailed rule description is specified in the XML language [7].

The schema of transformation process of the input model is shown in Fig. 7. It consists of the following steps:

1) A metamodel of the Project is prepared using a CASE tool and exported in the XMI format.

2) This metamodel is transformed from the UML format to the standard MOF format.

3) A metamodel of UML in the MOF format is delivered.

4) Metamodels of UML and Project are converted from the XMI to JMI standard.

5) Transformation rules are specified in the XML form accepted by the transformation tool.

6) A UML model, to be analyzed, is prepared in a CASE tool and exported to the XMI format.

7) The UML model is converted from the XMI format to the JMI standard.

8) The input UML model is transformed to its corresponding Project model, using UML and Project metamodels, and the appropriate transformation rules.

It should be noted that steps from 1 to 5 should be performed only once for a given input model notation. They are the tasks of a plug-in developer. Only steps 6-8 are performed by a model designer each time a new model is analyzed. Step 6 is realized manually - it is a proper design activity, whereas steps 7 and 8 are completed automatically using the previously prepared plug-in.

A result of the transformation process is the input model described as a Project model in the XMI format, suitable for the traceability analysis.

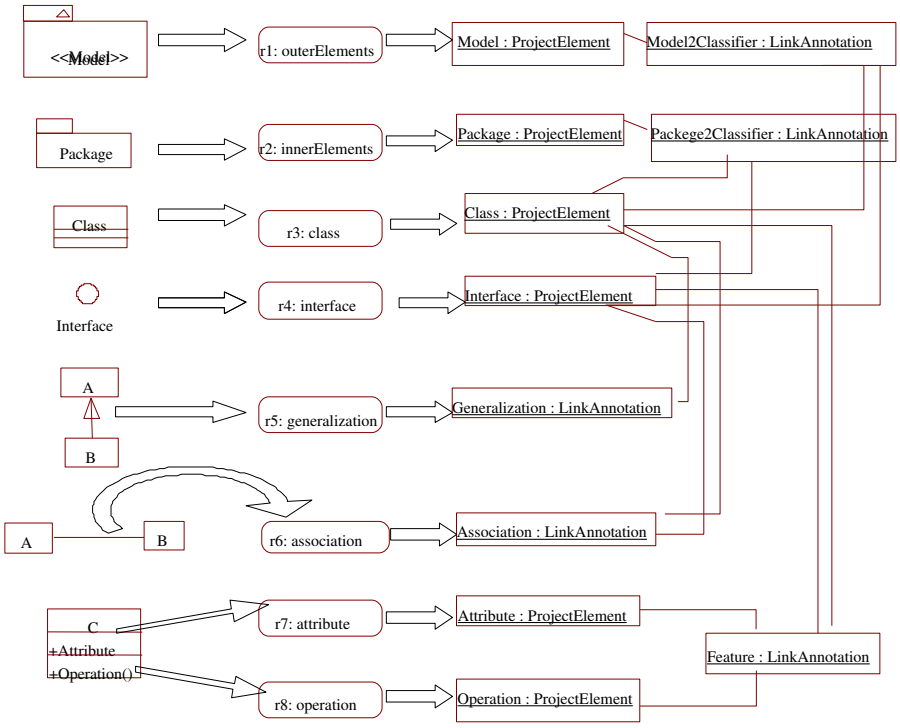


Fig. 6. Examples of transformation rules from a UML model to Project elements

In an exemplary implementation of the input plug-in we used following tools supporting the transformation process:

- UML2MOF - a tool for model conversion from UML to MOF,
- MDR [15] - a library implementing the JMI standard [16] that describes generation of interfaces for accessing model elements based on the metamodels given in the XMI,
- MdaTranfs [17] - a tool for model transformation based on the QVT approach.

The output transformation process has a similar structure as that shown for the input process (Fig. 7). Three types of input data should be prepared for a given output notation. If the notation of the output result is UML, three types of input data are metamodel of Dependency Area, metamodel of UML and output transformation rules (in XML). A dependency area generated by the traceability analyzer is the fourth input of the process. Using the same tools as shown for the input transformation process, we can automatically obtain the dependency area as a UML model. Once, having prepared the appropriate metamodels and transformation rules, the whole output transformation process is transparent to a user. Although a plug-in developer can modify the metamodels and/or transformation rules and, therefore, adjust the process on demand.

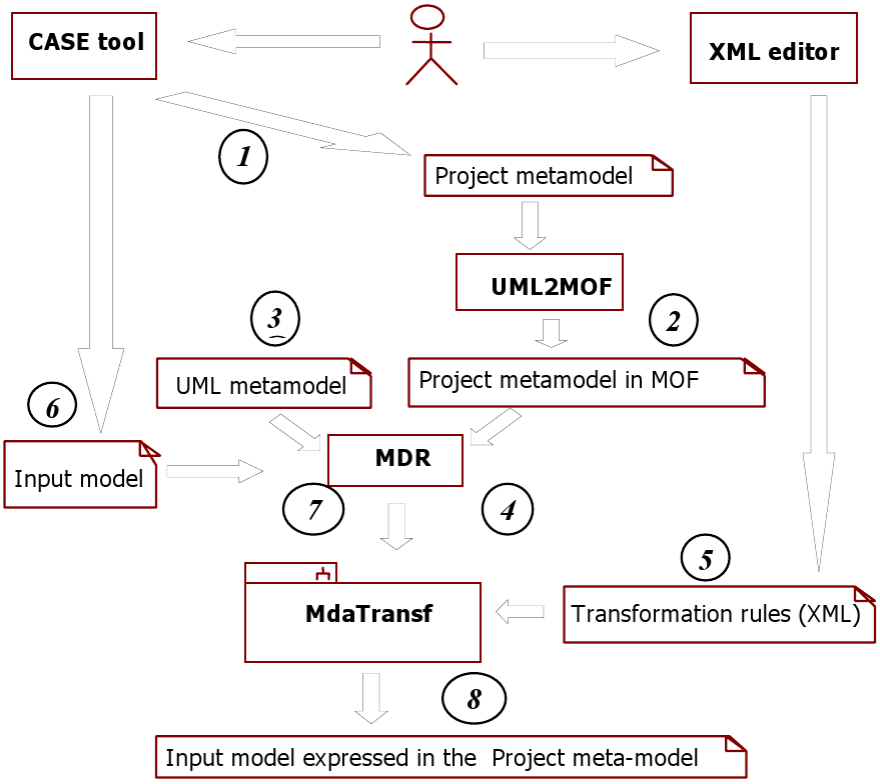


Fig. 7. Realization of the input transformation process

5 Final Remarks

We have showed how model transformation technology was used in the generic framework. The framework supports traceability in object-oriented designs. Model transformations assisted to attain the important advantages of the framework i.e., its flexibility, independence of the notation of an input model, separation of the traceability logic from the execution engine, and possibility of the simple evolution of the traceability strategy. However, in order to use this technology, it was necessary to provide specifications of the appropriate metamodels and prepare definitions of required transformation rules. In both cases of languages used for the transformation descriptions, the QVT and the automata-based approach, preparation of the transformation rules was very laborious. It was the price of the achieved universal solution.

Further development of the framework should deal with plug-ins for other UML subsets or specialized meta-models. Evaluation of different traceability strategies will

require more experiments with different sets of traceability rules. In the further evolution of the framework we can benefit from the model transformations applied in it.

Acknowledgment This work was supported by the Dean of the Department of Electronics and Information Technology, Warsaw University of Technology under grant no 503/G/1032/4000/000.

References

1. Object Management Group, *MDA Guide, Ver. 1.0.1,* OMG Document omg/2003-06-01 (2003).
2. Frankel D. S.: *Model Driven Architecture: Appling MDA to enterprise computing,* Wiley Press, Hoboken, NJ (2003).
3. Object Management Group, *MOF QVT Final Adopted Specification,* OMG Specifications ptc/05-11-01 (2005).
4. Czarnecki K., Helsen S.: *Feature-based survey of model transformation approaches,* IBM System Journal, Vol. 45, No 3 (206) 621-645.
5. Buttner F., Bauerdic H.: *Realizing UML model transformations with USE,* UML/MoDels Workshop on OCL (2006) 96-110.
6. Derezińska A., Zawłocki J.: *Generic framework for automatic traceability in object-oriented designs,* Annals of Gdańsk University of Technology Faculty of ETI, No 5 (2007) 291-298 (in polish).
7. Zawłocki J.: *A framework for traceability process in object-oriented models,* Master Thesis. Inst. of Comp. Science, Warsaw Univ. of Technology (2006) (in polish).
8. Letelier P.: *A Framework for Requirements Traceability in UML-based Projects,* In: Proc. 1st Int. Workshop on Traceability in Emerging Forms of Soft. Eng. by IEEE Conf. on ASE, Sept. 28, Edinburg, (2002).
9. Egyed A.: *Consistent Adaptation and Evolution of Class Diagrams during Refinement,* In: Proc. 7th Inter. Conf. on Fundamental Approaches to Software Engineering (FASE), Barcelona, Spain, March (2004) 37-53.
10. Spanoudakis G. et al.: *Rule-based Generation of Requirements Traceability Relations,* In: J. Systems and Software, 72(2), (2004) 105–127.
11. Derezińska A., Bluemke I.: *A Framework for Identification of Dependency Areas in UML Designs,* W Proc. of Conf. on Software Engineering and Application, SEA'05, IASTED, Phoenix, Arizona, USA, Nov. 14-16, Ed.: W.-T. Tsai and M.H. Hamza, Acta Press (2005) 177-182.
12. Derezińska A.: *Specification of Dependency Areas in UML Designs,* In: Annales UMCS Informatica AI 4, Vol. 4 (2006) 72-85.
13. Arujo J., Whittle J., Toval A. and France R. R.: *Integration and transformation of UML models,* In: ECCOP 2002 Workshops, Eds.: J. Hernandez, A. Moreira, LNCS Vol. 254, Springer Berlin Heidelberg (2002) 184-191.
14. Object Management Group, *MOF 2.0/XMI Mapping Specification,* Ver. 2.1, OMG Document formal/05-09-01 (2005).
15. MDR library, `http://mdr.netbeans.org/`
16 . JMI standard, `http://java.sun.com/products/jmi/`
17. West Team, MDA - Transf User Guide, `http://www.lifl.fr/west/modtransf/`

Proceedings of the International Multiconference on Computer Science and Information Technology pp. 725–734
ISSN 1896-7094
© 2007 PIPS

Performance of Index trees on Flash Memory

Hyun Seung Lee, Ha Yoon Song, and Kyung-Chang Kim

Department of Computer Engineering, Hongik University, Republic of Korea
lhseung@oodb.cs.hongik.ac.kr hayoon@wow.hongik.ac.kr
kckim@wow.hongik.ac.kr

Abstract. Flash memory can be a viable solution for the future embedded systems. Embedded systems usually carry database as a part of embedded software. It is well known that flash memory is far faster than usual hard disk storages especially for read however it takes much time for flash memory to delete than to read or to write. For a database system, index tree on flash memory has been widely studied for its better performance, but there have been no standard measure for flash memory database performance. In order to find the storage access performance of index trees, we measured index tree access time.
We choose B-tree, R-Tree and MR-tree for prominent index trees. We also choose hard disk and flash memory for storage devices. With several indexing profiles having different mixture of insert, search (read) and delete of indices, we measured access time performance with different combination of index tree structure and storage device. We hope our result will work as a basis for flash memory database system design and performance trimming.

1 Introduction

Nowadays ubiquitous computing environment with mobile devices, consumer electronics and so on widely adopts flash memory system with rapid development of memory technologies. Mobile phones, PDA, smart card, digital cameras are rapidly consumed by consumers and almost of them are using flash memory with nice characteristics such as low power, non-volatility, high performance, physical stability, portability, etc. Flash memory has been discussed as a future replacement of hard disk storages. And the memory characteristics of flash memory are still under road to be developed.

Usual Flash Memory (FM) is consisted of several blocks with 32 sectors, one sector of 512 byte data area and 16 byte spare area. Different from Hard Disk Drives (HDD), the read access to FM is comparable to that of Dynamic RAM (DRAM) however it has very slowly write access and has no ability of overwrite. Overwriting can be done by writing after block delete. In order to solve these problems, there are FTL (Flash Translation Layer) between file systems and FM. Delete operations can be translated by mapping of logical sector address to physical memory address which has been already deleted. Such sort of FTL operations can help the use of FM with usual file system. However, most of FTL are also under for future development with huge options.

In case of constructing database system with FM, index will be stored on FM while actual records will be stored on HDD. With rapid increase of FM capacity, actual records will be also ported on FM in the near future. The use of FM as index storage will be naturally increase DB access performance, but it totally depends on the type of index trees and the access profile as well.

We can discuss three types of index trees. B-tree is a data structure with efficient DB indexing. R-tree is for spatial data access index with multimedia DB. Also, we can consider MR-tree as an upgraded version of R-tree. These index trees must be updated with Insert, Delete, or internally by Split operations. Once we implement HDD based B-tree on FM, it may cause performance degradation. Most of past researches concentrated on delete operations and improved with buffering such as BFTL [5] or BOF [11]. But both of improvements lead in overhead of read operation and management overhead of buffers with data.

In addition, the limited number of FM writes will cause index update be evenly distributed over blocks in order to fully utilize FM life cycle.

Under those situations, we are going to measure the access time of various index trees on FM. With three types of index trees with several access profiles, we will compare performance of each index tree on FM and HDD. We believe this measurement will inform researchers a key to improve their database systems on flash memory.

Following sections are consisted as follows. In section 2 we will discuss flash memory, index trees and related works in detail. The next section contains our result of measurement with detailed database access profiles design. We will conclude this paper in the final section.

2 Background and Related Works

We will discuss about the characteristics of FM as well as index trees in this section. From the characteristics of FM, the index trees on it will have different nature of operations and new mechanism for tree indexing will be studied further.

2.1 Flash Memory

We focus on the NAND FM since it has better characteristics than NOR FM. The NAND flash is one of the prominent FM currently under commercial manufacturing. The access characteristics of NAND flash memory is as follows [3]:

- 1.2 microseconds for 2 Byte read
- 35.9 microseconds for 512 Byte read
- 201 microseconds for 2 Byte write
- 226 microseconds for 512 Byte write
- 2 milliseconds for 16 KB block delete

NAND FM has operations of read and write on the unit of sector, and write operation must be done after deleting the corresponding block. Delete operations are done in the unit of block, thus it take far more time than read and write

operations which can be done in the unit of sector. Because of this difference between the unit of write and delete operations, write operations cannot be done immediately. Write-pending pages should be copied on other blank pages with modifications and this specific operations are called out-place update. Free page is a clean page never has been written. Dead page is a page having out-of-date data after out-places update. Live page is a page having up-to-date data after out-places update [9]. The number of dead pages will increase with cumulated write operations and the number of free pages will be degenerated. Therefore garbage collection will be required. We need block recycling policy to choose a specific dead page deleted and converted to free page. Also we need wear leveling in order to uniform selection of dead page to be deleted since there are limitations in the number of block delete. Thus it is clear that write operation requires extra cost with wear leveling and garbage collection.

Flash Translation Layer (FTL) helps file system regard FM as a logical space such as that of HDD. FTL mapping algorithms are classified as sector mapping, block mapping and hybrid mapping. Also, the logical-physical address translation is classified as in place and out of place where logical sector first for that while physical sector first for this. For example, FAST(Full Associate Sector Translation) [10], which improved log block mechanism, logically partites FM memory space into data block, log block, and sequential write block. It marks "in-place" on a data block for a sector written with hybrid mapping. Once an overwritten takes place for this sector, it choose a specific sector on log block and marks it as "out-of-place", except the zero-th sector be overwritten. Instead, a sequential write block is chosen and then marked as "in-place". Once a zero-th sector is overwritten again, a merge operation happens or an exchange operation happens in order to minimize delete operation. There exists several log blocks where there could be FIFO merge operation if there are no free sector on a log block. FAST supports mapping between file system and flash memory efficiently and file system can regard FM as block device such as hard disk drive.

2.2 Index Trees

B-tree [4] is an index most representatively used in database management system that is based on disk and main memory. Insert, delete, search of data are done easily and it supplies efficient balance algorithm.

R-tree [1] is guided hierarchic data structure from B-tree structure for spatial data access such as multimedia data. Each node is consisted of smallest d-dimension quadrilateral including own child node. Leaf node includes pointer for actuality geometrical object instead of including child of node in database. Objects are expressed by minimum sorted quadrilateral that they are included. Because R-tree is height balanced tree, it announces node split to parent node, so about average 70% of entries have filled in leaf node. R-tree does to increase often height of tree for height balance of leaf nodes, and this causes unnecessary cache miss occurrence finally.

MR-tree [7] uses basic index structure of R-tree, but height between sub trees can be unbalanced and the difference is reconstructed by maximum below 1.

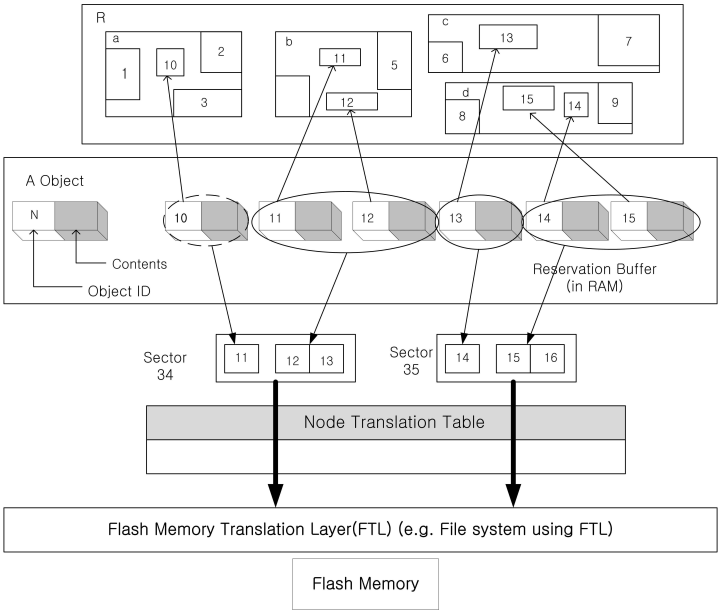


Fig. 1. R-tree on Flash Memory

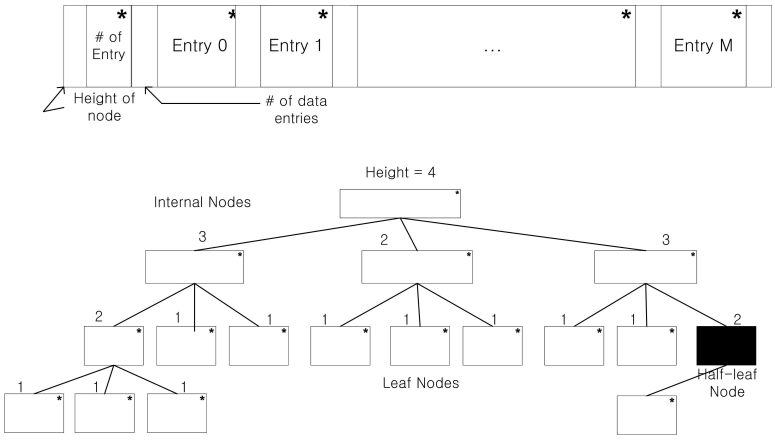


Fig. 2. Structure of MR-tree

If height between sub trees is restricted more than 1, it runs height balance algorithm. And, it has special feature that when split is produced, splits are delivered to parent node. In case of a node exists which has empty entry on insert operation and it judges re-insertion possibility electively in occasion of delete. MR-tree structure is shown in figure 2.

Index nodes of MR-tree are classified to internal node, leaf node and half-leaf node. Half-leaf node means nodes that have all entries for leaf nodes and data object. If new object is inserted in the half-leaf node continually, this node is changed to an internal node that has height of 2. Node structure of MR-tree other than R-tree shows two distinguished additional field. One field has a value that adds 1 to the value of the biggest heights of child nodes to field that display height of node, and is used to search the imbalance state of MR-tree height. Data entry number field used by half-leaf node and stores number of data entry stored in the node. Split of leaf node is delivered to parent node only if there is empty entry in internal node on insert operation. Therefore, it can certainly reduce whole number of node splits. MR-tree shortens tree height and indicate good performance in search by decreasing internal node MBR's size, improves entry space utilization ratio for internal node by delivering node split to parent node when more than one empty entry among internal nodes on insert operation, controls increase of tree height maximum, and reduces the count of cache miss.

2.3 Previous Works

When applying existing B-tree or R-tree to flash memory as it is, in worst case, we must modify one leaf node for every insert before split occurs. Then, write operation to flash memory space must require the same size as a leaf node. If leaf node is consisted of m records, write operation for the space of relevant leaf node is about m insertions in order to compose leaf node. And if modification of index pointer is required, all nodes inevitability is modified. Also data stored in the same area to a leaf node must be stored in another block m times with leaf node space because of out-place update characteristics of FTL. If node space occupies big space, free page is consumed rapidly and garbage collection and wear-leveling are done frequently. This causes each operation costs expensively and spoiled FM lifetime.

Resent researches are concentrated on these phenomena but only a few results have been reported. Most of them are specialized on B-tree. We have one prominent example to R-tree. In order to reduce write operation, Wu proposed BFTL and FTL applying to R-tree [6][5]. It provides a layer for tree index nodes with translation table and reservation buffer is also included. FTL applied to R-tree structure is shown in figure 1. Multiple insertions of leaf nodes will be translated to one write operations to FM blocks effectively by Wu's mechanism.

3 Access Time Measure

In this section, we will describe the performance measure approaches and will show performance comparison results. We choose storage device as FM and

HDD. HDD characteristics are already fully studied by researchers and therefore discussions are omitted. It is natural that FM performance always outperforms HDD performance and that is the reason why current technology rapidly adapts FM as a storage part of mobile or embedded (or both) systems. Even though it is natural in performance comparison, we have interested in the quantitative performance between two storage devices. In addition, the database access profile clearly controls the index tree performance on two different storage types. Considering FM has very low access time requirement of read operation that matches search operation of index trees while it has the higher cost of write even with wear leveling and out place update. And with the delete operation of the highest time requirement, we cannot clearly identify the performance of indexing on FM. Thus we conduct a simulation with various index profiles of database access on two different types of storages.

3.1 Database Access Profiles Design

It is critical that the delete operations must be a governing parameter of index timings. We can defer delete operations with the help of FTL however they can't be omitted and thus must be done in the long run.

Therefore we designed three profiles with different characteristics respectively. Each profile has more than ten thousand operations on index trees with different patterns of access sequence and frequency. The first one is random profile (R). We generated totally random profile with totally random distributions of insert, delete and search operations. Of course, split operations are translated into related sequence of operations. The second one is insert oriented profile (I). We designed number of operations of insert has ten times higher than the number of delete operations. We considered a very first stage of database construction with one thousand insert operations and one hundred delete operations. The final one is delete oriented profile (D). Almost 2000 insertions made and 2000 delete operations are then came by search operations. With these three access profiles we executed timing based simulation.

In addition, the timings include actual CPU time of tree operations on Windows XP PC with 2.40GHz Intel Pentium CPU and 1Gbyte of DRAM. Each flash memory timing is show in section 2 and Each HDD timing is actually measured by real indexing operations on the PC and resulted in 12.4ms per 512byte sector average. We put tree nodes in 512Byte block of HDD and FM both for fair comparison. For the results on following figures, the X-axis stands for the number of indexing operations and the Y-axis is the accumulated time of each indexing operation in microseconds.

3.2 Performance Comparisons

Basically B-tree, R-tree and MR-trees are simulated with three different index profiles on FM and HDD respectively. Figure 3 shows a result on random profile with three different trees on FM and HDD. B-tree with small node size clearly outperforms other trees since B-tree requires small node size and needs less

number of splits on 512byte sector size. In addition results on FM drastically outperform HDD as expected.

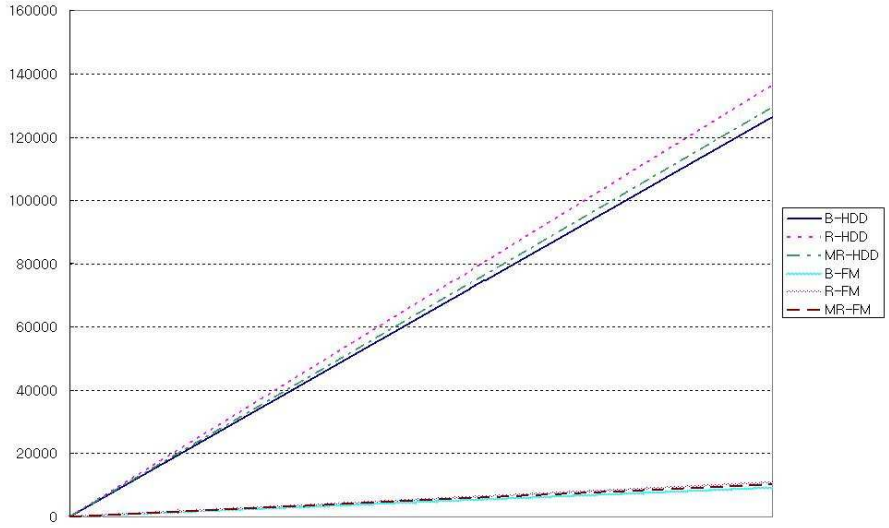


Fig. 3. Indexing timings over random access profile (R)

Figure 4 shows a result on insert oriented profile with three different trees on FM and HDD. It does not show any distinguished difference from figure 3 from the aspect of HDD but concentration on insert operation shows two distinguished patterns on FM (small graph on upper-left corner). Insert shows abrupt increase of cumulated times at the first stage and then mixture of delete and search lessens the slope.

Figure 5 shows the effect of delete operations. Inserts on front end of the graph show increase of operation timings while deletes on backend show shaper increase than insert. However, index timings on FM with huge number of deletes still outperform index timings on HDD. It is clear that the number of delete cannot exceed the number of insert, by which the effect of delete is strictly bounded.

Then we focused on index tree timings on FM. Every indexing profiles are then compared on FM. Figure 6 shows the comparison. Random profile with uniform distribution of delete, deleted oriented profile, and insert oriented profiles show timing results as expected. The three bottommost results of insert oriented profiles shows similar pattern of index timings, and B-tree with small node size outperforms others with less number of node splits on a FM block while MR-tree considering cache effect outperforms R-tree with the same node size on a block.

As a result, we can conclude that FM with any index tree outperforms HDD storage, and then MR-tree for multimedia database always outperforms R-tree.

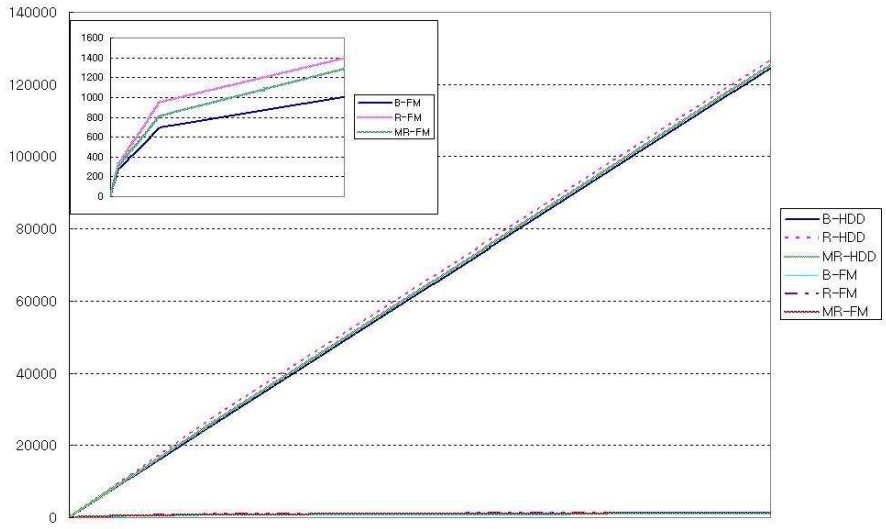


Fig. 4. Indexing timings over insert oriented profile with small number of delete (I)

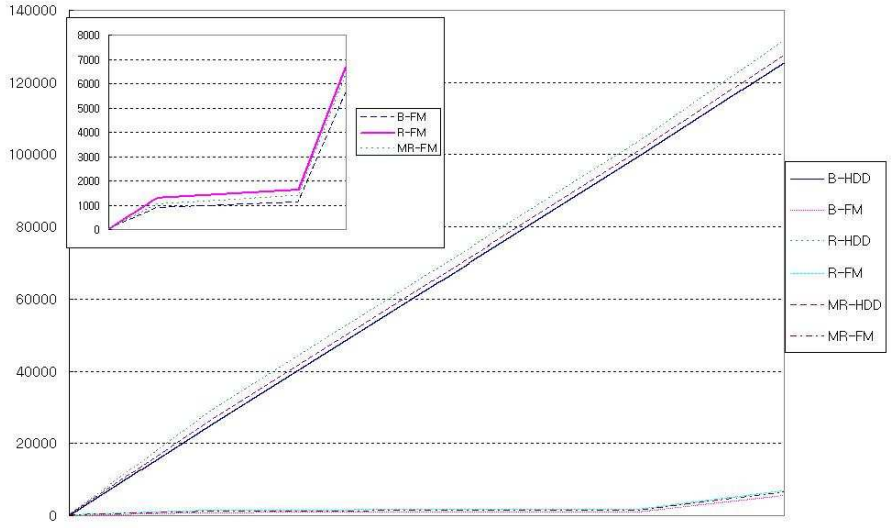


Fig. 5. Indexing timings over delete oriented profile with large number of delete (D)

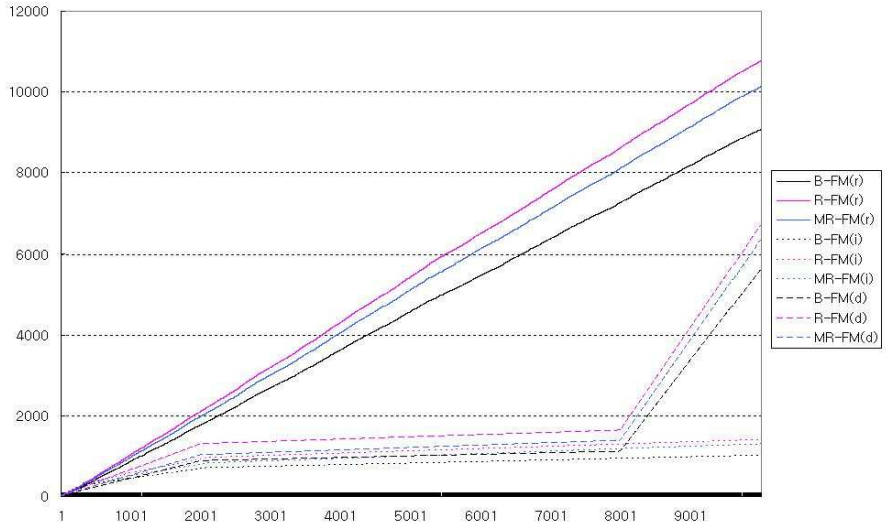


Fig. 6. Indexing timings on flash memory

We could recommend the use of FM as index tree storage in any cases. With normal usage of DB on FM, for example insert oriented profiles, index trees on FM shows satisfactory performance. Considering the up-to-date flash memories have better timing characteristics [12] than we used in section 2, we can expect at least the same performance with actual current DB applications.

4 Conclusions and Future Works

We focused on performance measure of various index trees on flash memory system and hard disk drive system. It is clearly a trend that flash memory will be a prominent solution for future embedded mobile systems. We draw a natural solution that index trees on flash memory always outperform hard disk system. And we found that indexing on MR-tree outperforms indexing on R-tree for its cache based operation when flash memory blocks and sectors acts like a cache block. Other interesting results show that B-tree performance outperforms other indexing trees because a small node size of B-tree reduces the number of splits inside a sector. However B-tree is not a best fit for multimedia database indexing therefore must be considered in another aspect. In the aspect of multimedia, MR-tree outperforms R-tree because of its considerations of cache effect. We hope our results could provide a guideline for performance measurement and trimming for existing database with flash memory as well as hard disk system.

Our future research will be concentrated in two categories. First, we will modify the cache block size of MR- tree to that of FM block size and harmonize them. We expect the cache usage of MR-tree will be combined with DRAM like

read time of FM so that the performance of indexing will be improved. Second, we will introduce on demand paging [8] especially for indexing trees as well as wear leveling. With the combination of indexing tree operations and on demand paging, we consider a higher performance with long lifetime of flash memory.

Acknowledgments. The authors give special thanks to Mr. KwangJo Hwang for his help to complete this english paper.

References

1. Antomn Guttman. R-trees—a dynamic index structure for spatial searching. In *International conference on management of data proceedings of the 1984 ACM SIGMOD,* pages 47–57, 1984.
2. Michael Wu and Willy Zwaenpoel. eNVy: a nonvolatile main memory storage system. In *proceedings, Fourth workshop on workstation operating systems(WWOS-III)*, 1993.
3. Jesung Kim, Jong-Min Kim, Noh S. H, Sang-Lyul Min and Yookun Cho. A space-efficient flash translation layer for compactflash systems. In *IEEE Transactions on Consumer Electronics*, pages 366–375, 2002.
4. Beng chin Ooi and Kian-lee Tan. B-trees: bearing fruits of all kinds. In *Australian Computer Science Communications, Proceedings of the 13th Australasian database conference (ADC02),* 2002.
5. Chin-Hsien Wu, Li-Pin Chang, and Tei-Wei Kuo. An efficient B-tree layer for flash memory storage systems. *The 9th, International conference on Real-Time and Embedded Computing systems and Applications.(RTCSA)*, 2003.
6. Chin-Hsien Wu, Li-Pin Chang, and Tei-Wei Kuo. An efficient R-tree implementation over flash-memory storage systems. In *Proceeding of the 11th ACM international symposium on Advances in geographic information systems*, pages 17–24, 2003.
7. Kyung-chang Kim and Suk-Woo Yun. MR-Tree: a cache-conscious main memory spatial index structure for mobile GIS. In *Web and wireless geographic information systems, 4th international workshop (W2GIS 2004)*, pages 167–180, 2004.
8. Chanik Park, Jeong-Uk Kang, Seon-Yeong Park, and Jin-Soo Kim Energy aware demand paging on Nand Flash-based embedded storages. In *Proceedings of IEEE/ACM international symposium on Low Power Electronics and Design(ISLPED 2004)*, 2004.
9. Eran Gal and Sivan Toledo. Algorithms and data structures for flash memories. In *ACM Computing Surveys(CSUR)*, pages 138–163, 2005.
10. Sang-won Lee, Dong-Joo Park, Tae-Sun Chung, Dong-Ho Lee, Sang-Won Park, and Ha-Joo Song. A log buffer based Flash Translation Layer using fully associative sector Translation. *ACM transactions on Embedded Computing systems*, 2005.
11. Jung-hyeon Nam and Dong-joo Park. The efficient design and implementation of the B-tree on flash memory. In *Korea information science society*, 2005.
12. SAMSUNG NAND flash SLC-small block. In `"http://www.samsung.com/global/business/semiconductor /productInfo.do?fmly_id=158&partnum=K9F1208R0C&&ppmi=1157"`, 2007.

Proceedings of the International Multiconference on
Computer Science and Information Technology, pp. 735 – 744
ISSN 1896-7094
© 2007 PIPS

Monitoring services in Service Oriented Architecture[1]

Ilona Bluemke, Marcin Warda

Institute of Computer Science, Warsaw University of Technology,
Nowowiejska 15/19, 00-665 Warsaw, Poland
{I.Bluemke}@ii.pw.edu.pl

Abstract. At the Institute of Computer Science Warsaw University of Technology a tool for monitoring services in Service Oriented Architectures (SOA) was designed and implemented. This tool was used in some experiments in real enterprise integrated architecture eg.: the effectiveness and the usage of resources in some flows were measured. These experiments are described and some conclusions are given. The monitoring module may be very useful in the maintenance of complex SOA systems.

1 Introduction

The definition of Service Oriented Architecture - SOA [1] is following: "An architectural style whose goal is to achieve loose coupling among interacting software agents. A service is a unit of work done by a service provider to achieve desired end results for a service consumer. Both provider and consumer are roles played by software agents on behalf of their owners." The real power of SOA and Web services becomes apparent when various constituents are added, removed, replaced, or upgraded without adversely impacting the whole system. SOA- and Web services-based solutions allow products from different vendors, running on different platforms, to work together. Despite attempts to standardize, a typical IT infrastructure is heterogeneous and it makes the management a very complex task. The problems of management in SOA architectures is discussed in [2].

With the development of the Service Oriented Architecture (SOA), organizations are able to compose complex applications from distributed services supported by third party providers. Under this scenario, large data centers provide services to many customers by sharing available IT resources. This leads to the efficient use of resources and the reduction of operating costs. Service providers and their customers often negotiate utility based Service Level Agreements (SLAs) to determine costs and penalties based on the achieved performance levels.

Service-based approaches are widely used to integrate heterogenous systems. Web services allow for the definition of highly dynamic systems where components (services) can be discovered and quality of service (QoS) parameters negotiated at run-time. This justifies the need for monitoring service at run-time.

At the Institute of Computer Science Warsaw University of Technology a tool for monitoring services in Service Oriented Architectures (SOA) was designed and im-

1 This work was supported by the Dean of the Department of Electronics and Information Technology Warsaw University of Technology under grant no 503/G/1032/4000/000

plemented. This tool was used in some experiments in real enterprise integrated architecture. These experiments are described and some conclusions are given. The monitoring module differs significantly from existing monitoring modules. Our program monitors services on the ESB (Enterprise Service Bus) level even, if the services are executed on different servers. Other programs are able to monitor business processes eg. Optimize for Process in webMethods or do not provide sufficient information if the execution is distributed on several servers. The monitoring module may be very useful in the maintenance of complex SOA systems.

The remainder of the paper is organized as follows. Section 2 describes some literature approaches. Section 3 introduces the overall system architecture. The experiments with the monitoring program are presented in Section 4. Section 5 contains some conclusions.

2 Related Work

In [3] is presented how to monitor dynamic service compositions with respect to contracts expressed via assertions on services. Dynamic compositions are represented as BPEL (Business Process Execution Language) processes which can be monitored at run-time to check whether individual services comply with their contracts. Monitors can be automatically defined as additional services and linked to the service composition. Two alternative implementations of the monitoring approach: one based on late-binding and reflection and the other based on a standard assertion system are described.

The paper [4] proposes a framework for monitoring the compliance of systems composed of web-services with requirements set for them. This framework assumes systems composed of web-services that are coordinated by a service composition process expressed in BPEL4WS (Business Process Execution Language for Web Services) and uses event calculus to specify the properties to be monitored. The monitorable properties may include behavioural properties of a system which are automatically extracted from the specification of its composition process in BPEL4WS and/or assumptions that system providers can specify in terms of events extracted from this specification.

In [5] the quality model suitable for capturing and reasoning about quality aspects of multichannel information systems is presented. In particular, the model enables a clear separation of modeling aspects of services, networks, and devices. Further, it embeds rules enabling the evaluation of end-to-end quality, which can be used to select services according to the actual quality perceived by users. In [5] a long list of references to papers concerning different aspects of quality of services can also be found.

3 Monitoring module

However there are some monitoring programs (some of them mentioned in section 2), we decided to design and implement a monitoring module for integration product webMethods [8]. Existing monitoring modules are able to monitor business processes eg. Optimize for Process in webMethods. The implemented module is able to moni-

tor services on the ESB (Enterprise Service Bus) level even, if the services are executed on different servers. Other programs monitoring ESB, are dedicated to only one server or do not provide sufficient information if the execution is distributed on several servers. Our goal was to monitor ESB services executed on distributed servers and to provide full context of the flow. Such information may be very useful for administrators of integrated systems.

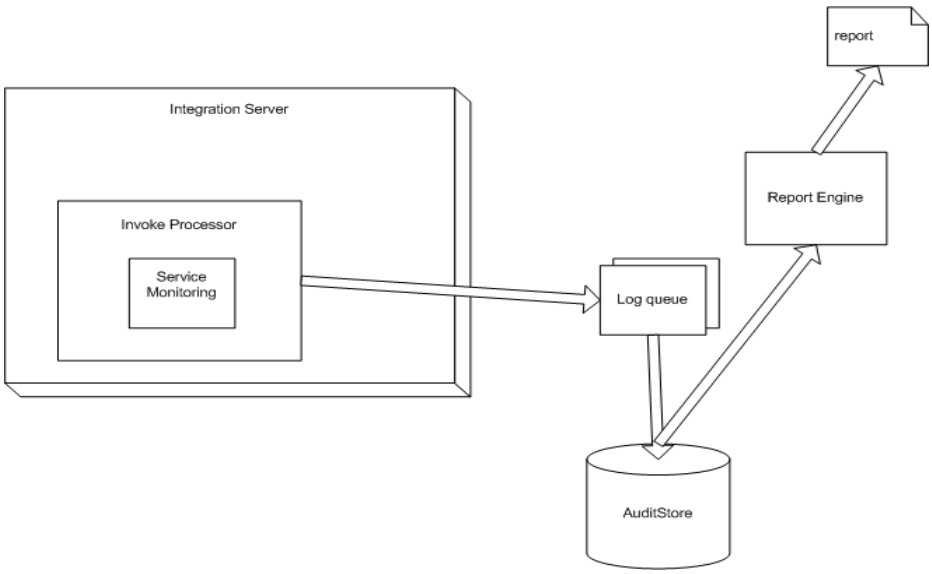


Fig. **1**. The architecture of monitoring module

The module for monitoring services on an integration platform contains processes responsible for:

- collecting information concerning service calls based on a „pre/post invoke" and storing it in memory of a local server (log queue),
- elaborating the log queue: store logs in a file on local server, write them to the data base (using connection of the local server or send them to the central service which will write logs to the data base),
- evaluating logs in data base: statistics calculation, deleting ancient logs.

The basics ideas of a monitoring module are presented in Fig.1. The implementation details of this module are described in [6,7].

4 Experiments

The implemented monitoring module was used in several experiments in real SOA environment. The integration platform was built with 6.1.5. webMethods product [8] and contained:

- webMethods Integration Server ,
- webMethods Broker,
- WmJdbcAdapter implementing JDBC protocol and enabling the connections with data bases in the integrated environment.

The environment contained five logical layers:

1. core-tier – abstract business services layer,
2. adapter-tier - contains adapting services enabling the connections between business services and physical resources of integrated systems (e.g. billing systems, client management system),
3. front-end-tier – covers adapter services provided for clients applications,
4. business-process-tier – layer providing business processes of the platform,
5. admin-tier – administrator server collects information about the availability of platform elements. This server was also used to write logs from monitoring module.

Admin server was using its own message broker. The integration environment had also configuration and operational data bases. The operational data base was also used to store logs from the monitoring module. In section 4.1 some technical details of the environment used for experiments are given. The results of experiments are presented in sections 4.2 - 4.5.

4.1 Environment for experiments

Each logical layer of integration servers had two instances of the server. Servers were working on 32-bit computers IBM Blade Server dual-core with 4GB RAM operation memory and operating system RedHat Enterprise Linux v.3. Integration Broker was located on Sun-Fire-V440 computer working under SunOS 5.9 operating system. Data base (engine Oracle 9i) was deployed at the same computer. Integrated systems were also using Oracle data bases (versions 8 or higher).

Calls of services from client were made with SOAP-RPC protocol, API webMethods implemented in Java or HTTP (GET) protocol used for direct transfers of xml files. Resources of integrated systems were accessible trough PL/SQL statements. In communication the JDBC adapter was used. The SOAP-RPC protocol was also used in synchronic internal calls between servers. In asynchronic calls, the message broker was used.

The experiments were made in an real environment of a mobile phone operator. The integrated platform consisted mainly of systems for clients and clients demands managements. The platform has approximately 300 services and 20 business processes. The analysis were made for selected, following flows:

1. `x.core.serivce:getServiceParamsValues` – getting parameters for client' services (synchronic)
2. `x.core.serivce:getAvailServices` – getting services available for the client (synchronic)
3. `x.core.service:getActiveServices` – getting active services for the client (synchronic)
4. `x.core.om.service:serviceModificationOrder` – putting an order for the modification for client' service (asynchronic)

5. `x.core.infoservice:infoservice` – checking resources available for the client in her/his electronic wallet (synchronic)
6. `x.core.contract:getContractData` – getting data from the client' contract (synchronic).

The logs were collected during one week of functional testing of the integrated platform. The functional testing was caused by new code release for the integrated systems. During testing the load of the system was significantly lower than normally (only 5% of the normal, working load). The flows during testing were not evenly distributed and this phenomena can be seen in the results of experiments. It was not possible, due to the owner constraints, to monitor normally working platform.

In next sections some analysis of the stored by monitoring module logs are shown. The results present possible application of the monitoring module. In a short paper not all application domains may be presented, more analysis are given in [6]. The performance analysis should not be made during the functional testing.

The measurements of a flow needs a lot of information e.g. user of the service, path of business services called during this flow, errors returned by service, time parameters, computers executing services etc. In an integrated environment there are usually many services, many flows, many users, etc. It can be easily imagined, that the analysis of stored by a monitor data is time consuming and very complex problem.

4.2 Analysis of the flow' results

Analysis of the data stored by the monitoring module makes possible to calculate the percentage of errors returned from the flows or even from a service being the part of the flow. The percentage of errors can be calculated for each user calling the flow. Based on this data the quality of the service (service level) can be calculated. Some errors returned from the flow do not mean that the service is incorrect e.g. in putting an order flow – not validated order. A dictionary, containing all errors which can be ignored in the flow, is necessary for the calculation of the service quality. In figure 2 the percentage of returned errors in some flows is presented. The flows are following:

1. `IVR. x.adapter.ivr:setServiceParams`
2. `IVR. x.adapter.ivr:getServiceParameters`
3. `IVR. x.adapter.ivr:infoService`
4. `SMS. x.adapter.sms:getSmsFromDb`
5. `ECARE_WWW.x.adapter.ecare:serviceModificationOrder`
6. `ECARE_WWW.x.adapter.ecare:getActiveService`
7. `ECARE_WWW.x.adapter.ecare: getAvailServices`
8. `ECARE_WWW.x.adapter.ecare:getContractData.`

On figure 2 some differences in the percentage of errors can be seen. The best results has flow `setServiceParams` (0.4%), the worst – `serviceModifcationOrder` (1.5%). The average availability was 98,5% - 99,5%. The information from this diagram can be used to find low quality flows. These flows are candidates to be improved. Other criteria for choosing a flow for the improvement process are: priority of service, complexity, number of calls of this service, cost. The diagram (fig.2) contains data collected during the whole testing period. Summarized data can be used to find trends. Similar data but gathered in short period of time, may help in finding system bottlenecks.

4.3 Analysis of the flow' effectiveness

The stored logs may produce the diagrams showing the effectiveness of flows. The basics criteria is the time to accomplish the flow. In figure 3 minimal, maximal and average time of several flows, listed in section 4.2 are presented.

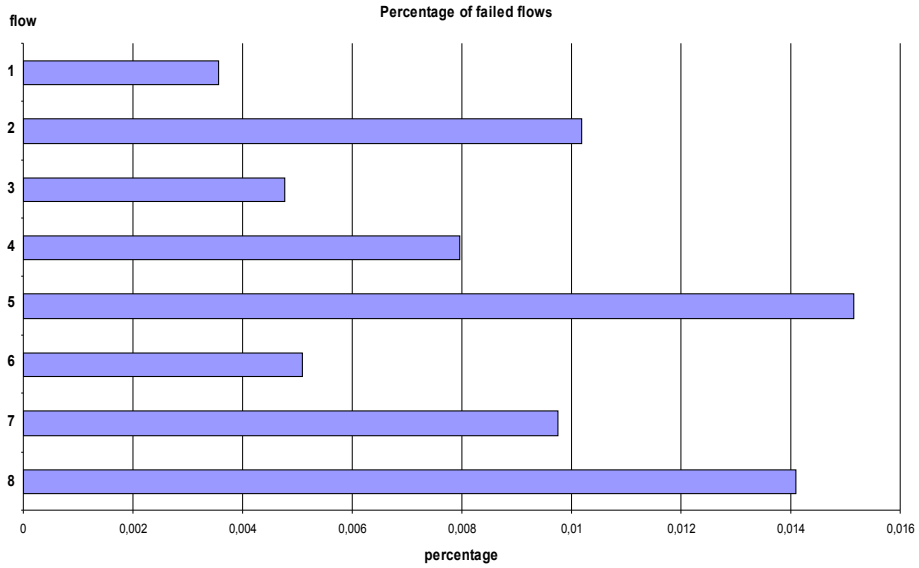


Fig. 2 Percentage of errors in flows

If the execution time is crucial for the client, the execution time of the flow can be the basis for the calculation of the service quality. The percentage of the execution times lower than the critical ones, should be compared with values in Service Level Agreement. In fig.3 SMS user calls are significantly slower than others. SMS calls are asynchronically realized because the execution time is not critical for user.

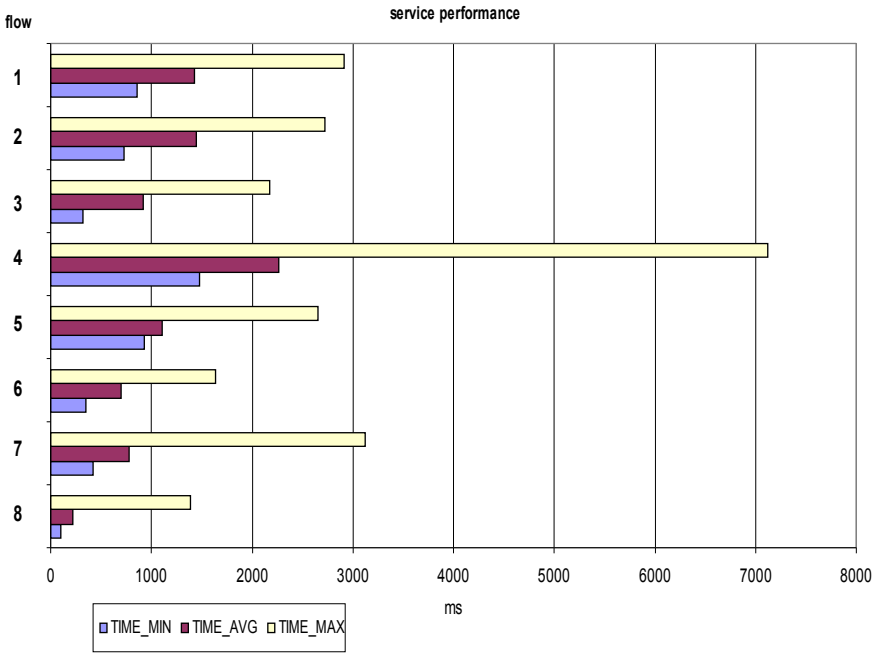


Fig. 3. Comparison of effectiveness of flows

4.4 Flows per time unit

The traffic in the integrated system (number of flows realized in a time unit) is very important for system administrator and manager. The flows in figure 4 are following:

1. `x.core.service:getServiceParamsValues`
2. `x.core.service:getAvailServices`
3. `x.core.service:getActiveServices`
4. `x.core.om.service:serviceModificationOrder`
5. `x.core.infoservice:infoservices`
6. `x.core.contract:getContractData`

In figure 4 the percentage of selected flows in total traffic are shown. Such information helps in estimation of the cost of maintaining the integrated system for a client.

In figure 5 the time distribution of these flows is shown. These information are very useful in the estimation of system capacity. System should be able to stand the maximal load (traffic). The data in figure 5 were obtained not during normal service of the systems but during test period, so the distribution of data is not typical.

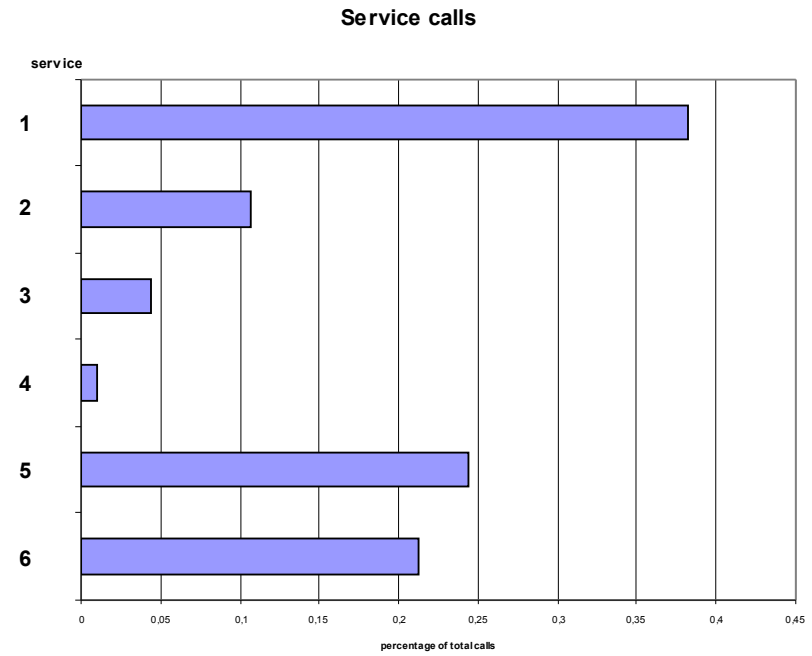


Fig. 4. The percentage of six flows in total traffic

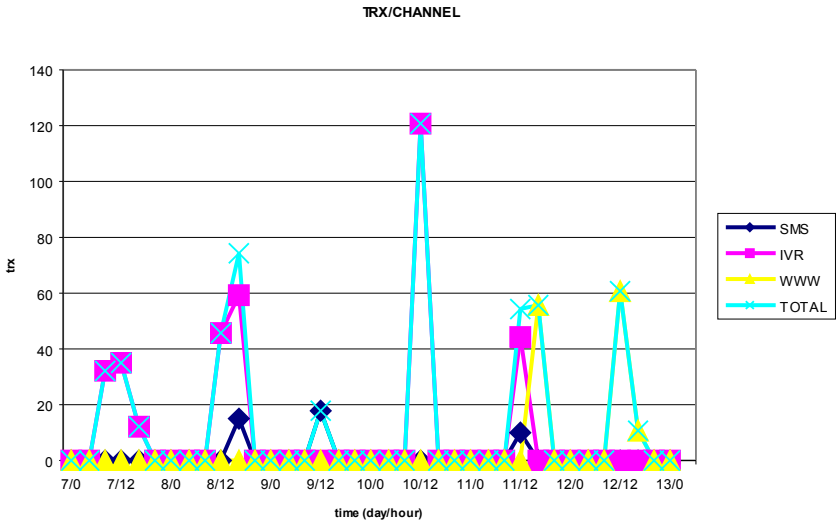


Fig. 5 Traffic in the integrated system in some time period

4.5 Analysis of the platform resources

The administrator of an integrated platform should be provided with information concerning the utilization of system resources. The information from section 4.4 is insufficient to calculate the number of servers and connections needed for correct systems operation. Important resources of an integrated platform are the total number of threads executing clients services and connections to external resources. Too high number of concurrently working threads will slow down the server or some services will be rejected. If the number of request to external data bases is higher than the number of connections, services will end with errors. To guarantee the correct platform operation the number of resources needed for the traffic should be estimated. For such estimations the number of flows and the time to execute them on a server is necessary.

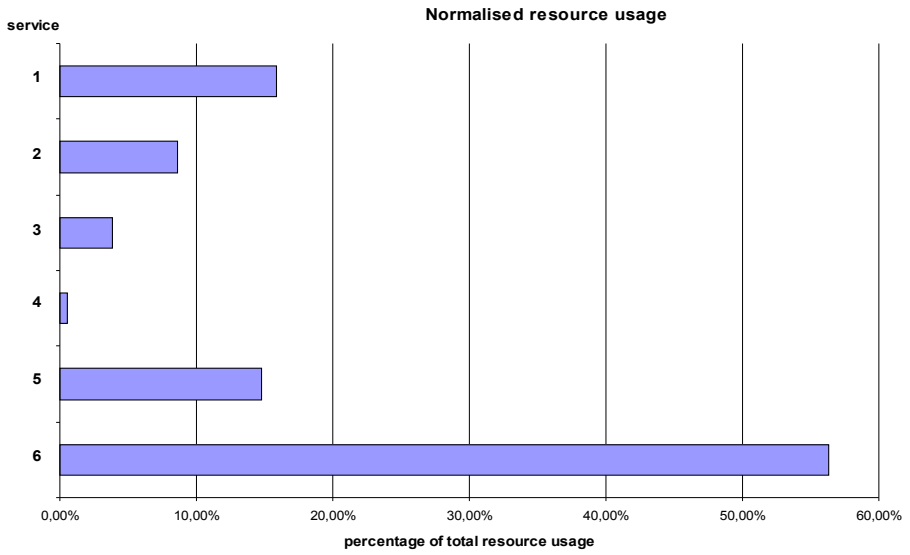


Fig. 6. The usage of server resources by selected services

In figure 6 the usage of server resources by selected six services is shown. The names of these flows are given in section 4.4. From this figure the service using maximal server resources can be identified. This service is a candidate for optimization. If the execution time of `x.core.contract:getContractData` service will be half the current value, 30% of platform resources will be set free. 30% of resources for a big platform is worth some optimizing actions. It may be effective to move some data used by this service to a fast, special data base.

5 Conclusions

The quality of services on an integrated platform is influence d by many factors: e.g. the physical and logical architectures, integrated systems, implementation of flows,

technology, etc. Finding bottlenecks or services of low quality is a very difficult task. The monitoring module implemented at the Institute of Computer Science Warsaw University of Technology [6,7] can be very helpful. This module is dedicated to webMethods [8] environment. The idea of monitoring services in SOA architectures, implemented in this tool, is general so the design of this tool can be easily implemented for others SOA integrating products. The data collected by monitoring module can be used by administrators and managers to improve the quality of service on the integrated platform. Examples of such analysis are given in sections 4.2 – 4.5. The monitoring module may be also used to detect bottlenecks of the integrated systems. Such measurements should be performed under normal, typical load of the system (not during functional testing as described in this paper).

The module is able to monitor services on the ESB (Enterprise Service Bus) level even, if the services are executed on different servers. The whole context of the flow is provided to the user. Such information may be very useful in the maintenance of integrated systems. After changes in some services, the monitoring module will collect data enabling the administrator to tune the system, to find services which should be optimized. The majority of existing monitoring modules is able to monitor business processes eg. Optimize for Process in webMethods. Other programs monitoring ESB, are dedicated to only one server or do not provide sufficient information if the execution is distributed on several servers.

Further work will concentrate on integration the monitoring tool with other tools enabling the generation of reports e.g. BMC Patrol [9], or HP OpenView [10].

References

1. He H.: *What is service-oriented architecture?* `http://webservices.xml.com/` (2003).
2. Kumar P.: *Web Services and IT management,* ACM QUEUE July/August (2005) 44-49.
3. Baresi L., Ghezzi C., Guinea S.: *Smart Monitors for Composed Services,* ACM ICSOC'04, November 15–19 (2004), New York, 193-202.
4. Mahbud K., Spanoudakis G.: *A Framework for Requirements Monitoring of Service Based Systems,* ICSOC'04, November 15–19, (2004), New York, New York, USA, 84-93.
5. Marchetti C., Pernici B., Plebani P.: *Quality Model for Multichannel Adaptive Information Systems,* WWW2004, May 17–22, 2004, New York, USA. ACM 1581139128/04/0005, 48-54.
6. Warda M.: *Architectures for Enterprise Application Integration,* MSC. Diploma (in polish), Institute of Computer Science Warsaw University of Technology (2006).
7. Bluemke I., Warda M.: *Monitoring module,* II Polish Conference on Data Processing Technology (in polish) (September 2007).
8. `http://www.webmethods.com` (April 2007)
9. BMC Patrol `http://www.bmc.com/` (April 2007)
10. OpenView `http://h20229.www2.hp.com/` (April 2007)

Proceedings of the International Multiconference on
Computer Science and Information Technology pp. 745–754
ISSN 1896-7094
© 2007 PIPS

Compositional Abstractions for Interacting Processes

Maciej Koutny[1], Giuseppe Pappalardo[2], and Marta Pietkiewicz-Koutny[1]

[1] School of Computing Science, Newcastle University,
Newcastle upon Tyne NE1 7RU, U.K.
{marta.koutny,maciej.koutny}@ncl.ac.uk
[2] Dipartimento di Matematica e Informatica, Università di Catania,
I-95125 Catania, Italy,
pappalardo@dmi.unict.it

Abstract. A promising way of dealing with complex behaviours of networks of communicating processes is to use abstractions. In our previous work, interface abstraction, modelled through a suitable relation, allowed us to 'interpret' the behaviour of an implementation process as that of a specification process, even in the event that their interfaces differ. The proposed relation is compositional, in the sense that a composition of communicating sub-systems may be implemented by connecting their respective implementations. But so far abstraction has been shown to distribute only over network composition which restricts its usefulness for compositional correctness analysis. In this paper we extend the treatment to other process constructs which proved to be useful in the development of complex distributed applications.

Introduction

The basic issue we aim at addressing in this work is the notion of implementation, and its relationship to abstraction, in the framework of communicating sequential processes. In general, we say that a process Q *implements* a process P when its behaviour, *conveniently interpreted through an appropriate form of abstraction*, is a potential behaviour of P, i.e., when the interpretation of Q is more deterministic than P. In the following, we refer to Q as the *implementation* (process), and to P as the corresponding *specification* or *target* or *base* (process).

Development or *refinement* consists in replacing a target P with a (usually) more complex or detailed process Q representing a proper implementation of P for the intended abstraction. This may possibly be accomplished in a stepwise fashion. Every refinement step should undergo *verification*, to be formally proved correct. A refinement step may be an instance of a known pattern (as in *program transformation* [3]), avoiding the need of verification through a specific, dedicated proof. Moreover, a correctness proof may also be an instance of application of a general theorem and/or methodology, as in compositional verification.

Conventionally, in process algebras, such as [12, 14], the notion of implementation differs from the framework outlined, in that it does not employ abstraction to interpret the behaviour of the implementation process Q as that of the target process P. The behaviour of a correct Q must indeed simply *be* (part of) that of P. Of course, if we shift

our attention from process behaviour (i.e., *meaning*, a semantic notion) to *expressions* of the process algebra at hand (a syntactic notion), we can still see (a different kind of) abstraction operating. Then, process *expression* q may be seen as an implementation of a syntactically different expression p, built out of different parts and operators, if the behaviour denoted by q, say Q, is part of that denoted by p, say P. Thus,'abstraction' here means abstraction from syntactic structure, and is applied to both the implementation and the target process expressions, q and p respectively. Though fundamental, this abstraction is irrelevant to our notion of implementation, which deals with 'pure' processes, Q and P. In it, a process *is* its behaviour. At the syntactic level, conventional refinement of course allows the designer to change the control structure of the target into the desired implementation. However, as implied by the previous discussion, their *interfaces* must coincide.

In refinement, it is often natural to implement abstract, high-level *interface* actions at a lower level of detail and in a more concrete manner. For example, an ideal channel in the target may in practice need to be replaced by a data/acknowledgement unreliable channel pair. Our abstraction-based implementation relation does provide a solution, which we deem in principle somehow more flexible than the *action-refinement* approach (cf. [13, 4, 9]).

In our previous work (see, e.g., [4, 5, 10]), we proposed abstraction-based implementation relations satisfying *realisability*, which is a property ensuring that an implementation may be put to good use, and *compositionality* which requires implementation relation to distribute over system composition. Thus, a specification composed of a number of connected systems may be implemented by connecting their respective implementations. Compositionality is important in avoiding the state explosion problem in carrying out automated verification.

Previously, we only dealt with compositionality over the parallel composition operator, a key tool in the construction of concurrent and distributed systems. However, there are other process combinators which proved to be useful for practical applications. Therefore, in this paper, we aim at extending proper distributivity results to other commonly used composition operators.

Another limitation of our previous results was that interprocess communication was assumed to be point-to-point, with typed input and output channels. We here remove this restriction as well, so that the proposed treatment can deal with arbitrary patterns of communication, e.g., broadcast.

The paper is organised as follows. In the next section, we briefly recall some basic notions used throughout the paper. The following section outlines our previous approach and results, and the last section sketches the proposed extension. Detailed motivations for the work on abstraction of interprocess communication, illustrative examples, and description of other related work, in particular [1, 2, 7, 11, 13, 15], can be found in [4, 9].

1 Preliminaries

We use the CSP process algebra [8, 14]. A CSP *process* P can be regarded as a black box which may engage in interaction with its environment. Atomic instances of this

interaction are called *actions* and must be elements of P's finite *alphabet*, αP. P's *traces*, or τP, are the finite sequences of actions that P can engage in. After a given trace t, process P may *refuse* to engage in a set of actions R, which is a convenient device to model deadlock-like situations as well as non-determinism. All such pairs (t, R) form the set of P's *failures*, ϕP. Finally, some of the traces may lead to unproductive internal loops, forming the set of P's *divergences*, δP.

We now recall the standard CSP operators. Parallel composition, $P \| Q$, models synchronous communication between processes in such a way that each of them is free to engage independently in any action that is not in the other's alphabet, but they have to engage simultaneously in any action in the intersection of their alphabets. Parallel composition is commutative and associative. The deterministic choice, $P [] Q$, and nondeterministic choice, $P \sqcap Q$, offer an alternative between the behaviours of P and Q. In the latter case, no external process has control about which of these options is actually followed (this property is useful, e.g., to model execution errors). Prefixing, $a \rightarrow P$, is a construct which first executes a and then behaves like P. Hiding a set of actions A in P yields process $P \backslash A$ that behaves like P with the actions occurring at the channels in A made invisible. Hiding is here the only operator which may introduce divergence. This happens whenever P can execute an infinite sequence of hidden actions.

There are also atomic CSP processes like STOP_A which denotes a terminated process with the alphabet A. One can also use recursive process definitions. For example, $P = (a \rightarrow P) [] (b \rightarrow \text{STOP})$ defines a process which can execute action a any number of times, and then perhaps execute b and terminate (often STOP's subscript alphabet is implicit). We use the following notations concerning traces.

- The empty trace is denoted by $\langle\,\rangle$, and $\circ$ is the concatenation operation for traces.
- $w \leq u$ means that trace w is a prefix of u, and $w < u$ that w is a proper prefix of u.
- A mapping f from traces to traces is monotonic if $t \leq u$ implies $f(t) \leq f(u)$.
- If w is a prefix of u then $u - w$ is the suffix of u after deleting the initial w.
- Given a trace t and a set of actions A, the trace $t \restriction A$ is obtained by deleting from t all the actions that do not belong to A.

2 Previous results

In our previous work (see, e.g., [4, 5, 10]), we regard processes $P_1, \ldots, P_n$ as forming a *network* if no channel is shared by more than two P_i's. (Note that such a process network assumes a one-to-one interprocess communication.) We then define $P_1 \otimes \cdots \otimes P_n$ to be the process obtained by taking the parallel composition of the processes and then hiding all interprocess communication, i.e., the process $(P_1 \| \cdots \| P_n) \backslash B$, where B is the set of actions shared by two different processes in the network. Network composition is commutative and associative.

We also use structured actions of the form $b{:}v$ with v a data *message* and b a communication *channel*, and partition the channels of a process P into the input channels, $in\ P$ (depicted by incoming arrows), and output channels, $out\ P$ (depicted by outgoing arrows). No two processes in a network have a common input (or output) channel.

In [10], we place restriction on the kind of allowed base processes, called *input-output process* (IO processes), by assuming them to be non-diverging processes with

all input channels being *value independent*. Intuitively, in an IO process the data component of a message arriving on an input channel c is irrelevant as far as accepting it is concerned; if one such message can be refused then so can any other message. In practice, standard programming constructs like $c?x$ for receiving messages give rise to value independent input channels. The requirement that an IO process P should be non-diverging is standard in a CSP based framework, as divergences basically signify totally unacceptable behaviour. The class of base IO processes is compositional, i.e., a network of IO processes is an IO process provided that the composition does not generate a divergence.

The notion of extraction pattern ep (used in [5, 9, 4]) relates behaviour on a set of "source" channels, B, in an implementation process to that on a "base" channel, b, in the target process. It has two main functions: that of interpretation of behaviour, necessitated by interface difference, and the encoding of some correctness requirements. The key part of ep is an *extraction* mapping, $extr$, which interprets a trace over the source channels B in terms of a trace over base channel b (thought of as belonging to the target process). Moreover, mapping $extr$, by way of its domain, identifies behaviour over source channels that is correct functionally (i.e., in terms of traces); indeed, "incorrect" traces over B need not (or cannot sensibly) be interpreted, thus making the domain of $extr$ potentially smaller. Another mapping, ref, is used to define correct behaviour in terms of failures, as it gives bounds on refusals after execution of a particular trace sequence over the source channels. The extraction mapping $extr$ is monotonic, as receiving more information cannot decrease the current knowledge about the transmission. Both notions can be lifted to a finite set of extraction patterns operating on disjoint sets of channels.

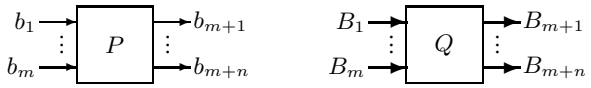


Fig. 1. Base process P and its implementation Q.

Suppose that we intend to implement a base IO process P in Figure 1 using another process Q with a possibly different communication interface, as in Figure 1 where thick arrows represent *sets* of channels.. The correctness of the implementation will be expressed in terms of two sets of extraction patterns, $In = \{ep_1, \ldots, ep_m\}$ and $Out = \{ep_{m+1}, \ldots, ep_{m+n}\}$, where each ep_i is an extraction pattern from B_i to b_i. The former (with sources $in\,Q$ and targets $in\,P$) will be used to relate the communication on the input channels of P and Q, the latter will serve a similar purpose for the output channels.

Under the above assumptions, Q is an *implementation* of P w.r.t. sets of extraction patterns In and Out, denoted $Q \preceq^{In}_{Out} P$, if the following hold: (i) all correct traces of Q can be interpreted as traces of P; (ii) it is not possible to execute Q indefinitely without extracting any actions of P; and (iii) a refusal by Q on a source channel set B_i,

exceeding the bound set by ref, must correspond to a refusal by P to interact at all on the target channel b_i.

A direct comparison of an implementation process Q with the base process P is only possible if there is no difference in the respective communication interfaces. This corresponds to the situation that both In and Out are sets of *identity* extraction patterns with $B_i = \{b_i\}$ and $extr_i$ an identity mapping (ref may be left undefined). In such a case, we simply denote $Q \preceq P$.

If $Q \preceq P$ then, in particular, all the refusals on input channels are preserved *entirely*, while for output channels any refusal by Q to output anything on a given channel is also present in P. The latter should indeed be considered as a very satisfactory state of affairs: Q will never fail to provide an output consistent with the specification, unless the specification process itself explicitly allows no output at all to be produced.

One can therefore consider that $Q \preceq P$ embodies a fully adequate notion of *realisability*. To further justify this, it is interesting to compare it with the standard *refinement* ordering of CSP, denoted by $\sqsupseteq$, such that $Q \sqsupseteq P$ (i.e., Q 'CSP implements or *refines*' P) basically amounts to stating that $\phi Q \subseteq \phi P$.

To start with, it is not difficult to check that $Q \sqsupseteq P$ implies $Q \preceq P$. Moreover, $\preceq$ collapses to $\sqsupseteq$ for the rather wide class of *output-determined* IO base processes (for such a process, the result produced on a given output channel is deterministic at any given point of its execution). Another significant comparison can be made in terms of what can be established by considering the way P and Q interact with a possible environment, as shown in Figure 2.

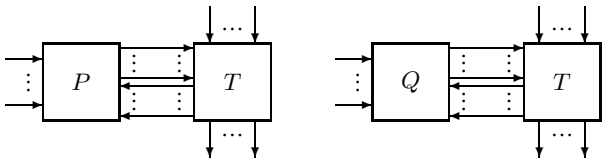


Fig. 2. Relating base (P) and implementation (Q) processes in the context of an environment T.

Here P is any base IO process, $Q \preceq P$ its implementation w.r.t. suitable identity extraction patterns, and T an IO process representing the environment. It can then be shown that $Q \otimes T \sqsupseteq P \otimes T$. Thus $Q \otimes T$ is at least as *deterministic* a process as $P \otimes T$ in the sense of CSP (see [8, 14]). This makes Q at least as good as P (and possibly much better) as a process to be used in practice.

We finally have a fundamental result, that the implementation relation is compositional. Let K and L be two base IO processes whose composition is non-diverging, as in Figure 3, and let Ep_c, Ep_d, Ep_e, Ep_f, Ep_g and Ep_h be sets of extraction patterns whose targets are respectively the channel sets C, D, E, F, G and H. Then

$$M \preceq^{Ep_c \cup Ep_h}_{Ep_d \cup Ep_e} K \ \wedge \ N \preceq^{Ep_d \cup Ep_f}_{Ep_g \cup Ep_h} L \implies M \otimes N \preceq^{Ep_c \cup Ep_f}_{Ep_e \cup Ep_g} K \otimes L \, .$$

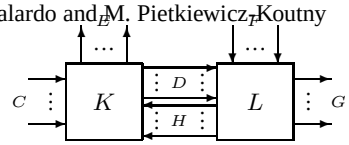


Fig. 3. Base processes used in the formulation of the compositionality result.

Hence the implementation relation is preserved through, or distributes over, network composition, and the only restriction on combining base processes is that their network should be designed in a divergence-free way.

3 Extending the model of process abstraction

In the proposed extension of the previous approach, we make a simplifying assumption that all processes are divergence-free. This is aimed mainly at easing the presentation, without effectively restricting the applicability of the resulting technique. As a consequence, a CSP process P can be identified with the pair $(\alpha P, \phi P)$ (ignoring the δP attribute). Furthermore, since hiding can introduce divergence, we assume in this section that it is a partial operation defined only if divergence is not generated. In fact, hiding leading to a divergence indicates a serious mistake in the construction of a process; this should be detected and eliminated before the proposed abstraction approach is applied, in full agreement with the standard CSP philosophy.

It should be stressed that in this section we: (i) do not make any assumptions about the channels a process can use, just work with the generic alphabets; and (ii) no longer assume any special properties of the base processes.

To begin with, we introduce a general definition of an abstraction mapping, which generalises the role played by the extraction patterns.

Definition 1. *Let Src (sources) and Trg (targets) be finite non-empty sets of actions. An* abstraction *from Src to Trg is a pair $abs \stackrel{\text{df}}{=} (extr, ref)$, where:*

- *$extr : dom \to Trg^*$, where $dom \subseteq Src^*$, is a mapping from traces over Src to the traces over Trg. It is assumed that the domain dom is non-empty and prefix-closed, and that $extr$ is monotonic, strict (i.e., $extr(\langle\,\rangle) = \langle\,\rangle$), and effective (i.e., for every infinite sequence $t_1 < t_2 < \ldots$ of traces in dom, the sequence $extr(t_1) \leq extr(t_2) \leq \ldots$ is unbounded).*
- *$ref : dom \times 2^{Src} \to 2^{Trg}$ is a total mapping which is subset-monotonic (i.e., if $t \in dom$ and $R \subseteq R' \in 2^{Src}$ then $ref(t, R) \subseteq ref(t, R')$) and terminable (i.e., $ref(t, Src) = Trg$, for all $t \in dom$).*

Moreover, we denote $abs(t, R) \stackrel{\text{df}}{=} (extr(t), ref(t, R))$, for all $t \in dom$ and $R \subseteq Src$, and call $extr$ and ref an extraction and refusal mappings, respectively.

The intuition behind the above definition is that if (t, R) is a trace/refusal pair for a set of actions Src in an implementation process, then these should be interpreted as the trace/refusal pair $abs(t, R)$ for a set of actions Trg in a base process. Since $extr$

is effective it is impossible for an implementation to execute an infinite trace which is 'invisible' as far as the base process is concerned. In some sense this can be viewed as generalising a divergence freedom requirement on the application of abstraction. Moreover, as ref is terminable, any termination in an implementation process must correspond to termination in the specification process.

3.1 An algebra of abstractions

To develop a satisfactory treatment of (interface) abstraction for constructors other than network composition, we now look at the problem of composing abstractions so as to match the way CSP processes are constructed.

In what follows, given $dom \subseteq Src^*$ and $dom' \subseteq Src'^*$, we define $dom \| dom' \subseteq (Src \cup Src')^*$ to be the set of all traces which, when projected on Src, give a trace in dom, and likewise for Src'.

We first characterise pairs of abstractions which can be composed. Below, we assume that $abs = (extr, ref)$ and $abs' = (extr', ref')$ are two abstractions, respectively from Src to Trg, and from Src' to Trg', and with domains dom and dom'.

Definition 2. *Two abstractions, abs and abs', are:*

- disjoint *if* $Src \cap Src' = Trg \cap Trg' = \varnothing$.
- overlapping *if* $Src \cap Src' = Trg \cap Trg' = \varnothing$.
- 0–compatible *if* $extr(t) = extr'(t)$, *for all* $t \in dom \cap dom'$.
- 1–compatible *if* $ref(t, R) = ref'(t, R)$, *for all* $t \in dom \cap dom'$ *and* $R \subseteq Src \cap Src'$.
- 2–compatible *if, for all* $t \in dom \cap dom'$ *and* $R, R' \subseteq Src \cap Src'$,

$$ref(t, R \cup R') \cup ref'(t, R \cup R') = ref(t, R) \cup ref'(t, R') \ .$$

For a pair of disjoint abstractions, abs and abs', we now construct an abstraction $abs \oplus abs'$ from $Src \cup Src'$ to the set $Trg \cup Trg'$, and with the domain $dom \| dom'$. The extraction mapping is defined by induction on the length of traces, by stipulating that it be strict, and for all $t \circ \langle a \rangle$ in the domain:

$$extr_{abs \oplus abs'}(t \circ \langle a \rangle) \stackrel{\text{df}}{=} \begin{cases} extr_{abs \oplus abs'}(t) \circ \\ \quad (extr(t \restriction Src \circ \langle a \rangle) - extr(t \restriction Src)) & \text{if } a \in Src \\ extr_{abs \oplus abs'}(t) \circ \\ \quad (extr'(t \restriction Src' \circ \langle a \rangle) - extr'(t \restriction Src')) & \text{otherwise .} \end{cases}$$

The definition of the refusal mapping is more straightforward, as for all $t \in dom_{abs \oplus abs'}$ and $R \subseteq Src \cup Src'$, we have:

$$ref_{abs \oplus abs'}(t, R) \stackrel{\text{df}}{=} ref(t \restriction Src, R \cap Src) \cup ref'(t \restriction Src', R \cap Src') \ .$$

Intuitively, $abs \oplus abs'$ captures the situation when the two abstractions operate on disjoint parts of the interface of a system, and the interpretations of behaviours executed at these interfaces are independent of each other (though the behaviours may be related in the semantic or operational sense).

Proposition 1. *$abs \oplus abs'$ is an abstraction in the sense of Definition 1.*

For a pair of overlapping $0\&1$–compatible abstractions, abs and abs', we construct an abstraction $abs \sqcap abs'$ from $Src = Src'$ to $Trg = Trg'$, and with the domain $dom_{abs \sqcap abs'} \stackrel{\text{df}}{=} dom \cup dom'$. Moreover, for all $t \in dom_{abs \sqcap abs'}$ and $R \subseteq Src$,

$$(abs \sqcap abs')(t, R) \stackrel{\text{df}}{=} \begin{cases} abs(t, R) & \text{if } t \in dom \\ abs'(t, R) & \text{otherwise .} \end{cases}$$

Intuitively, $abs \sqcap abs'$ captures the situation when the two abstractions characterise the behaviour of two alternative versions of a subsystem.

Proposition 2. *$abs \sqcap abs'$ is an abstraction in the sense of Definition 1.*

From a pair of overlapping $0\&2$–compatible abstractions, abs and abs', we construct an abstraction $abs \| abs'$ from $Src = Src'$ to $Trg = Trg'$, and with the domain $dom_{abs \| abs'} \stackrel{\text{df}}{=} dom \cap dom'$. Moreover, for all $t \in dom_{abs \| abs'}$ and $R \subseteq Src$,

$$(abs \| abs')(t, R) \stackrel{\text{df}}{=} (extr(t), ref(t, R) \cup ref'(t, R)) \ .$$

Intuitively, $abs \| abs'$ captures the situation when the two abstractions characterise the behaviour of two parallel subsystems at a shared interface.

Proposition 3. *$abs \| abs'$ is an abstraction in the sense of Definition 1.*

As in the case of extraction patterns and the realisability results recalled in the previous section, we also need some kind of abstraction allowing a direct comparison of interactions. The *identity* abstraction for a set of actions A is an abstraction $idabs_A$ from $Src = A$ to $Trg = A$ with the domain $dom = A^*$, and such that $idabs_{Src}(t, R) \stackrel{\text{df}}{=} (t, R)$, for all $t \in dom$ and $R \subseteq Src$.

Proposition 4. *If $A \cap A' = \varnothing$ then $idabs_{A \cup A'} = idabs_A \oplus idabs_{A'}$.*

3.2 Implementation relation

We now introduce the central notion of this paper which, despite its simplicity, is all we need to capture what it means for one process to correctly implement another, base, process with a possibly different communication interface.

Suppose that we intend to implement a base process P using another implementation process Q with possibly different alphabet. The correctness of the implementation will be expressed in terms of an abstraction from the alphabet of Q to that of P.

Definition 3. *Let P and Q be processes and abs be an abstraction from αQ to αP. Then Q is an* implementation *of P w.r.t. abs if $\tau Q \subseteq dom$ and $abs(\phi Q) \subseteq \phi P$.*[3] *We denote this by $Q \hookrightarrow_{abs} P$.*

[3] Note that $\tau Q \stackrel{\text{df}}{=} \{t \mid (t, R) \in \phi Q\}$ and so the first condition ensures that all failures of Q can be interpreted by the abstraction abs.

Hence it is possible to interpret the whole behaviour of an implementation process as (part of) the behaviour of a base process. This is, clearly, very close to the idea of one process being a refinement of another in the standard treatment of CSP processes.

A direct comparison of an implementation process Q with the corresponding base process P is only possible if they have the same alphabet A. Then, if $Q \hookrightarrow_{idabs_A} P$, we simply denote $Q \hookrightarrow P$ and obtain the strongest possible realisability result.

Theorem 1. *$Q \hookrightarrow P$ if and only if $Q \sqsupseteq P$.*

3.3 Compositionality results

We now establish compositionality properties linking the proposed algebra of abstractions with the algebra of CSP processes.

We start by assuming that there are two base processes, P and P', working in parallel, and two implementation processes, Q and Q', also working in parallel. In addition to that there are four abstractions: (i) abs from $\alpha Q \setminus \alpha Q'$ to $\alpha P \setminus \alpha P'$; (ii) abs' from $\alpha Q' \setminus \alpha Q$ to $\alpha P' \setminus \alpha P$; and (iii) abs_0 and abs'_0, both from $\alpha Q \cap \alpha Q'$ to $\alpha P \cap \alpha P'$, which are $0\&2$–compatible and overlapping.

Theorem 2. *Assuming the above, if $Q \hookrightarrow_{abs \oplus abs_0} P$ and $Q' \hookrightarrow_{abs' \oplus abs'_0} P'$, then:*

$$(Q \| Q') \hookrightarrow_{abs \oplus abs' \oplus (abs_0 \| abs'_0)} (P \| P') \ .$$

The next compositionality result concerns non-deterministic choice.

Theorem 3. *Let $Q \hookrightarrow_{abs} P$ and $Q' \hookrightarrow_{abs'} P'$ where abs and abs' are $0\&1$–compatible overlapping abstractions. Then $(Q \sqcap Q') \hookrightarrow_{abs \sqcap abs'} (P \sqcap P')$.*

The last compositionality result deals with hiding.

Theorem 4. *Let $Q \hookrightarrow_{abs \oplus abs'} P$, and $P \setminus Trg$ be a well-defined defined process. Then $Q \setminus Src$ is also defined and $(Q \setminus Src) \hookrightarrow_{abs'} (P \setminus Trg)$.*

Proof. (Sketch) The fact that abs is effective ensures that hiding the actions Src in Q does not create divergence in view that hiding the actions Trg does not create divergence in P. Thus $Q \setminus Src$ is defined, and using the terminability of abs we may guarantee that any total blocking on the hidden actions in the implementation process has a corresponding total blocking on the hidden actions in the base process. This allows one to prove the inclusions of abstracted traces and failures. □

We have demonstrated how one can deal with abstractions in the context of parallel composition, hiding and non-deterministic choice. Other standard operators of CSP, such as prefixing, can be treated in a similar way.

4 Conclusions

We have outlined a general compositional approach which allows one to deal with abstractions in networks of communicating processes. There are several issues which are of an immediate interest in the future work, such as applying the proposed approach to case studies and adding treatment for other CSP operators. However, we feel that the crucial one is a provision of algorithms and tools for checking whether an implementation relation between two processes indeed holds. Such a problem can be attempted using techniques similar to those introduced in [4] for the setup based on extraction patterns.

Acknowledgement This research was supported by the EC IST grant 511599 (RODIN).

References

1. M. Abadi and L. Lamport: The Existence of Refinement Mappings. *Theoretical Computer Science* 82 (1991) 253-284.
2. E. Brinksma, B. Jonsson, and F. Orava: Refining Interfaces of Communicating Systems. Proc. of *Coll. on Combining Paradigms for Software Development*, Springer-Verlag, Lecture Notes in Computer Science 494 (1991) 297-312.
3. R. M. Burstall and J. Darlington: A Transformation System for Developing Recursive Programs. *Journal of the ACM* 24 (1977) 44-67.
4. J. Burton, M. Koutny and G. Pappalardo: Relating Communicating Processes with Different Interfaces. *Fundamenta Informaticae* 59 (2004) 1-37. 87-96
5. J. Burton, M. Koutny, G. Pappalardo and M. Pietkiewicz-Koutny: Relating Communicating Processes with Different Interfaces. In: *Concurrency in Dependable Computing*, P. Ezhilchelvan and A. Romanovsky (Eds.). Kluwer Academic Publishers (2002) 1-20.
6. P. Collette and C. B. Jones: Enhancing the Tractability of Rely/Guarantee Specifications in the Development of Interfering Operations. CUMCS-95-10-3, Department of Computing Science, Manchester University (1995).
7. R. Gerth, R. Kuiper and J. Segers: Interface Refinement in Reactive Systems. Proc. of *CONCUR '92*, Springer-Verlag, Lecture Notes in Computer Science 630 (1992) 77-93.
8. C. A. R. Hoare: *Communicating Sequential Processes*. Prentice Hall (1985).
9. M. Koutny and G. Pappalardo: Behaviour Abstraction for Communicating Sequential Processes. *Fundamenta Informaticae* 48 (2001) 21-54.
10. M. Koutny, G. Pappalardo and M. Pietkiewicz-Koutny: Towards an Algebra of Abstractions for Communicating Processes. Proc. of *ACSD'06*, IEEE Computer Society (2006) 239-250.
11. L. Lamport: The Implementation of Reliable Distributed Multiprocess Systems. *Computer Networks* 2 (1978) 95-114.
12. R. Milner: *Communication and Concurrency*. Prentice Hall (1989).
13. A. Rensink and R. Gorrieri: Vertical Implementation. *Information and Computation* 170 (2001) 95-133.
14. A. W. Roscoe: *The Theory and Practice of Concurrency*. Prentice-Hall (1998).
15. H. Schepers and J. Hooman: Trace-based Compositional Reasoning About Fault-tolerant Systems. Proc. of *PARLE'93*, Springer-Verlag, Lecture Notes in Computer Science 694 (1993) 197-208.

Proceedings of the International Multiconference on
Computer Science and Information Technology pp. 755–764
ISSN 1896-7094
© 2007 PIPS

The Concept of Quasi-objects in a Temporal Intelligent System

Maria Antonina Mach

University of Economics,
Department of Artificial Intelligence Systems
Komandorska 118/120
53-345 Wroclaw, Poland
maria.mach@ae.wroc.pl

Abstract. The paper presents the notion and concept of quasi-objects that form a representation layer of a temporal intelligent system. The tasks of such a system are briefly discussed to justify the use of quasi-objects, the origin of the concept is presented, and the main features of quasi-objects are outlined. Two examples of quasi objects are presented and discussed. Finally, advantages of this concept are briefly shown.

Keywords: quasi-object, temporal intelligent system, heterogeneous domains.

1 Introduction

Modern enterprises that operate in a complex economic environment have to react to changes of this environment quickly and properly [10]. Any enterprise has to react to all types of changes: the past, the present and the anticipated ones. Analysis of past changes may be the starting point for prognosis, present changes influence current operations, while anticipated changes allow formulating strategic plans for the future. It is obvious that the time factor is absolutely crucial in the strategic analysis.

We assume that the analysis of the environment that respects and reflects its changes is performed by an intelligent system. As the analysis is put in the temporal context, the system has to be temporal in its nature to provide a complete analysis. By "complete" analysis we mean: putting elements of the environment on the time axis, providing information on past, present and future changes, which is followed by the historical analysis, the analysis of current trends, and analysis of future changes. Also the causal relationships between economic phenomena in time have to be taken into account, as they allow a proper evaluation of phenomena and processes in the environment of an enterprise [11]. Such an analysis will allow better, more accurate and quicker decisions on the current and future strategy.

2 A Temporal Intelligent System

By a temporal intelligent system we understand an artificial intelligence system that explicitly and directly performs temporal reasoning. Such a system not only contains a fact base, a rule base and an inference mechanism, but also takes and expresses time in a direct way. For an intelligent system to be considered temporal, explicit time references should be found at least in the representation and reasoning layers.

Temporal intelligent system may be considered a decision support system in a sense that its purpose is to support decisions that need an advanced temporal analysis of the environment. To perform this task, the system should be able to:

1. represent time – discrete and/or dense one, depending on the needs and on the nature of phenomena being analysed,
2. represent causal relationships between actions and/or phenomena in the environment, represent conditions to perform an action,
3. represent processes, among which are conditional and concurrent ones,
4. analyse future changes.

A temporal intelligent system is composed of several layers. The representation layer contains knowledge about the environment and is composed of several temporal knowledge bases. Such a division of the representation layer is based on several observations. The first one concerns heterogeneity of economic environment, that is of phenomena observed there, and of the pace of changes of these phenomena. This in turn leads to different time granularities, which justify using different representations for the elements of the environment. Moreover, the representation containing different temporal formalisms allows depicting knowledge about the environment in a more precise and complete manner. Using single temporal formalism may lead to omission of some features, if it is impossible to represent them with the formalism chosen. In consequence, the knowledge about the environment would be "flattened", that is would not concern some important facts or features. All these observations justify using several knowledge bases in the first layer of the system. This layer is responsible for a proper representation and description of elements in the heterogeneous environment of the enterprise. Choosing appropriate temporal formalisms fulfils this task. As these formalisms usually contain inference rules, that are also taken into account in the representation process (see e.g. [12] p. 172; [7]; [2] p. 129nn; [14]; [15]; [19]), they do not only allow to describe and represent heterogeneous elements of the environment and their temporal features, but they also provide a way of reasoning about each knowledge bases' behaviour. These temporal knowledge bases are called quasi-objects. This notion, its genesis and main features will be discussed in the next section.

The second – reasoning – layer is responsible for reasoning about the domain. It contains individual inference mechanism, one for each individual knowledge base.

Next there is an integration layer. Regardless of the reasoning process concerning individual parts of the environment, it is necessary to integrate the

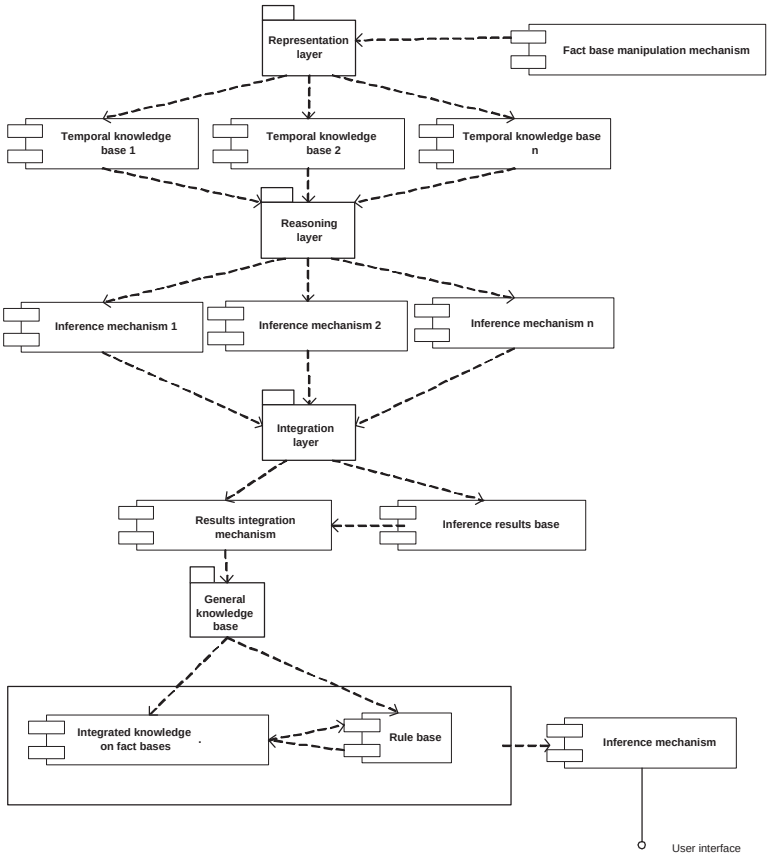


Fig. 1. General architecture of a temporal intelligent system with quasi-objects (temporal knowledge bases) in the first layer

temporal knowledge from the first layer, in order to obtain a general knowledge base. Having such knowledge base the system is able to reason about changes in the environment treated as a whole. The need for integration comes directly from the idea of quasi-objects: each of them is formalised in a different way, there occur differences in time granularity and semantic ones. Also the type of knowledge in each quasi-object is different, because characteristics of elements of the environment are different.

The last layer of the system is the one performing integrated reasoning and the analysis of future changes.

The general architecture of the system, with the first layer containing quasi-objects (temporal knowledge bases), is presented in Fig. 1.

Although each of the layers in the system is worthy of a detailed description, in the paper we will focus on the basic concept linked with this architecture,

namely the concept of quasi-objects, that constitute the layer responsible for temporal knowledge representation. The next section is devoted to this concept.

3 The Concept of Quasi-objects

A quasi-object is an individual knowledge base concerning a selected element of the environment (domain), formally represented by the means of a temporal logic, containing temporal knowledge and inference rules specific for the selected logic. Therefore the representation layer consists of a spectre of n knowledge bases (quasi-objects), where n is the number of domain elements represented.

The term "quasi-object" refers to the object-oriented paradigm, and emphasizes the fact, that despite some similarities with this paradigm, the concept of quasi-objects does not adopt mechanically all the assumptions of the paradigm. Using the term "quasi-object" is justified because – as Beynon-Davies points out – "there is no common agreement on what object paradigm really is" [3] p. 103. It seems that there is much freedom in using the term "object".

Although the concept of quasi-objects adopts some notions form the object-oriented paradigm, the two concepts are not identical. The comparison of both paradigms is presented in Table 1.

4 Examples

Formally, a quasi-object is defined as:

$$QO_n = \{K_n, A_n, R_n, O_n\} \tag{1}$$

where:
K_n – knowledge in a quasi-object n,
A_n – axioms of a logic used to formalize knowledge,
R_n – rules characteristic of a given logic,
O_n – assumed ontology of time.

Let us consider two examples of quasi-objects. The first one contains knowledge on legal barriers to entry to a market space, the second one – knowledge on capital barriers to entry. According to the above definition, the first quasi-object is built of the following elements:

a) Knowledge

Knowledge in this quasi-object concerns the need of obtaining a licence, according to the Polish law on economic activity [16]. It is encoded in the LTR language [20] in the following way[1]:

```
Attributes(licence, {who_issues, who_gets})
Attributes(is_issued, {what})
Attributes(valid, {what},_,)
Granularity(day)
```

[1] Basic concepts of the LTR theory are presented in Appendix A.

Table 1. Objects vs. quasi-objects.

CLASSIC OBJECT NOTIONS	QUASI-OBJECT NOTIONS
Object – a package of data and procedures	Quasi-object – temporal representation and temporal inference rules of a specific logic
Object – composed of state (value) and behaviour (operations)	Quasi-object – state (in a given moment) and evolution rules
Object – data structures and algorithm	Quasi-object – knowledge and inference rules
Object – attributes and methods	Quasi-object – features resulting from a given temporal logic, and knowledge base evolution rules
Object – represents a real-world phenomenon	Quasi-object – represents a real-world phenomenon
Object – a symbol representing one or more real-world „beings"	Quasi-object – represents a selected segment of the domain (environment)
Objects – grouped into classes	Quasi-objects may be grouped into classes but not necessarily
Class hierarchy and inheritance	No class hierarchy nor inheritance
Possible changes with reference to objects: – value of attribute – domain of attribute – a set of attributes – composition of objects – class membership of an object – types of relations between objects and/or classes – a set of methods describing object's behaviour	Possible changes with reference to quasi-objects: – appearance/disappearance of objects (resulting from changes in the domain) – a set of rules describing object's behaviour – a set of inference rules connected to a quasi-object – causality relationships between quasi-objects (domain elements) – features (attributes) of quasi-objects (domain elements)

```
If TT1: licence(issuing_authority, enterprise)
   TT2: is_issued(TT1)
   Occurs(TT2)
   Then Occurs(valid(TT1), instant(TT2))
If TT2: valid(TT1)
   Occurs(TT2)
   Then Holds_on(valid(TT1), period(TT3))
   Period(TT3) Equals [5y, 50y]
```

b) Axioms of the knowledge representation logic

The LTR theory is based on Vila and Schwalb's IP-CD theory [19], therefore axioms connected with this quasi-object are as follows:

IP_1: $\neg(i < i)$,

IP_2: $i < i' \Rightarrow \neg(i' < i)$,

IP_3: $i < i' \wedge i' < i'' \Rightarrow i , i''$,

IP_4: i < i' ∨ i' < i ∨ i = i',
$IP_{5.1}$: ∃i' (i' < i),
$IP_{5.2}$: ∃i' (i < i'),
IP_6: begin(i, p) ∧ end(i', p) ⇒ i < i',
$IP_{7.1}$: ∃i begin(i, p),
$IP_{7.2}$: ∃i end(i, p),
$IP_{8.1}$: begin(i, p) ∧ begin(i', p) ⇒ i = i',
$IP_{8.2}$: end(i, p) ∧ end(i', p) ⇒ i = i',
IP_9: i < i' ⇒ ∃p (begin(i, p) ∧ end(i', p)),
IP_{10}: begin(i, p) ∧ end(i', p) ∧ begin(i, p') ∧ end(i', p') ⇒ p = p'.

Axioms IP_1-IP_4 provide a strict linear order of time points, axioms $IP_{5.1.}$ and IP $_{5.2.}$ mean that a set of time points is unbounded, axiom IP_6 provides an ordering of interval's endpoints, axioms IP_7-IP_8 assure an unique startpoint and endpoint of any interval, axioms IP_9 and IP_{10} state that for a given ordered pair of timepoints an interval exists.

Moreover, the standard syntax and semantics of the 1st order predicate calculus (see e.g. [8], p. 43-54) is in force. It is augmented with temporal predicates Holds, Occurs, Holds_at, Holds_on (details are to be found in [20]).

c) LTR inference rules

The LTR theory is based on the 1st order predicate calculus, and because it assumes points and intervals of time as basic temporal entities, therefore the inference rules characteristic of this theory encompass predicate calculus rules and mixed point-interval TCSP.

d) Time ontology

According to the LTR theory assumptions, the basic temporal entities are points and intervals. The set of temporal relations is given as {<, *begin, end, next, previous, immediateBefore, immediateAfter*}. Time is ordered by a strict linear order.

The second quasi-object is built of the following elements:

a) Knowledge on capital barriers to entry

It is encoded using Allen's temporal logic [1]:

occurs(unstable(DER), p) ⇒ holds(high(barier), p)
& holds(high(barrier), p') & before(p, p') ∨ meets(p, p')
∨ overlaps(p, p')

occurs(high(barrier), p_1) ⇒ occurs(increase_of(OP), p_2)
∨ occurs(increase_of(DER), p_2) & before(p_2, p_1)

ecause(increase_of(rate), p_3, increase_of(capital, p_4)
& before(p_3, p_4) ∨ meets(p_3, p_4) ∨ overlaps(p_3, p_4)

(DER – dollar exchange rate, OP – oil prices)

b) Axioms of logic – Allen's theory

As knowledge in this quasi-object is formalized using Allen's theory, the part of this object are axioms concerning interval relations [18] p. 7:

A_1: $\forall$p, p' $\in$ P, R $\in$ AR $\exists$p': (R(p, p')) – given any interval there exists another interval related to it by a relation from Allen's set of relations, where AR denotes this set.

A_2: $\forall$p, p' $\in$ P, R $\in$ AR $\forall$R' $\in$ AR-R : R(p, p') $\Rightarrow$ $\neg$R'(p, p') – relations are mutually exclusive

A_3: relations have a transitive behavior, according to Allen's transitive table [1], p. 836.

c) Inference rules

Allen's interval logic is based on the 1st order predicate calculus, therefore its inference rules are the same as in this calculus, augmented with qualitative TCSP, because only intervals are considered as basic temporal entities.

d) Ontology of time

Allen assumes a time ontology consisting of basic temporal entities – intervals and 13 basic temporal relations between time intervals (see [1]; [2]).

5 Quasi-objects in a Temporal System – Advantages

Using the concept of quasi-objects in a temporal intelligent system may reveal the same advantages as those pointed out with reference to object-oriented paradigm (see e.g. [13], p. 218). Speaking in terms of a temporal system, these advantages would be as follows:

1. the concept of quasi-objects will make the construction and maintenance process of a knowledge base spectre easier. In case of changes (resulting e.g. from changes of domain elements) only one knowledge base will have to be changed (compare to objects' changes – [13], p. 254);
2. each quasi-object may be represented with means of different temporal logic, thanks to this the reality will be depicted in a more precise way;
3. knowledge divided into fragments, put into quasi-objects, will be more clear and ordered;
4. with quasi-objects, a temporal intelligent system will be more domain-oriented. In case when each knowledge base is constructed separately, it is more easy to understand and to represent fragments of the environment;
5. The construction process of a representation layer, based on quasi-object concept, is easier than constructing a traditional knowledge base, not divided into fragments;
6. Division of a knowledge base into quasi-objects makes the formalization of changes of domain elements easier, and enables modelling of domain's dynamic aspects.

Knowledge representation may be assessed according to different criteria. In regard of the criteria presented in [21], p. 31, the knowledge representation based on temporal formalisms, and organised in the form of quasi-objects, is:

1. clear – temporal formalisms, arising from classical logic, allow an easy identification of the domain represented,

2. precise – especially in the temporal aspect, thanks to the explicit notion of time,
3. natural – it presents the reality being modelled in an easy way, moreover, due to the features of temporal logics it is close to human commonsense reasoning and perception,
4. efficient – allows an easy access to the knowledge needed. It is so because the representation layer is divided into separate quasi-objects (access to knowledge on a selected fragment of the environment) and because of explicit temporal references (access to knowledge from any point/interval of time),
5. adequate to the domain (changing in time) and to the tasks of the system (temporal analysis),
6. modular – fragments of knowledge are independent (because quasi-objects are independent), but at the same time there are causal relationships between them. Nevertheless changes in one knowledge base (quasi-object) do not entail changes in other knowledge bases.

Summing up, the advantages of the proposed representation come from the advantages of the concept of quasi-objects and from the advantages of temporal logics. The latter are discussed e.g. in [9]; [4]; [6].

6 Conclusions and Future Research Directions

To the best of our knowledge, up till now in the literature concerning intelligent temporal systems only one knowledge base, and in consequence only one logical formalism has been assumed. If there was a need for integrating representations of elements in the domain, it was performed "beyond" the system (see e.g. [5]; [17]). Such a solution could lead to a loss of information and knowledge, as it has been pointed out in the paper. The concept of quasi-objects is an answer for the need to represent heterogeneous domains as precisely as possible.

It seems that this concept may lead to a better and more precise representation of knowledge about complex, changing domain of enterprise's environment. It is so despite some difficulties that may arise due to the necessity of formalizing knowledge using several temporal formalisms, and to the necessity of subsequent integration in higher layers of the system. With a more precise temporal knowledge representation, decisions generated by the system would be also more precise and accurate.

The concept of quasi-objects constitutes a first step in a way leading to construction of a heterogeneous temporal intelligent system. Such a representation may cause significant problems that will have to be solved while working on next system's layers. Once you have more than one formalism to represent different quasi-objects, there must be developed a technique to merge their semantics and/or inference procedures. In other words, having more than one logic language with its own formal semantics, putting them together may lead to an undecidable logic. If you want to reason on the (temporal) knowledge you have on all the quasi-objects in your system, you have to map all the languages (both

their syntax and their formal semantics) in a common language. If you have no restriction on temporal languages, the common language will very probably be FOL in all its expressiveness. And then you will have an undecidable language. Therefore, having these serious possible problems in mind, the future work will concern mainly developing an integration procedure (second layer) and inference mechanisms – to reason both about individual quasi-objects and merged knowledge gained from them.

References

1. Allen J. F., *Maintaining Knowledge about Temporal Intervals.* „Communications of the ACM", Vol. 26 No. 11, November 1983.
2. Allen J. F., *Towards a General Theory of Action and Time.* „Artificial Intelligence", Vol. 23 No. 2, July 1984.
3. Beynon-Davies P., *Systemy baz danych.* Wydanie trzecie rozszerzone i zmienione. Wydawnictwa Naukowo-Techniczne, Warszawa, 2003.
4. Hajnicz E., *Reprezentacja logiczna wiedzy zmieniającej się w czasie.* Akademicka Oficyna Wydawnicza PLJ, Warszawa 1996.
5. Hunter A., *Merging structured text using temporal knowledge Source.* "Data & Knowledge Engineering", Vol. 41, Iss. 1, April 2002, pp. 29–66, Elsevier Science Publishers B. V. Amsterdam, The Netherlands, The Netherlands.
6. Klimek R., *Wprowadzenie do logiki temporalnej.* Uczelniane Wydawnictwa Naukowo-Dydaktyczne AGH, Kraków 1999.
7. Levesque H., Pirri F., Reiter R., *Foundations for the Situation Calculus.* Linköping Electronic Articles in Computer and Information Science, Vol. 3(1998): nr 18. `http://www.ep.liu.se/ea/cis/1998/018` December 22, 1998.
8. Ligęza A., *Logical Foundations for Rule-Based Systems.* Uczelniane Wydawnictwa Naukowo-Dydaktyczne Akademii AGH w Krakowie, Kraków 2005.
9. Mach M., *Wspomaganie podejmowania decyzji za pomocą wnioskowania temporalnego.* In: Nycz M., Owoc M. L. (ed.), *Pozyskiwanie wiedzy i zarządzanie wiedzą.* Prace naukowe AE nr 1011, Wrocław 2004.
10. Nowak E. (ed.), *Prognozowanie gospodarcze.* Agencja Wydawnicza PLACET, Warszawa 1998.
11. Radosiński E., *Systemy informatyczne w dynamicznej analizie decyzyjnej.* Wydawnictwo Naukowe PWN, Warszawa-Wrocław 2001.
12. Russell S. J., Norvig P., *Artificial Intelligence – a Modern Approach.* Prentice Hall, Inc., USA 1995.
13. Silberschatz A., Korth H. F., Sudarshan S., *Database Systems Concepts.* The McGraw-Hill Companies, Inc., 1997.
14. Thielscher M., *Introduction to the Fluent Calculus*, Linköping Electronic Articles in Computer and Information Science, Vol. 3(1998): nr 14. `http://www.ep.liu.se/ea/cis/1998/14/` October 4, 1998.
15. Traverso P., Spalazzi L., *A Logic for Acting, Sensing and Planning.* Proc. IJCAI-95: 14^{th} International Joint Conference on Artificial Intelligence, Montreal, Canada, Vol. 2, pp. 1941-1947, Morgan Kaufmann 1995.
16. Ustawa z dnia 2 lipca 2004 „O swobodzie działalności gospodarczej", Dz. U. Nr 173, poz. 1807, 2004.

17. Visser U., Hübner S., *Temporal Representation and Reasoning for the Semantic Web. Technical report.* TZI-Bericht Nr. 28, 2003, http://www.tzi.de/fileadmin/publikation/TZI-Berichte/TZI-Bericht28.pdf
18. Vila L., *Formal Theories of Time and Temporal Incidence.* In: Fisher M., Gabbay D., Vila L. (Eds.), *Handbook of Temporal Reasoning in Artificial Intelligence.* Elsevier B. V., 2005.
19. Vila L., Schwalb E., *A Theory of Time and Temporal Incidence based on Instants and Periods.* TIME-96: Third International Workshop on Temporal Representation and Reasoning. Key West, Florida, USA, May 19-20, 1996.
20. Vila L., Yoshino H., *Time in automated legal reasoning*, In: Martino A., Nissan E. (eds.), "Information and Communications Technology Law", Special issue on formal models of legal time. Vol. 7 No. 3, 1998.
21. Zieliński J. S. (ed.), *Inteligentne systemy w zarządzaniu. Teoria i praktyka.* Wydawnictwo Naukowe PWN Warszawa 2000.

Appendix A – Basic Concepts of the LTR Theory

The LTR theory is based on the 1st order predicate calculus. The set of temporal relations is given as {*<, begin, end, next, previous, immediateBefore, immediateAfter*}. Time is ordered by a strict linear order, points and intervals are basic temporal elements. Basic axioms of the LTR theory are to be found in [20], of course standard axioms of the 1^{st} order predicate calculus also are in force. Predicates of the LTR are based on Allen's time theory. The *Holds* predicate is used both for points and intervals. The *holds_on* predicate concerns holding of a feature over an interval, and *holds_at* – holding of a feature in a time point.

The most important feature of the LTR theory, that needs more explanation here, concerns the so-called temporal tokens. They are used to link propositions with their times, which means a temporal qualification of propositions. Tokens may also be predicates' arguments. The method comes from a temporal arguments method, where time is introduced as one or more additional arguments, for example:

Valid(act, t1, t2)

Tokens, instead, link propositions with the time of occurrence, e.g.:

Valid(act, tt1)

which allows for the following interpretations: begin(tt1) = 01/01/1990 – the starting point for token tt1; or period(tt1) = [01/01/1990; 31/12/1990] – an interval over which a token tt1 is valid. It can be therefore said that a token represents a special temporal case of a temporal proposition.

A law article is formalised in the LTR theory as a rule or rules, that express relations between the occurrences of events (under certain conditions) and their effects, being the holding of certain features. The LTR language is a rule one, and does not need any assumption on reasoning method.

Proceedings of the International Multiconference on
Computer Science and Information Technology, pp. 765 – 778
ISSN 1896-7094
© 2007 PIPS

Federated Method Invocation with Exertions

Michael Sobolewski

Computer Science, Texas Tech University
SORCER Research Group, http://sorcer.cs.ttu.edu
sobol@cs.ttu.edu

Abstract. Six generations of RPC systems can be distinguished including Federated Method Invocation (FMI) presented in this paper. Some of them—CORBA, Java RMI, and Web/OGSA services—support distributed objects. However, creating object wrappers implementing remote interfaces doesn't have a great deal to do with object-oriented distributed programming. Distributed objects developed that way are usually ill-structured with missing core object-oriented traits: encapsulation, instantiation, inheritance, and network-centric messaging by ignoring the real nature of networking. A distributed system is not just a collection of distributed objects—it's the network of dynamic objects. In particular, the object wrapping approach does not help to cope with network-centric messaging, invocation latency, object discovery, dynamic object federations, fault detection, recovery, partial failure, etc. The Jini™ architecture does not hide the network; it allows the programmer to deal with the network reality: leases for network resources, distributed events, transactions, and discovery/join protocols to form service federations. An exertion-based architecture presented in this paper implements FMI to support service-oriented metaprogramming. The new triple Command pattern architecture presented in this paper uses Jini service management for managing the network of FMI objects.

1 Introduction

Socket-based communication forces us to design distributed applications using a read/write (input/output) interface, which is not how we generally design non-distributed applications based on procedure call (request/response) communication. In 1983, Birrell and Nelson devised remote procedure call (RPC) [2], a mechanism to allow programs to call procedures on other hosts. So far, six RPC generations can be distinguished:

1. First generation RPCs [2]—Sun RPC (ONC RPC) and DCE RPC, which are language, architecture, and OS independent;
2. Second generation RPCs—CORBA [20] and Microsoft DCOM-ORPC, which add distributed object support;
3. Third generation RPC—Java RMI [17] is conceptually similar to the second generation but supports the semantics of object invocation in different address spaces that are built for Java only. RMI fits cleanly into the language with no need for standardized data representation, external interface definition language, and with behavioral transfer that allows remote objects to perform operations that are determined at runtime;

4. Fourth generation RPCs—next generation of Java RMI, Jini Extensible Remote Invocation (Jini ERI) with dynamic proxies, smart proxies, network security, and with dependency injection defining exporters, end points, and security;
5. Fifth generation RPCs—Web/OGSA Services RPC [16, 27] and the XML movement including Microsoft WCF/.NET;
6. Sixth generation RPC—Federated Method Invocation (FMI), presented in this paper, allows for concurrent invocations on multiple federating hosts (virtual metacomputer) in the SORCER environment [25].

All the RPC generations are based on a form of service-oriented architecture (SOA) discussed in Section 2. However, CORBA, RMI, and Web/OGSA services are in fact object-oriented wrappers of network interfaces that hide distribution and ignore the real nature of network through classical abstractions of object-oriented programming using existing network technologies. The fact that object-oriented languages are used to create these object wrappers doesn't mean that developed distributed objects have a great deal to do with object-oriented distributed programming. For example, CORBA defines many services, and implementing them using distributed objects does not make them well structured with core object-oriented traits: encapsulation, instantiation, inheritance, and network-centric messaging. Similarly in RMI, marking objects with the `Remote` interface does not help to cope with network-centric messaging, object discovery, dynamic federation, fault detection, recovery, partial failure, etc.

Programmers use abstractions all the time. The source code written in programming language is an abstraction of the machine language. From machine language to object-oriented programming, layers of abstractions have accumulated like geological strata. Every generation of SW architects and programmers uses it's era's programming languages and tools to build programs of the next generation. Each architecture and programming language used reflects a relevant abstraction, and usually the type and quality of the abstraction implies the complexity of problems we are able to solve. For example, a procedural language provides an abstraction of an underlying machine language. Building on the object-oriented distributed paradigm is the Federated Service Object-Oriented (FSOO) paradigm exemplified by the Jini architecture [12] in which the network objects come together on the fly to play their predefined roles. In the Service-ORiented Computing EnviRonmet (SORCER) developed at Texas Tech University [25], a service provider is a remote object that accepts network requests — called *exertion*s—from service requestors to execute an item of work. While conventional objects encapsulate *data* and *operations*, exertions encapsulate *data*, *operations*, and *control strategy*. An exertion can federate on multiple hosts according to its *control strategy*.

The SORCER metacomputing environment adds an entirely new layer of abstraction to the practice of grid computing—*exertion-oriented programming*. The exertion-oriented programming makes a positive difference in service-oriented programming primarily through a new metaprogramming abstraction as experienced in many grid-computing projects including projects deployed at GE Global Research Center, GE Aviation, Air Force Research Lab, and SORCER Lab [19, 8, 9, 14, 22].

An exertion submitted to any provider in SORCER dynamically bounds to all relevant and currently available service providers on the network. The providers that

dynamically participate in this invocation are collectively called an *exertion federation*. This federation is also called a *virtual metacomputer* since federating services are located on multiple physical compute nodes held together by the FSOO infrastructure so that, to the individual exertion requestor, it looks and acts like a single computer.

The SORCER environment provides the means to create interactive FSOO programs [22] and execute them using the SORCER runtime infrastructure presented in Section 3. Exertions can be created using interactive user interfaces downloaded on the fly from service providers. Using these interfaces, the user can execute and monitor the execution of exertions within the FSOO metacomputer. The exertions can be persisted for later reuse, allowing the user to quickly create new applications or programs on the fly in terms of existing, usually persisted, exertions.

SORCER is based on the evolution of concepts and lessons learned in the FIPER project [5,21], a $21.5 million program founded by NIST (National Institute of Standards and Technology). Academic research on exertion-oriented programming has been established at the SORCER Laboratory, TTU, [25] where 23 SORCER related research studies have been investigated so far [26]. The current version of FMI used in SORCER is described in this paper.

The paper is organized as follows. Section 2 provides a brief description of a service oriented architecture with a related discussion of distribution transparency; Section 3 describes the SORCER methodology; Section 4 presents federated method invocation; Section 5 provides concluding remarks.

2 SOA and Distribution Transparency

Various definitions of a Service-Oriented Architecture (SOA) leave a lot of room for interpretation. In general terms, SOA is a software architecture using loosely coupled software services that integrates them into a distributed computing system by means of service-oriented programming. Service providers in the SOA environment are made available as independent service components that can be accessed without a priori knowledge of their underlying platform or implementation. While the client-server architecture separates a client from a server, SOA introduces a third component, a service registry. In SOA, the client is referred to as a service requestor and the server as a service provider. The provider is responsible for deploying a service on the network, publishing its service to one or more registries, and allowing requestors to bind and execute the service. Providers advertise their availability on the network; registries intercept these announcements and add published services. The requestor looks up a service by sending queries to registries and making selections from the available services. Requestors and providers can use discovery and join protocols to locate registries and then publish or acquire services on the network. We can distinguish the *service object-oriented architectures* (SOOA), where providers are network objects accepting remote invocations, from *service protocol oriented architectures* (SPOA), where a communication protocol is fixed and known beforehand by the provider and requestor. Based on that protocol and a service description obtained from the service registry, the requestor can bind to the service provider by creating a proxy used for remote communication over the fixed protocol. In SPOA a service is usually identified by a name. If a service provider registers its

service description by name, the requestors have to know the name of the service beforehand.

In SOOA, a proxy—an object implementing the same service interfaces as its service provider—is registered with the registries and it is always ready for use by requestors. Thus, in SOOA, the service provider publishes the proxy as the active surrogate object with a codebase annotation, e.g., URLs to the code defining proxy behavior (RMI and Jini ERI). In SPOA, by contrast, a passive service description is registered (e.g., an XML document in WSDL for Web/OGSA services, or an interface description in IDL for CORBA); the requestor then has to generate the proxy (a stub forwarding calls to a provider) based on a service description and the fixed communication protocol (e.g., SOAP in Web/OGSA services, IIOP in CORBA). This is referred to as a bind operation. The binding operation is not required in SOOA since the requestor holds the active surrogate object obtained from the registry.

Web services and OGSA services cannot change the communication protocol between requestors and providers while the SOOA approach is protocol neutral [30]. In SOOA, how an object proxy communicates with a provider is established by the contract between the provider and its published proxy and defined by the provider implementation. The proxy's requestor does not need to know who implements the interface or how it is implemented. So-called smart proxies (Jini ERI) grant access to local and remote resources; they can also communicate with multiple providers on the network regardless of who originally registered the proxy. Thus, separate providers on the network can implement different parts of the smart proxy interface. Communication protocols may also vary, and a single smart proxy can also talk over multiple protocols including application specific protocols.

SPOA and SOOA differ in their method of discovering the service registry. SORCER uses dynamic discovery protocols to locate available registries (lookup services) as defined in the Jini architecture [12]. Neither the requestor who is looking up a proxy by its interfaces nor the provider registering a proxy needs to know specific locations. In SPOA, however, the requestor and provider usually do need to know the explicit location of the service registry—e.g., the IP address of an ONC/RPC portmapper, a URL for RMI registry, a URL for UDDI registry, an IP address of a COS Name Server—to open a static connection and find or register a service. In deployment of Web and OGSA services, a UDDI registry is sometimes even omitted (WSDL descriptions are shared via files outside of the system); in SOOA, lookup services are mandatory due to the dynamic nature of objects identified by service types. Interactions in SPOA are more like client-server connections (e.g., HTTP, SOAP, IIOP), in many cases with no need to use service registries at all.

Let's emphasize the major distinction between SOOA and SPOA: in SOOA, a proxy is created and always owned by the service provider, but in SPOA, the requestor creates and owns a proxy which has to meet the requirements of the protocol that the provider and requestor agreed upon a priori. Thus, in SPOA the protocol is always a generic one, reduced to a common denominator—one size fits all—that leads to inefficient network communication in some cases. In SOOA, each provider can decide on the most efficient protocol(s) needed for a particular distributed application.

Service providers in SOOA can be considered as independent network objects finding each other via service registries and communicating through message passing. A collection of these objects sending and receiving messages—the only way these

objects communicate with one another—looks very much like a service object-oriented distributed system.

Do you remember the eight fallacies of network computing? [4] We cannot just take an object-oriented program developed without distribution in mind and make it a distributed system, ignoring the unpredictable network behavior. Most RPC systems, except Jini, hide the network behavior and try to transform local communication into remote communication by creating distribution transparency based on a local assumption of what the network might be. However every single distributed object cannot do that in a uniform way as the network is a heterogeneous distributed system and cannot be represented completely within a single entity.

The network is dynamic, can't be constant, and introduces latency for remote invocations. Network latency also depends on potential failure handling and recovery mechanisms so we cannot assume that a local invocation is similar to remote invocation. Thus complete transparency distribution—by making calls on distributed objects as though they were local—is impossible to achieve in practice. The distribution is not just an object-oriented implementation of a single type of distributed object; it's a metasystemic issue in object-oriented distributed programming.

Exertion-based programming is introduced to handle the metasystemic distribution in SORCER by using indirect remote method invocation with no service provider explicitly specified in the network request (exertion). Specific infrastructure objects support exertion-oriented programming combined with FMI. That infrastructure defines SORCER's distributed object modularity, extensibility, and reuse of service-oriented components consistent with the relevant metacomputing granularity and dependency injection—key features of object-oriented distributed programming that are usually missing in SPOA programming environments.

3 Federated Service Object-oriented Computing Environment: SORCER

SORCER is a federated service-to-service (S2S) metacomputing environment that treats service providers as network objects with well-defined semantics of a federated service object-oriented architecture (FSOOA). It is based on Jini semantics of services [12] in the network and Jini programming model with explicit leases, distributed events, transactions, and discovery/join protocols. While Jini focuses on service management in a networked environment, SORCR is focused on exertion-oriented programming and the execution environment for exertions. SORCER uses Jini discovery/join protocols to implement its FSOOA and FMI.

In SOOA, a service provider is an object that accepts remote messages from service requestors to execute an item of work. These messages are called *service exertions* that describe service data, operations and provider's control strategy. A *task exertion* is an elementary service request, a kind of elementary remote instruction (elementary statement) executed by a single service provider or a small-scale federation. A composite exertion called a *job exertion* is defined hierarchically in terms of tasks and other jobs, a kind of network procedure executed by a large-scale federation. The executing exertion is a service-oriented program that is dynamically bound to all needed and currently available service providers on the network. This

collection of providers identified in runtime is called an *exertion federation*. This federation is also called an *exertion space*. While this sounds similar to the object-oriented paradigm, it really isn't. In the object-oriented paradigm, the object space is a program itself; here the exertion space is the *execution environment* for the exertion that is a service-oriented distributed program. This changes the programming paradigm completely. In the former case the object space is hosted by a single computer, but in the latter case the service providers are hosted by the network of computers.

The overlay network of service providers is called the *service provider grid* and an exertion federation is called a *virtual metacomputer*. The *metainstruction set* of the metacomputer consists of all operations offered by all service providers in the grid. Thus, a service-oriented program is composed of metainstructions with its own service-oriented control strategy and service context representing the metaprogram parameters. The service context describes the data that tasks and jobs work on. Exertion-oriented programs (metaprograms) can be created interactively [22] and allow for a dynamic federation to transparently coordinate their execution within the grid. Please note that these metacomputing concepts are defined differently in classical grid computing where a job is just an executing process for a submitted executable code with no federation being formed.

Each SORCER provider offers services to other peers [8] on the object-oriented overlay network. These services are exposed indirectly by methods in well-known public remote interfaces and considered as elementary (tasks) or compound (jobs) statements of the FSOOA [21]. Requestors do not need to know the exact location of a provider beforehand; they can find it dynamically by discovering Jini lookup services and then looking up a needed service implementing required service types. Service providers do not have mutual associations prior to the execution of an exertion; they come together dynamically (federate) for all nested tasks and jobs in the exertion. Specialized providers within the federation, or task peers, execute service tasks. Jobs are coordinated by a *rendezvous* or *job peer* called a *Jobber*, one of SORCER infrastructure services. However, a job can be sent to any service provider (peer). A peer that is not a Jobber type is responsible for forwarding the job to one of available rendezvous peers in the SORCER grid and returning results to the requestor.

Thus implicitly, any peer can handle any job or task. Once the job execution is complete, the federation dissolves and the providers disperse to seek other exertions to join. Also, SORCER supports a traditional approach to grid computing similar to those found in Condor [28] and Globus [27]. Here, instead of exertions being executed by services providing business logic for requested exertions, the business logic comes from the service requestor's executable programs that seek compute resources on the network.

Grid-based services in the SORCER environment include *Grider* services collaborating with Jobber services for traditional grid job submission, and *Caller* and *Methoder* services for task execution [13]. Callers execute conventional programs via a system call as described in the service context of a submitted task. Methoders download required Java code (task method) from requestors to process any submitted context accordingly with the downloaded code. In either case, the business logic comes from requestors; it is conventional executable code invoked by Callers with the standard Caller's service context or mobile Java code executed by Methoders with any service context provided by the requestor.

4 Federated Method Invocation (FMI)

Each programming language provides a specific computing abstraction. Procedural languages are abstractions of assembly languages. Object-oriented languages abstract elements in the application domain that refer to "objects" communicating via message passing as their representation in the corresponding solution space. The object-oriented distributed programming should allow us to describe the distributed problem in terms of the intrinsic unpredictable network problem instead of in terms of distributed objects that hide the notion of the network.

What intrinsic distributed abstractions are defined in SORCER? Well, *service providers* are "objects", but they are specific objects—they are *network objects* with a *network state*, *network behavior*, and *network type(s)*. There is still a connection to distributed objects: each service provider looks like a distributed object (compute node) in that it has a network state, network behavior, and network types(s). Service providers act also as *network peers;* they are replicated and dynamically provisioned for reliability to compensate for network failures [18]. They can be found dynamically in runtime by type(s) they implement. They can federate for executing a specific network request called an *exertion* and perform hierarchically nested (component) exertions. An exertion encapsulates service *data*, *operations*, and provider's *control strategy*. The component exertions may need to share context data of ancestor exertions, and the top-level exertion is complete only if all nested exertions are successful.

With that very concise introduction to the abstractions of exertion-based programming, let's look in detail at how Federated Method Invocation (FMI) is structured.

4.1 Service Messaging and Exertions

In object-oriented terminology, a message is the single means of passing control to an object. If the object responds to the message, it has an operation and its implementation (method) for that message. Because object data is encapsulated and not directly accessible, a message is the only way to send data from one object to another. Each message specifies the name (identifier) of the receiving object, the name (selector) of operation to be invoked, and its parameters. In the unreliable network of objects; the receiving object might not be present or can go away at any time. Thus, we should postpone receiving object identification as late as possible. Grouping related messages per one request for the same data set makes a lot of sense due to network invocation latency and common errors in handling. These observations lead us to service-oriented messages called *exertions* that encapsulate both multiple *service signatures* that define operations and s*ervice context* as data. Different types of exertions define own execution control strategies. Two basic exertion types are distinguished: elementary and composite exertion called *service task* and *service job* respectively. There are two ways of invoking exertions. In the first case, an `Exertion` can be invoked by calling `Exertion.exert(Transaction)`. The second way is explained in Subsection 4.6.

4.2 Service Signatures

An exertion initiates the dynamic federation of all needed service providers dynamically—as late as possible—as specified by signatures of top-level and nested exertions. Thus, FMI is defined as exerting signatures, which is essentially an indirect invocation of network methods specified by the exertion signatures for related service contexts. SORCER service providers and requestors usually communicate via FMI.

A service `Signature` is defined by:

- signature name— a custom name
- service type— a Java interface name
- selector of the service operation—an operation name of the service type
- operation type— `Signature.Type` : `PROCESS` (default), `PREPROCESS` , `POSTPROCESS`
- service access type— `Signature.Access` ; `PUSH` (default) direct binding to Jobbers or Taskers , or `DROP` using the Spacer service—shared exertion space
- priority— used by exertion's control strategy
- execution time flag—if true, the execution time is returned in the service context
- notifyees—list of email addresses to notify upon exertion completion
- service attributes—requestor's attributes matching provider's registration attributes

An exertion can comprise of a collection of `PREPRROCESS` and `POSTPROCESS` signatures, but only one `PROCESS` signature. The `PROCESS` signature defines the binding provider for the exertion.

4.3 Exertion Types

A `Task` is the analog of a statement in conventional programming languages—here an elementary step of the exertion-oriented program. Thus, it is a minimal unit of structuring in exertion-oriented programming. If the provider responds to a `Task`, it has a method for the task's `PROCESS` signature. Other signatures associated with the `Task` provide for preprocessing and postprocessing by the same or its federating providers. An `APPEND` signature provides for the context received from the provider identified by this signature to be appended in runtime to the task's currently processed service context. Appending a service context allows a requestor to use actual data in runtime not available to the requestor when a task is submitted. A `Task` is the single means of passing control to a `PROCESS` provider. Note that a task is a batch of operations that operate on the same service context—a `Task` shared execution state—and all operations of the `Task`, as defined by signatures, can be executed by the same provider or a group of federating providers coordinated by the `PROCESS` provider—the provider identified by the `PROCESS` signature of the `Exertion`.

A `Job` is the analog of a procedure in conventional programming languages—here a federated procedure is an exertion-oriented program. It is a composite of exertions (see Figure 4) that makeup the federated procedure. The following flow control exertion types define algorithmic logic of exertion-oriented programming: **`ServiceTask`, `ServiceJob`, `IfExertion`, `WhileExertion`, `ForExertion`,** `DoExertionThrowExertion`, `TryExertion`, `BreakExertion`, `ContinueExertion`. Currently implemented flow control `Exertion` types in SORCER are indicated above in bold.

4.4 Service Contexts

A service context is a data structure that describes service provider ontology along with related data. A provider ontology is controlled by provider vocabulary that describes objects and the relations between them in a provider's namespace within a specified service domain of interest. A requestor submitting an exertion to a provider has to comply with that ontology. In service context attributes and their values are used as atomic conceptual primitives, and complements are used as composite ones. A complement is an attribute sequence (path) with a value at the last position. An elementary context property consists of a context subject (main complement) and a set of context complements, and usually corresponds to a simple sentence of natural language.

A service context is a tree-like structure described conceptually in the EBNF conceptual syntax specification as follows:

1. context = [subject ":"] complement { complement }.
2. subject = element.
3. complement = element ";".
4. element= path ["=" value].
5. path = attribute { "/" attribute } [{ "<" association ">" }] [{ "/" attribute }].
6. value = object.
7. attribute = identifier.
8. relation = domain product.
9. association = domain tuple.
10. product = attribute { "|" attribute }.
11. tuple = value { "|" value }.
12. attribute = identifier.
13. domain = identifier.
14. association = identifier.
15. identifier = letter { letter | digit }.

A relation with a single attribute is called a *property* and is denoted as attribute | attribute. To illustrate the idea of context, let's consider the following example:

laboratory/name = SORCER: university=TTU;
university/department/name=CS;
university/department/room/number=20B;
university/department/room/phone/number=806-742-
university/department/room/phone/ext=237;
director <person | Mike | W | Sobolewski> /email=sobol@cs.ttu.edu;

where the relation person is defined as follows: person | firstname | initial | lastname.

A context leaf node, or *data node* is where the actual data resides. The service context—all context paths—denotes an application domain namespace, and a *context model* is its context with data nodes appended to its context paths. A context path is a hierarchical name for a data item in a leaf node. Note that Context can be represented as an XML document—what has been done in SORCER for interoperability—but the power of object Contexts comes from the fact that any Java object can be naturally used as a data node. In particular exertions themselves can be used as data nodes and then executed and controlled by providers to run complex iterative programs, e.g., nonlinear multidisciplinary optimization [14].

4.5 Service-to-Service (S2S) Computing

Tasks are usually executed by providers of the `Tasker` type (task peer). A `Job` contains a service context called *control context* that describes the control strategy for the job exertion. Dedicated service providers of the `Jobber` type (job peer also called rendezvous peer), interpret and execute a job's control context in terms of the job's nested exertions accordingly. A `Jobber` manages a shared context (shared execution state) for the job federation and provides a substitution for input context parameter mappings. A `Jobber` creates a federation of required service providers (`Tasker`s and `Jobber`s) in runtime. A SORCER peer (`Servicer`) that is unable to execute an `Exertion` for any reason forwards the `Exertion` to any available `Servicer` matching the exertion's `PROCESS` signature and returns the resulting exertion back to its requestor.

All SORCER service providers are service peers as they implement the top-level `Servicer` interface. As a result, each `Servicer` can initiate a federation created in response to `Servicer.service(Exertion, Transaction)`. Servicers come together to form a federation participating in execution of the same exertion. When the exertion is complete, `Servicer`s leave the federation and seek a new exertion to join. Note that the same exertion can form a different federation for each execution due to the dynamic nature of looking up `Servicer`s by their implemented custom interfaces. Despite the fact that every `Servicer` can accept any exertion, `Servicer`s have well defined roles in SORCER S2S exertion-oriented programming:

a) `Tasker`s – process service tasks
b) `Jobber`s – process service jobs
c) `Contexter`s – provide service contexts for APPEND Signatures
d) `FileStorer`s – provide access to federated file system providers [1, 23]
e) `Cataloger` – `Servicer` registries
f) `Persister`s – persist service contexts, tasks, and jobs to be reused for interactive exertion-based programming
g) `Spacer`s – manage exertion spaces shared across Servicers for space-based computing [7]
h) `Relayer`s – gateway providers, transform exertions to native representation, for example integration with Web services and JXTA
i) `Autenticator`s, `Authorizer`s, `Policer`s, `KeyStorer`s – provide support for service-oriented security
j) `Auditor`s, `Reporter`s, `Logger`s – support for accountability, reporting and logging
k) `Grider`s, `Caller`s, `Methoder`s – support conventional grid computing
l) Generic `ServiceTasker` and `ServiceJobber` implementations are used to configure domain specific providers via dependency injection—configuration files for smart proxying and inserting business objects called SORCER service beans.

4.6 FMI Triple Command Pattern

Polymorphism lets us encapsulate a request—an exertion—then establish the signature of operation to call and vary the effect of calling the underlying operation by varying its implementation. The Command design pattern [10] establishes an operation signature as an interface and defines various implementations of the interface. In FMI, the following three operations are defined:

1. `Exertion.exert(Transaction):Exertion`—join the federation;
2. `Servicer.service(Exertion, Transaction):Exertion`—request a service in the federation from the top-level `Servicer` obtained by the receiver;
3. `Exerter.exert(Exertion, Transaction):Exertion`—execute the argument exertion by the target provider in the federation.

The above *Triple Command* pattern defines various implementations of these interfaces: `Exertion`, `Servicer`, and `Exerter`. This approach allows for the P2P environment via the `Servicer` interface, extensive modularization of `Exertion`s and `Exerter`s, and extensibility from the triple design pattern so requestors can submit any service-oriented programs (exertions) they want with or without transactional semantics. FMI triple Command Pattern is used as follows:

1. An exertion can be invoked by calling `Exertion.exert(Transaction).` The `Exertion.exert` operation implemented in `ServiceExertion` uses `ServicerAccessor` to locate in runtime the provider matching the exertion's PROCESS signature.
2. If the matching provider is found, then on its access proxy (that can also be a smart proxy) the `Servicer.service(Exertion, Transaction)` method is invoked.
3. When the requestor is authenticated and authorized by the provider to invoke the method defined by the exertion's `PROCESS` signature, then the provider calls its own `exert` operation: `Exerter.exert(Exertion, Transaction)`.
4. `Exerter.exert` method calls `exert` either of `ServiceTasker` or `ServiceJobber` (depending on the type of the exertion: either `Task` or `Job`) that by reflection calls the method specified in the `PROCES` signature (interface and selector) of the exertion. All application domain methods of any application interface (custom `Tasker` interfaces) have the same signature: a single `Context` type parameter and a `Context` type return vale. Thus a custom interface looks like an RMI interface with the above simplification on the common signature for all interface methods.

In the FMI approach, a requestor can create any `Exertion`, composed from any hierarchically nested `Exertion`s, with any service provider supplied ontology. The provider's service context ontology, object proxies and registration object attributes are network-centric; all of them are part of the provider's registration so they can be accessed via `Cataloger` or lookup services by any requestor on the network, e.g., service browsers [11] or custom service UI user agents [29] providing interactive exertion-oriented programming. In SORCER, using these zero-install service UIs, the user can define data for downloaded ontology and create a task/job to be executed on the virtual metacomputer.

Individual `Provider`s, in particular `Tasker`s and `Jobber`s, implement their own `exert(Exertion, Transaction)` methods according to their service semantics and control strategy; in SORCER implemented by `ServiceTasker` and `ServiceJobber` respectively. SORCER specific domain providers either subclass `ServiceTasker` or `ServiceJobber`, or by dependency injection (using Jini configuration methodology) configure either one with one of 12 proxying methods developed in SORCER. In general, many different types of taskers and jobbers can be used in SORCER at the same time (currently one `ServiceTasker` and one `ServiceJobber` implementation exists) and exertions via their signatures will make appropriate runtime choices as to what virtual metacomputer to run.

Invoking an exertion, let's say `ext`, is similar to invoking an executable program `ext.exe` at the command prompt. If we use the Tenex C shell (`tcsh`), invoking the program is equivalent to: `tcsh ext.exe`, i.e., passing the executable `ext.exe` to `tcsh`. Similarly, to invoke a metaprogram using FMI, in this case the exertion `ext`, we call `ext.exert(null)` if no transactional semantics is required. Thus, the exertion is the metaprogram and the network shell at the same time, which might first come as a surprise, but close evaluation of this fact shows it to be consistent with the meaning of object-oriented federated programming. Here, the *virtual metacomputer* is a federation that does not exist when the exertion is created. Thus, the notion of the *virtual metacomputer* is enclosed in the exertion exemplified by FMI.

The observation concluding that the exertion is the metaprogram and the network shell at the same time brings us back to the distribution transparency issue discussed in Section 2. It might appear that `Exertion` objects are network wrappers as they hide network intrinsic unpredictable behavior. However, `Exertion`s are not distributed objects, as do not implement any remote interfaces; they are local objects. `Servicer`s are distributed objects and there are many types of `Servicer`s addressing different aspects of networking. The network intrinsic unpredictable network behavior is addressed by the SORCER object-oriented distributed infrastructure: `Tasker`s, `Jobber`s, `Cataloger`s, `Spacer`s, `FileStorer`s, `Authenticator`s, `Authorizer`s, `Policer`s, etc. The `Servicer`-based infrastructure facilitates exertion-oriented programming and concurrent metaprogram execution using the presented FMI and allows for constructing reliable object oriented distributed systems from unreliable distribute components - `Servicer`s.

5 Conclusions

A distributed system is not just a collection of distributed objects—it's the network of dynamic objects. From an object-oriented point of view, the network of dynamic objects is the *problem domain* of object-oriented distributed programming that requires relevant abstractions in the *solution space*. The exertion-based programming introduces the new RPC abstraction with *service providers* and *exertions* instead of object-oriented conventional *objects* and *messages*. Service providers can register proxies, including smart proxies, via dependency injection using twelve methods investigated in SORCER. Executing a top-level exertion means forming a dynamic

federation of currently available providers in the network that collaboratively processes service contexts of all nested exertions. Services are invoked by passing exertions on to providers indirectly via service object proxies that in fact are access proxies allowing for service providers to enforce a security policy on access to services. When a permission is granted, then the operation defined by an exertion's `PROCESS` signature is invoked by reflection. FMI allows for the P2P environment via the `Servicer` interface, extensive modularization of `Exertion`s and `Exerter`s, and extensibility from the triple command design pattern. The presented FMI has been successfully deployed and tested in multiple concurrent engineering and large-scale distributed applications.

References

1. Berger M., and Sobolewski M., *SILENUS – A Federated Service-oriented Approach to Distributed File Systems,* In Next Generation Concurrent Engineering, ISPE/Omnipress, pp. 89-96 (2005).
2. Birrell A. D. & Nelson B. J., *Implementing Remote Procedure Calls,* XEROX CSL-83-7, October 1983.
3. Edwards W. K., *Core Jini,* 2nd ed., Prentice Hall (2000).
4. *Fallacies of Distributed Computing,* Accessed on: July 15, 2007. Available at: `http://en.wikipedia.org/wiki/Fallacies_of_Distributed_Computing`
5. *FIPER: Federated Intelligent Product EnviRonmet,* Available at: `http://sorcer.cs.ttu.edu/fiper/fiper.html` Accessed on: July 15, 2007.
6. Foster I., Kesselman C., Tuecke S., *The Anatomy of the J. Supercomputer Applications,* 15(3) (2001).
7. Freeman E., Hupfer S., & Arnold K. *JavaSpaces™ Principles, Patterns, and Practice,* Addison-Wesley, ISBN: 0-201-30955-6 (1999).
8. Goel S., Shashishekara Talya S. S., Sobolewski M., *Service-based P2P overlay network for collaborative problem solving,* Decision Support Systems, Volume 43, Issue 2, March 2007, pp. 547-568 (2007).
9. Goel S, Talya S., and Sobolewski M., *Preliminary Design Using Distributed Service-based Computing,* Proceeding of the 12th Conference on Concurrent Engineering: Research and Applications, ISPE, Inc., pp. 113-120 (2005).
10. Grand M., *Patterns in Java,* Volume 1, Wiley, ISBN: 0-471-25841-5 (1999).
11. *Inca X™ Service Browser for Jini Technology,* Available at: `http://www.incax.com/index.htm?http://www.incax.com/service-browser.htm`
12. Jini architecture specification, Version 2.1. Available at: `http://www.sun.com/software/jini/spec/jini1.2html/jini-title.html` Accessed on: March 15, 2007 (2001)
13. Khurana V., Berger M., Sobolewski M., *A Federated Grid Env. with Replication Services,* In Next Generation Concurrent Engineering, ISPE/Omnipress (2005).
14. Kolonay R. M., Sobolewski M., Tappeta R., Paradis M., Burton S. 2002, *Network-Centric MAO Environment,* The Society for Modeling and Simulation International, Westrn Multiconference, San Antonio, TX (2002)
15. Lapinski M., Sobolewski M., *Managing Notifications in a Federated S2S Environment,* International Journal of Concurrent Engineering: Research & Applications, Vol. 11, pp. 17-25 (2003).
16. McGovern J., Tyagi S., Stevens M. E., Mathew S., *Java Web Services Architecture,* Morgan Kaufmann (2003).

17. Pitt E., McNiff K., *java.rmi: The Remote Method Invocation Guide,* Addison-Wesley Professional (2001).
18. Project Rio, A Dynamic Service Architecture for Distributed Applications. Available at: `https://rio.dev.java.net/`. Accessed on: March 15, 2007.
19. Röhl P. J., Kolonay R. M., Irani R. K., Sobolewski M., Kao K. *A Federated Intelligent Product Environment,* AIAA-2000-4902, 8th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, Long Beach, CA, September 6-8 (2000).
20. Ruh W. A., Herron T., Klinker P., *IIOP Complete: Understanding CORBA and Middleware Interoperability,* Addison-Wesley (1999).
21. Sobolewski M., *Federated P2P services in CE Environments,* Advances in Concurrent Engineering, A. A. Balkema Publishers, 2002, pp. 13-22 (2002).
22. Sobolewski M., Kolonay R., *Federated Grid Computing with Interactive Service-oriented Programming,* International Journal of Concurrent Engineering: Research & Applications, Vol. 14, No 1., pp. 55-66 (2006).
23. Sobolewski M., Soorianarayanan S., Malladi-Venkata R-K. 2003, *Service-Oriented File Sharing, Proceedings of the IASTED Intl., Conference on Communications, Internet, and Information technology,* pp. 633-639, ACTA Press (2003).
24. Soorianarayanan S., Sobolewski, M., *Monitoring Federated Services in CE,* Concurrent Engineering: The Worldwide Engineering Grid, Tsinghua Press and Springer Verlag, pp. 89-95 (2004).
25. *SORCER Research Group,* Available at: `http://sorcer.cs.ttu.edu/`
26. *SORCER Research Topics,* Available at: `http://sorcer.cs.ttu.edu/theses/`
27. Sotomayor B., Childers L., *Globus® Toolkit 4: Programming Java Services,* Morgan Kaufmann (2005).
28. Thain D., Tannenbaum T., Livny M. *Condor and the Grid,* In Fran Berman, Anthony J. G. Hey, and Geo rey Fox, editors, Grid Computing: Making The Global Infrastructure a Reality. John Wiley (2003).
29. *The Service UI Project,* Available at: `http://www.artima.com/jini/serviceui/index.html`. Accessed on: July 15, 2007.
30. Waldo J., *The End of Protocols,* Available at:`http://java.sun.com/developer/technicalArticles/jini/protocols.html`. Accessed on: March 15, 2007.

Proceedings of the International Multiconference on
Computer Science and Information Technology, pp. 779 – 788
ISSN 1896-7094
© 2007 PIPS

Dependability of the Explicit DMC Algorithm for a Rectification Process

Piotr Gawkowski[1], Maciej Ławryńczuk[2], Piotr Marusak[2], Janusz Sosnowski[1], and Piotr Tatjewski [2]

[1] Institute of Computer Science
{P.Gawkowski, J. Sosnowski}@ii.pw.edu.pl
[2] Institute of Control and Computation Engineering
{M.Lawrynczuk, P.Marusak, P.Tatjewski}@ia.pw.edu.pl
Warsaw University of Technology, ul. Nowowiejska 15/19, 00-665 Warsaw, Poland

Abstract. The paper studies dependability of software implementation of the explicit DMC (Dynamic Matrix Control) Model Predictive Control (MPC) algorithm applied for a rectification column. The process with two inputs and two outputs with strong cross-couplings and significant time delays is studied. The algorithm's control law is calculated off-line. Dependability is evaluated experimentally using software implemented fault injection approach. The injected faults influence the quality of rectification process.

1 Introduction

Faults appearing during system operation may be critical for implemented applications. They can result in logical errors and application failure [1, 8]. It is particularly critical in many reactive systems (e.g. nuclear plants, satellites, aircrafts, chemical industry, medicine). Hence, an important practical issue is to evaluate dependability of software applications in the presence of faults. This paper studies the dependability of software implementation of the explicit version of Dynamic Matrix Control (DMC) Model Predictive Control (MPC) algorithm applied to a rectification column [21]. In the research the software implemented fault injector adapted to reactive applications is used [3].

Model Predictive Control is practically the only advanced control technique which have found acceptance in the process industry and is successfully applied practice [10-12, 15, 19, 20]. In the MPC algorithms the control action is calculated using a model of the process. If the model is accurate enough, performance offered by MPC algorithms can be better than the one offered by classical control algorithms, especially for processes with difficult dynamics, e.g. with significant time delay. Moreover, thanks to using the process model, the MPC algorithms contain the decoupling mechanism inside. It is very important for Multi-Input Multi-Output (MIMO) processes with strong cross-couplings. Among different MPC techniques, DMC is often applied in practice because it uses a step-response model of the process which is very easy to obtain in industry [2, 19].

The paper is structured as follows. In the next Section the explicit DMC control algorithm is described in a version for MIMO control processes. In Section 3 fault injection test-bed, experiment set-up and some new aspects of experiment conduction

according to control algorithms specificity are presented. In Section 4 results of conducted experiments are discussed. The last section summarises the paper.

2 DMC Algorithm for MIMO Processes

The distinctive feature of the MPC algorithms is that the on-line calculation of control policy considers the future behaviour of the control process many time-steps ahead. The prediction is made using a dynamic model of the control process. Typically, the future control values are chosen in such a way that the predicted behaviour of the control algorithm minimises a performance function formulated usually as follows:

$$J = \sum_{j=1}^{n_y} \sum_{i=1}^{N} \psi^j \cdot \left(\bar{y}_k^j - y_{k+i|k}^j \right)^2 + \sum_{j=1}^{n_i} \sum_{i=0}^{N_u-1} \lambda^j \cdot \left(\Delta u_{k+i|k}^j \right)^2 , \tag{1}$$

where $\bar{y}_k^j$ is a set-point value for the j^{th} output, $y_{k+i|k}^j$ is an output value for the $(k+i)^{\text{th}}$ sampling instant predicted at k^{th} sampling instant (the way of its calculation depends on a model used for prediction), $\Delta u_{k+i|k}^j$ are future changes in the manipulated variables, $\psi^j \geq 0$ and $\lambda^j \geq 0$ are weighting coefficients for the predicted control errors of the j^{th} output and for the changes of the j^{th} manipulated variable, respectively, N and N_u denote prediction and control horizons, n_y, n_i denote the number of outputs and inputs, respectively.

The performance function (1) can be rewritten in the following way:

$$J = \left(\bar{\boldsymbol{y}} - \boldsymbol{y} \right)^T \cdot \boldsymbol{\Psi} \cdot \left(\bar{\boldsymbol{y}} - \boldsymbol{y} \right) + \boldsymbol{\Delta u}^T \cdot \boldsymbol{\Lambda} \cdot \boldsymbol{\Delta u} , \tag{2}$$

where $\boldsymbol{\Psi}$ is a weighting matrix of dimensionality $n_yN \times n_yN$, $\boldsymbol{\Lambda}$ is a weighting matrix of dimensionality $n_iN_u \times n_iN_u$, $\boldsymbol{\Delta u}$ is a vector of dimensionality n_iN_u composed of the future increments of control values, $\bar{\boldsymbol{y}}$ is a vector of dimensionality n_yN composed of set–point values $\bar{y}_k^j$, $\boldsymbol{y}$ is a vector of dimensionality n_yN composed of output values $y_{k+i|k}^j$ predicted using a control process model. If the prediction is performed using a linear process model then the superposition principle can be used and the vector $\boldsymbol{y}$ can be decomposed as:

$$\boldsymbol{y} = \boldsymbol{y}^0 + \boldsymbol{G} \cdot \boldsymbol{\Delta u}, \tag{3}$$

where $\boldsymbol{G}$ is a matrix of dimensionality $n_yN \times n_iN_u$ called a dynamic matrix composed of the elements of the process step response, $\boldsymbol{y}^0$ is a vector of dimensionality n_yN called a free response because it contains elements equal to the values of the process outputs in the future in the situation if manipulated signals are frozen at the k^{th} sampling instant [11, 15, 19].

If the performance function (2) is minimised without constraints, the optimal unique solution is:

$$\boldsymbol{\Delta u} = \boldsymbol{K} \cdot \left(\bar{\boldsymbol{y}} - \boldsymbol{y}^0 \right), \tag{4}$$

where

$$\boldsymbol{K} = \left(\boldsymbol{G}^{\boldsymbol{T}} \cdot \boldsymbol{\Psi} \cdot \boldsymbol{G} + \boldsymbol{\Lambda}\right)^{-1} \boldsymbol{G}^{T} \cdot \boldsymbol{\Psi} \,. \tag{5}$$

Only the elements $\Delta u^{j}_{k|k}$ of the vector $\boldsymbol{\Delta u}$ are applied to the process and then the procedure is repeated in the next sampling instant. The step-response model of the controlled process used by the DMC algorithm is following:

$$y^{j}_{k} = \sum_{m=1}^{n_i} \sum_{i=1}^{D-1} s^{j,m}_{i} \cdot \Delta u^{m}_{k-i} + s^{j,m}_{D} \cdot u^{m}_{k-D}, \tag{6}$$

where Δu^{m}_{k} is a change in the m^{th} manipulated variable at the k^{th} sampling instant, $s^{j,m}_{i}$ ($i = 1,\ldots, D$) are step response coefficients of the controlled process describing influence of the m^{th} input on the j^{th} output, D is equal to the number of time instants after which the coefficients of the step responses can be assumed as settled, u^{m}_{k-D} is a value of the m^{th} manipulated variable at the $(k–D)^{\text{th}}$ sampling instant.

The DMC control law (4) can be formulated:

$$\begin{bmatrix} \Delta u^{1}_{k|k} \\ \Delta u^{2}_{k|k} \\ \vdots \\ \Delta u^{n_i}_{k|k} \end{bmatrix} = \boldsymbol{K}^{\boldsymbol{e}} \begin{bmatrix} \bar{y}^{1}_{k} - y^{1}_{k} \\ \bar{y}^{2}_{k} - y^{2}_{k} \\ \vdots \\ \bar{y}^{n_y}_{k} - y^{n_y}_{k} \end{bmatrix} - \sum_{j=1}^{D-1} \boldsymbol{K}^{\boldsymbol{u}}_{\boldsymbol{j}} \begin{bmatrix} \Delta u^{1}_{k-j} \\ \Delta u^{2}_{k-j} \\ \vdots \\ \Delta u^{n_i}_{k-j} \end{bmatrix}, \tag{7}$$

where K^{e} is a matrix of dimensionality $n_i \times n_y$ and K^{u}_{j}, j=1, …, D–1 are matrices of dimensionality $n_i \times n_i$ composed of controller coefficients. Because these matrices and the resulting control law are calculated off-line, the discussed MPC algorithm is named the explicit one. The detailed description of the explicit DMC algorithm derivation can be found in [14, 19].

3 Experiment Set-up

The concept of the SoftWare Implemented Fault Injector (SWIFI) is based on the software emulation of a fault during the run-time of the application under test. In this research FITS fault injector is used [4, 17]. It uses standard Win32 *Debugging API* to control the execution of the software application under tests. In the following the process controlled by the analysed control algorithm (DMC) is described, instrumentation (facilitating further analysis) of the tested application is introduced, then the fault insertion policy, and finally, applied result qualification is given.

MIMO Process Description. The process is a rectification column with two manipulated (inputs) and two controlled (outputs) variables shown in Fig. 1. It is described by the continuous-time transfer function model [14, 21] (time constants in minutes):

$$\begin{bmatrix} Y^1(s) \\ Y^2(s) \end{bmatrix} = \begin{bmatrix} \dfrac{12,8}{16,7s+1} & \dfrac{-18,9e^{-4s}}{21,0s+1} \\ \dfrac{6,6e^{-8s}}{10,9s+1} & \dfrac{-19,4e^{-4s}}{14,4s+1} \end{bmatrix} \cdot \begin{bmatrix} U^1(s) \\ U^2(s) \end{bmatrix} + \begin{bmatrix} \dfrac{3,8e^{-4s}}{14,9s+1} \\ \dfrac{4,9}{13,2s+1} \end{bmatrix} \cdot U^3(s), \tag{8}$$

where the controlled variables are:y^1 – methanol concentration in the distillate (the top product), y^2 – methanol concentration in the effluent (the bottom product), the manipulated variables are: u^1 – flow rate of the reflux, u^2 – flow rate of the steam into a boiler, u^3 is feed flow rate (a disturbance). All process variables are scaled.

For the considered rectification process the explicit DMC algorithm is designed: the sampling period T_p=1 min is assumed, the dynamics horizon is equal to the prediction horizon D=N=100, the control horizon N_u=50, the values of weighting coefficients are: ψ^1=ψ^2=1, λ^1=λ^2=10. The simulation horizon is 300 discrete time-steps. Structure of the control system with the explicit DMC algorithm is shown in Fig. 2.

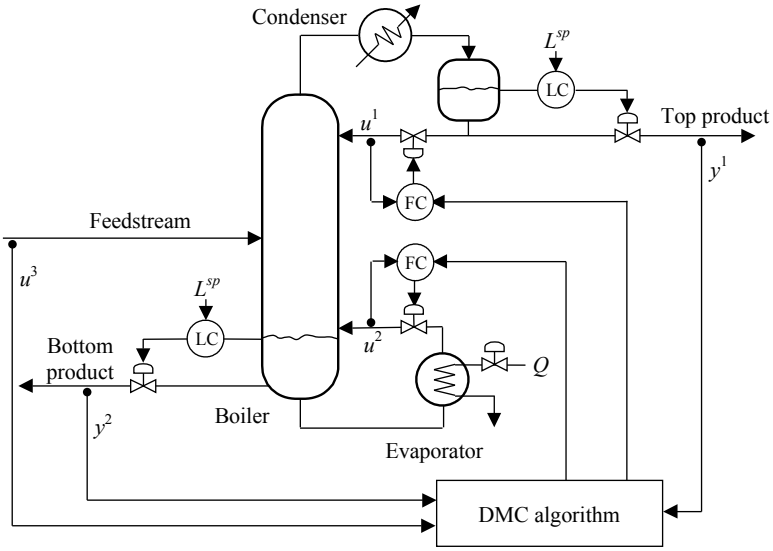


Fig. 1. Rectification column control system structure

Code instrumentation. FITS disturbs directly the tested application only within so-called testing areas [3]. Testing areas limit the scope of disturbances only to the selected parts of the application. Here, the code of the controlled process model is added. The parts of the tested application disturbed during the experiments (dashed box) as well as process models (not disturbed) are marked in Fig. 2. The tested

application is also instrumented to send some measures (e.g. related to internal variables values, output signal deviations) to the fault injector using user-defined messages (collected by FITS) [3].

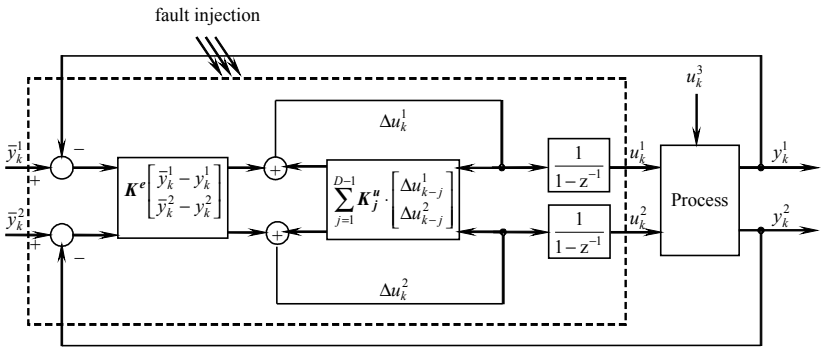


Fig. 2. Structure of the control system with the explicit DMC controller

Fault injection policy. FITS simulates faults by disturbing the running application. In this study the single bit-flip faults within CPU and FPU registers, application's data and machine instruction code are considered. Faults are injected pseudorandomly in time (program execution) and space (bit position within disturbed resource, distribution over application's memory). Such a fault model well mimics Single Event Upset (SEU) effects [7, 8, 16].

Qualification of experimental results. The correctness qualification of results produced by the application under test is more complicated in case of applications related to control systems [16] than in case of simple calculation-oriented ones. Control algorithms require complex analysis of the controlled process behaviour. The standard factor ISE (Integrated Sum of Squared Errors) is used as a measure of result (y^1, y^2) correctness. The reference ISE value (obtained during referential execution – non-faulty) is 2.53 (due to delayed response and feed stream disturbances). The whole experiment is conducted by FITS automatically. At the end of the experiment synthetic (aggregated) results for each fault location are given. In general, 4 classes of test results are distinguished:

- C: correct behaviour (ISE<10),
- INC: incorrect (unacceptable) behaviour (ISE≥10),
- S: test terminated by the system due to un-handled exception,
- T: timed-out test.

Analysis of fault effects requires detailed information upon the faults injected and the application behaviour. FITS provides details about every test (simulated fault injection). Hence, manual replay of the whole test execution can be done. Moreover, all the events and user messages occurring during the test are recorded. The tested application is instrumented to save its outputs (here simulation results, i.e. a set of control signals in subsequent sampling instants) into separate files for each test (file names are managed by FITS). This gives a possibility for post-experiment analysis of fault effects in the correlation with the injected fault and observed behaviour for each test.

4 Simulation Results

Implementation of the control algorithm takes 124 machine instructions (405 bytes of the static code). The algorithm needs execution of 1020000 instructions for the whole simulation horizon (300 discrete sampling instants). As the static and dynamic profile is different, the distribution of mnemonics in the static code as well as in the executed stream (dynamic) is presented in Fig. 3.

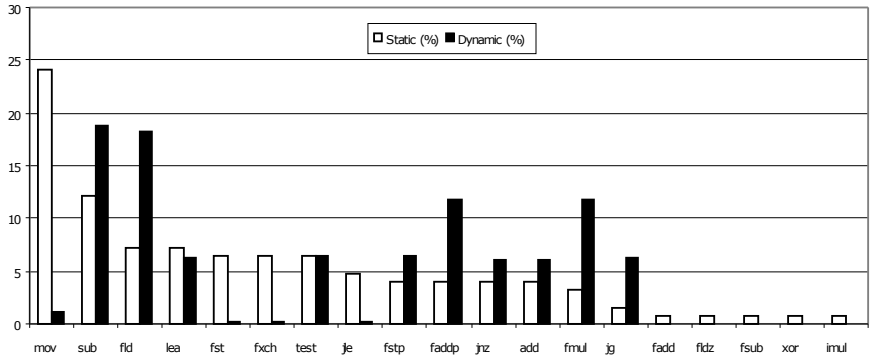


Fig. 3. Distribution of DMC mnemonics in static and dynamic profile

It is worth noting that floating point instructions contains 38% of the code in the static code while dynamically they are executed in 54,8% of time. Moreover, instructions organising computational loops in the DMC implementation (*sub*, *test*, *add*, *jnz*, *jg*) take another 38%. Hence, the application is strongly computational with high degree of FPU utilisation. Nevertheless, the instruction set used is rather limited. On the other hand, the activity ratio for CPU resources is high (98, 94, 80, 98, 97, 81 % for EAX, EBX, ECX, EDX, ESI and EDI, respectively) [16].

Faults in the CPU and FPU registers, data area of the application, executed instruction stream, and static code image are considered. For each fault location approximately 1000 disturbed executions are investigated (single fault injected in each application execution). The summary of results (according to categories described in Section 3) is presented in Fig. 4. As the algorithm uses many parameters (400) it is very robust to fault located in the data area – there are only few data memory locations critical for the algorithm – most of the data correspond to the algorithm's parameters. The high degree of FPU robustness could be astonishing. Past experience shows that the FPU is rarely used hard [8] (e.g. only few FPU stack locations used simultaneously). This results in overall low fault sensitivity of the FPU. Nevertheless, there are some very sensitive locations within the FPU (e.g. control registers).

The most fault sensitive resource of the DMC controller is its code. Fortunately, there are software techniques (e.g. exception handling, duplication of critical data and code) that can be applied at the source code level to provide fault robustness [5-7,9]. They have already proved their advantages; nevertheless, their effectiveness in the considered algorithm is going to be investigated in the further research. One can expect, that in the DMC case some simplifications in the fault hardening implementa-

tions can be applied without noticeable aggravation of dependability and performance (i.e. rare errors on the controlled process inputs are not critical for its behaviour).

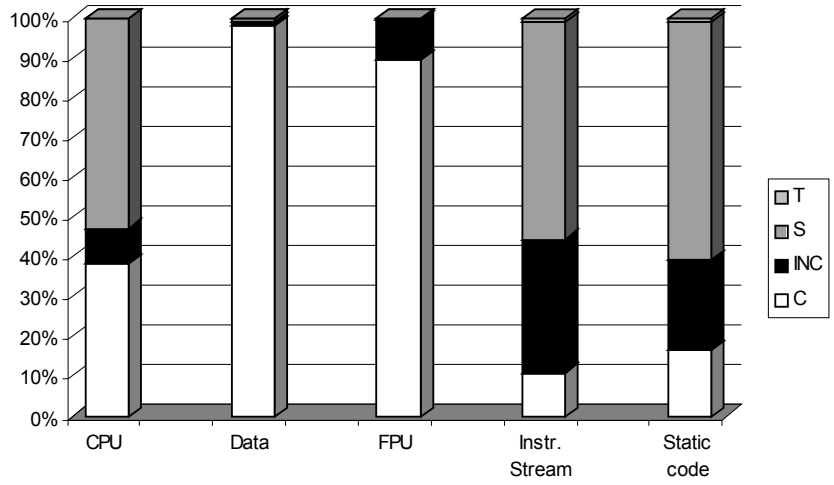


Fig. 4. Experimental results

Analysing fault susceptibility it is worth correlating the observed effects (simulated process behaviour) with the injected fault details in accordance to the source and machine code of the disturbed DMC. Fig. 5 and 6 present plots of the application outputs (y^1, y^2 – left plots and u^1, u^2 – right plots) over the DMC iteration number in case of sample fault disturbed executions. For reference the undisturbed simulation results are shown, i.e. the golden run (solid lines). The simulation scenario is as follows (solid lines). At the beginning, the process is driven to a given set-point. Then, at sampling instant 30 the change in the feed stream flow rate (u^3) is introduced (from 0 to 0.1). Another change in u^3 is made at the instant 140 (from 0.1 to –0.05).

In the case considered in Fig 5, the fault is injected at the sampling instant 28. It results in the change of the *faddp* instruction into the *fmulp* (operands remained unchanged). The instruction disturbed is used to calculate the *du1* variable of the DMC algorithm (corresponding to the Δu_k^1 in Fig. 2). As a result the source code statement `du1+=r1[i]*vektup[i]` is changed to `du1*=r1[i]*vektup[i]`. The result of this disturbance varies on the control signal and process states. For instance, the considered fault injected at the 28th sampling instant results in ISE=3.39 (Fig. 5a), at the 101st sampling instant in ISE=4.25 (Fig. 5b), at the 224th – ISE=2.53 (Fig. 5c) (the same as the reference ISE value). Hence, it disturbs the top product composition.

More critical situation is illustrated in Fig. 6. Single bit inversion (at 61st sampling instant) within the same instruction as described above destabilises the process. In this case the instruction mnemonic remain unchanged while its first operand changed from *st(2)* to *st(0)*. Observed ISE is 4.40.

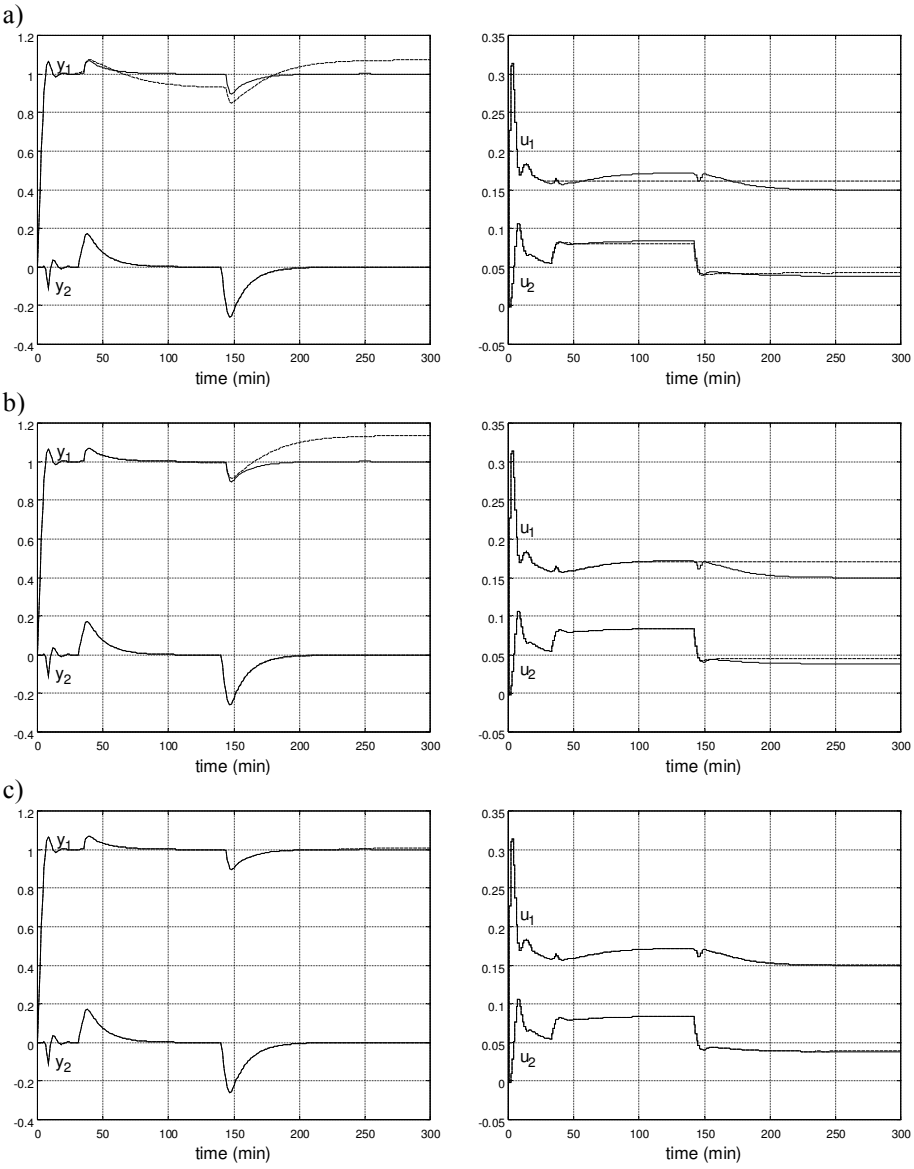


Fig. 5. Single bit inversion within the *faddp* instruction disturbs the top product composition. Faults injected at sampling instants: a) 28th, ISE=3.39, b) 101st, ISE=4.25, c) 224th ISE=2.53; golden run responses – solid line, fault injected responses – dashed line.

It is worth noting that overwhelming majority of incorrect behaviour relates to very high ISE values (higher than 1000). It means that the control values errors cumulates. At the other hand it gives the possibility to easily detect such big deviations using additional diagnostic subroutines.

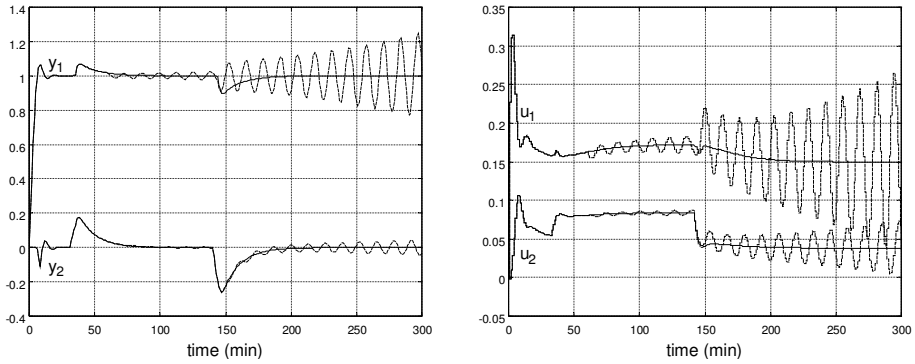


Fig. 6. Single bit inversion within the *faddp* instruction destabilises the process. Fault injected at the 44th sampling instant, ISE=4.40; golden run responses – solid line, fault injected responses – dashed line.

5 Conclusion

The paper studies dependability of software implementation of the explicit DMC algorithm applied to a rectification process. The process has two manipulated variables, two controlled variables and significant time-delays. Dependability is examined using software implemented fault injector.

Results of the experiments carried out clearly indicate that the well-known and widely used formulation of the DMC algorithm is susceptible to software faults. It is particularly important in case of industrial processes such as the considered rectification column because faults are likely to lead to undesirable behaviour of the process. More specifically, in the least difficult situation a fault can disturb composition of the products whereas in the most dangerous case it can destabilise the system. Technological and financial consequences of faults are of fundamental importance. Energy losses and unacceptable compositions of the products (which means that the product cannot be sold) are just two examples of such situations.

In order to increase fault robustness of software implementation of the DMC algorithm, a few techniques can be applied. In particular, current research embrace DMC algorithm based fault detection. In particular, the most critical situations (technologically unacceptable control values leading to instability) may be detected and handled by an additional diagnostic subroutine.

Acknowledgement. This work was supported by a grant from the dean of the Faculty of Electronics and Information Technology, Warsaw University of Technology.

References

1. Benso A., Prinetto P.: *Fault injection techniques and tools for embedded systems reliability evaluation,* Kluwer Academic Publishers. (2003).
2. Cutler R., Ramaker B.: *Dynamic matrix control – a computer control algorithm,* AIChE National Meeting, Houston. (1979).
3. Gawkowski P., Sosnowski J.: *Experiences with software implemented fault injection,* Int. Conf. on Architecture of Computing Systems. VDE Verlag GMBH. (2007) 73–80.
4. Gawkowski P., Sosnowski J.: *Analysing system susceptibility to faults with simulation tools,* Annales UMCS Informatica AI. **4** (2006) 123–134.
5. Gawkowski P., Sosnowski J.: *Software implemented fault detection and fault tolerance mechanisms – part I: Concepts and algorithms,* Kwartalnik Elektroniki i Telekomunikacji. **51** (2005) 291–303.
6. Gawkowski P., Sosnowski J.: *Software implemented fault detection and fault tolerance mechanisms -- part II: Experimental evaluation of error coverage,* Kwartalnik Elektroniki i Telekomunikacji. **51** (2005) 495–508.
7. Gawkowski P., Sosnowski J., Radko B.: *Analyzing the effectiveness of fault hardening procedures,* 11th IEEE Int. On-Line Testing Symposium. (2005) 14–19.
8. Gawkowski P., Sosnowski J.: *Dependability evaluation with fault injection experiments,* IEICE Transactions on Information & System. **E86-D** (2003) 2642–2649.
9. Gawkowski P., Sosnowski J.: *Experimental Validation Of Fault Detection And Fault Tolerance Mechanisms,* 7th IEEE Int. Workshop on High Level Design Validation And Test. Cannes (2002).
10. Ławryńczuk M., Marusak M., Tatjewski P.: *Multilayer and integrated structures for predictive control and economic optimisation,* IFAC Symposium on Large Scale Systems, Gdańsk (2007), accepted for publication.
11. Maciejowski J. M.: *Predictive control with constraints,* Prentice Hall. Harlow. (2002).
12. Morari M., Lee J. H.: *Model predictive control: past, present and future,* Computers and Chemical Engineering. 23 (1999) 667–682.
13. Qin S. J., Badgwell T. A.: *A survey of industrial model predictive control technology,* Control Engineering Practice. 11 (2003) 733–764.
14. Pułaczewski J.: *Multidimensional DMC algorithm,* Report of ICCE WUT no 98–11, Warsaw (1998), in Polish.
15. Rossiter J. A.: *Model-based predictive control,* CRC Press, Boca Raton. (2003)
16. Sosnowski J., Gawkowski P., Lesiak A.: *Fault injection stress strategies in dependability analysis,* Control and Cybernetics. **33** (2005) 679–699.
17. Sosnowski J., Lesiak A., Gawkowski P., Włodawiec P.: *Software implemented fault inserters,* IFAC Workshop on Programmable Devices and Systems. Ostrava (2003) 293–298.
18. Sosnowski J., Gawkowski P., Lesiak A.: *Fault injection stress strategies,* 4th IEEE LATW 2003 Workshop. (2003) 258-263.
19. Tatjewski P.: *Advanced control of industrial processes, Structures and algorithms,* Springer. London. (2007).
20. Tatjewski P., Ławryńczuk M., Marusak P.: *Linking nonlinear steady-state and target set-point optimisation for model predictive control,* IEE International Control Conference ICC 2006, Glasgow (2006), CD–ROM.
21. Wood R. K., Berry M. W.: *Terminal Composition Control of a Binary Distillation Column,* Chememical Engineering Science **28** (1973) 1707–1717.

Proceedings of the International Multiconference on
Computer Science and Information Technology, pp. 789 – 798
ISSN 1896-7094
© 2007 PIPS

On communication management as a key element of successful IT program

Rafał W. Cegielski[1], Jarosław Chudziak[2], Joanna Meyer[1]

[1]Prokom Software S.A. ul. Grójecka 127
02-383 Warsaw, Poland
{cegielskir, meyerj}@prokom.pl
[2]Institute of Computer Science
Warsaw University of Technology
ul. Nowowiejska 15/19
00-665 Warsaw, Poland
j.chudziak@ii.pw.edu.pl

Abstract. In this paper we discuss the role of communication management in each phase of the large implementation of IT solutions. It is well known that the majority of project failures in the recent years are not due to lack of technical competency of the supplying team, but rather because of not well enough prepared and executed communication between various participants of the implementation process. In the paper we present various dimensions of communication process, its role and objectives, as well as, main participants and several modern tools and channels, which can be used in the process.

1. Introduction

The process of project/program management, as well as, various techniques of software engineering is very well defined and described in both academic literature and practical textbooks [1, 2, 5]. For many years project management was very often focused on scope, time and resource management with various additional activities like risk, change management, ROI etc. But along with many years of observation of IT projects of various type and magnitude, we still see a significant amount of inefficiency and ineffectiveness in project management caused by lack of proper understanding of the role which well defined and then properly executed communication stream can play.

The goal of our paper is to put communication management in the right balance with other activities of the "classical" project management approach [3, 7]. Communication management has become more and more important nowadays with all new technologies and implementation techniques which make development and then deployment of new solutions a very communication intensive process. It is also very closely connected to all activities which are done in parallel to the main "technology" stream like for example: business model change management, knowledge transfer, as well as, such a "soft" area like change of the company culture.

We treat communication management as a strategic part of project/program management of any large IT initiative, in particular in times of booming solutions like

corporate intranets, intensive internal corporate communication management and IT governance solutions which migrate decisions towards "business" stakeholders.

In the next sections, we present some of the elements of communication management model prepared for large implementation of IT projects. Based on reviewed and verified data from implemented projects, the proposed model assumes the complete project lifecycle for communication management process. It starts from the assessment phase and ends up with post-implementation feedback review. We also describe examples of key elements of the model with selected predefined templates for:

- Scope and objectives definition.
- Process definition.
- Identification of participants/stakeholders with their roles and responsibilities.
- Definition of potential communication streams and their supporting tools, performance measures with indication how to use them and monitor them during project execution.

2. Communication management in the context of large IT implementation programs

2.1. Dimensions of communication management

In many cases, large IT implementation programs do not achieve planned benefits on the expected level or even end up with a failure with very significant costs. Some of these cases are very spectacular and are being commented in academic and business publications.

One of the key reasons is that while the business and technical related aspects of the implementation are planned and managed with appropriate attention, the change management dimension is very often underestimated and thus inadequately treated.

Time horizon of large IT implementation programs takes several months or even years, but the path for change management activities is even more challenging and longer. For example, an IT system may be implemented in 6-9 months, the business processes redesigned and implemented in 3 months. But it will take over 12 months for the end users to learn the new processes, new system functionalities and accept them. If the end users will not accept the new system, the planned business benefits will never be fully realized.

One of the critical components of the change management process is communication management. The role of communication management in a large IT implementation can be considered in the following dimensions:

Managing stakeholder perceptions from awareness to acceptance. With a large IT implementation, the change management process begins before the actual change is implemented and ends far after the actual change occurred. Every IT project or program has different stakeholders – groups of people that have different views, interests and impact on the implementation. All these perceptions impact the readiness

for change of stakeholders and should be analyzed and assessed. Communication management activities must ensure that the end users know why they have to change, what benefits the change will bring to them, what are the potential barriers and problems with the implementation. Communication management establishes a shared vision of change that the IT program will bring. It provides all information about what is happening with the project/program – now and in the future.

Development of new skills and competences. Large IT implementation usually results with a new system in place that will require new skills from the end users. It is critical to provide appropriate training for all types of users. The trainings should be delivered with tools, that are tailored to needs of different stakeholders. For example, direct end users should be trained in detail about the new ERP system, while sales representatives that don't have direct contact with the solution, should have enough knowledge to explain to their clients how this new system will affect cooperation with them. Apart from traditional classroom trainings, the end users should have direct and online access to different tools providing the required knowledge that will reinforce and enhance the new set of skills e.g. computer-based-training materials, manuals, presentations etc.

Knowledge management. IT implementation is a continuous learning process. As the end users are starting to utilize their new skills in practice, their own experience and recommendations are being developed. Communication management should provide the right platform for sharing knowledge and experience among all stakeholders [6].

Crisis management. Due to their complexity, large IT implementations are high risk initiatives. The nature of programs, and uncertainty associated with them, mean that inherent risks are always a threat to success. Program and project managers must be proactive in identifying potential risks and ensuring that appropriate mitigating actions are taken before these risks impact the cost, timescales or quality of project deliverables [3, 6]. Planning appropriate communication actions in case a risk turns into an issue or crisis is one of the key areas of risk mitigation.

Communication within the project team. It is vital for the success of the program/project to provide the team members with communication tools and platforms supporting their activities in the areas of planning, progress monitoring, coordination, risk management, benefits tracking and others [3].

Feedback from stakeholders. Communication during large IT program implementation is a two way process during the whole cycle. Information requirements of the stakeholders are researched and analyzed since inception of the initiative. During the project, stakeholder perceptions and acceptance of change is being monitored and potential improvements and modifications are being implemented. Another extremely important aspect is to "listen" to the stakeholders – their view on the system implementation, perceived problems, comments, recommendations and fears about change. This means that appropriate communication channels and tools that enable collection of such information from the stakeholders should be established.

Dimensions of communication management are summarized on the diagram below.

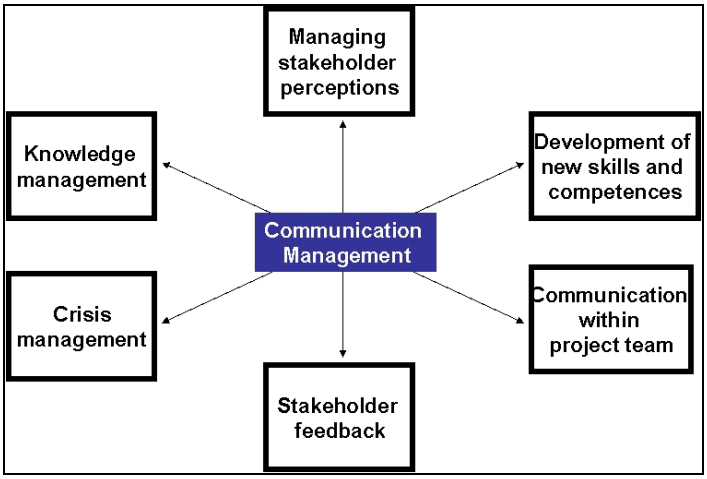


Fig. 1. Dimensions of communication management in large IT implementation programs

2.2. Communication management process

Communication management activities in a large IT implementation program can be structured into a set of interdependent phases:

Assessment. Before the actual change starts, all the requirements for communication activities have to be analyzed and assessed. They are strictly related to the scope and depth of the planned implementation as well as to readiness for change in the organization. All stakeholder groups have to be identified and their perceptions and impact on the planned change assessed. This phase is critical for the success of communication management. Poorly analyzed requirements may cause major issues during the implementation of the IT solution.

Planning. At this stage a detailed communication plan has to be developed. Content, communication channels, tools and feedback mechanisms have to be designed in accordance with the assessment results. All planned activities have to fulfill the requirements of identified stakeholder groups.

Development. After planning is commenced, an infrastructure supporting all the designed tools has to be implemented. Depending on the level of sophistication of the designed tools and communication channels, this may be a demanding task in terms of time, budget and quality. Development and implementation costs of the infrastructure shouldn't exceed the expected benefits. It is extremely important, that all channels and tools are tested before execution – failure may cause mistrust to the overall communication process and become a barrier during implementation.

Execution. All activities are executed according to dates or frequencies defined in the communication plan. Before execution all content must be carefully inspected so only up to date information is sent to the stakeholders

Feedback. All communication activities should be monitored, assessed and appropriate actions undertaken if required. Feedback from the stakeholders may require additional analysis, changes in the communication plan, improvement of the

tools and channels used. Additionally effectiveness of the communication channels is measured. The results should be analyzed in detail and implemented in a way that will not disturb the change management process.

The diagram below summarizes the communication management process.



Fig. 2. The communication management process in large IT implementation programs

Usually, communication management is in the area of the program/project manager responsibilities. In many cases, due to his wide range of responsibilities and heavy workload, communication management falls into the lower priorities on his agenda. Successful communication management requires an additional role on the program/project team – Communication Manager. His responsibilities cover the following areas:

- Identification of stakeholder groups.
- Analysis of the information needs of the stakeholders.
- Development of the communication plan.
- Acceptance of the content that will be delivered to the stakeholders.
- Supervisory of execution of the communication activities.
- Supervisory of maintenance and administration of the communication channels and tools.
- Monitoring, measuring and analyzing feedback from the communication management activities and deciding on improvements and modifications.

Depending on the scale of the IT implementation program, the Communication Manager has to be supported by a team specialized in content creation, moderators etc.

3. Objectives

The objectives of communication management are linked with business and IT implementation program objectives [3]. They should be tailored to specific requirements and milestones of the project phases. Each phase has different requirements and perceptions of the stakeholders. Communication management activities have to be planned in accordance with them. The objectives should be linked with key performance indicators (KPI) that will be measured to assess whether the planned change management activities are fulfilling the requirements.
Decomposition of objectives into project cycle and communication management is presented on the diagram below.

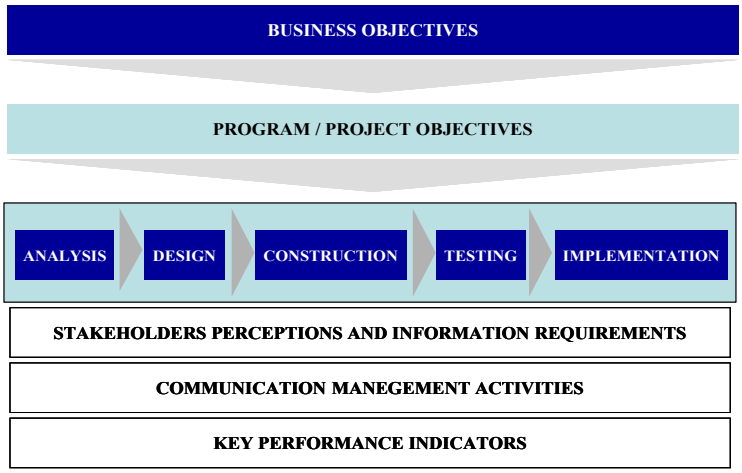


Fig. 3. Decomposition of business objectives into project cycle and communication management

Objectives of communication management in a large IT implementation consist of the following elements:

- Establishment of a communication platform with stakeholders regarding the change process.
- Development of effective communication tools required for a successful implementation of the IT solution.
- Support for the program managers in elimination of risks associated with the implementation.
- Build acceptance for the implemented solution among stakeholders.
- Preparation of end users to effectively work with the new solution.

Communication management objectives presented above are the most common to large IT implementation programs and for each program/project they depend on specific scope, business objectives, environment etc.

4. Key performance indicators

For each IT program/project a specific set of key performance indicators (KPI) should be defined. They have to be related to each communication dimension, for example:

- Understanding of new business solution.
- Knowledge and understanding of new processes, roles and responsibilities.
- Acceptance of new technology solutions.
- Maturity level of new competency and skills.

Each KPI has to be measured against a common scale consisting of 5 levels of excellence:

- Awareness.
- Understanding.

- Simple skills.
- Advanced skills.
- Leader of change.

A curve describing maturity levels and its growth over the period (phases) of project development is defined and monitored for each KPI. Target levels of each KPI for every group of stakeholders are defined in the communication strategy. The diagram below summarizes the transformation of communication dimensions into KPI's and then into maturity levels.

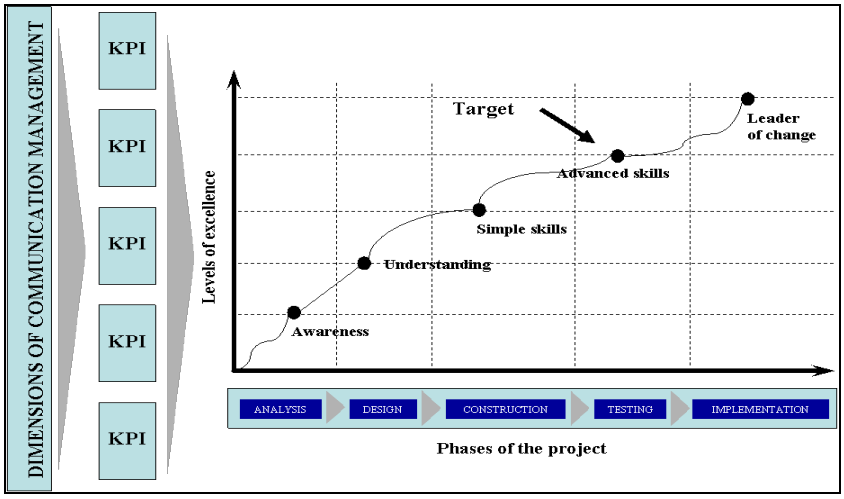


Fig. 4. Transformation of communications management dimensions into KPI's and maturity models

5. Communication management tools

The range of tools that can be used in large IT implementation programs is related to the information requirements of different stakeholder groups. Every group of stakeholders has different information needs that require specific information content, intensity, frequency and channels to be used.

5.1. Key stakeholders

The characteristics of the stakeholder group differ on every project. However four generic groups of stakeholders can be defined:

Project team. This is the group of employees that is directly involved in daily operations of the IT implementation program. This group is relatively small in quantity in comparison to others and requires the most intensive information which

consists of project progress, risks, problems, change requests, technical information required to project implementation, access to documentation.

Employees cooperating with the project team. This group consists of all employees of the company that are supporting the project team during implementation: delivering business and technical information, coordinating implementation activities within their departments, giving feedback to the project deliverables. Communication to this group covers information related to project progress and system implementation within the area of cooperation with the project team.

End users. This group is critical in terms of IT implementation success. Information requirements of this group are focused around knowledge about the new solution as well as around the benefits it is offering. Communication to end users must ensure their acceptance of the newly implemented system.

All employees. This is the largest group of stakeholders. In the most critical cases, large IT implementation programs affect all employees of the company. Usually, the information requirements of this group are focused around awareness of the functionality and benefits it brings to the organization.

The key stakeholder groups are summarized on the diagram below.

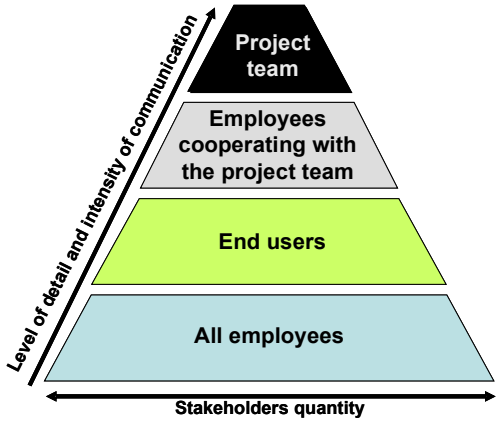


Fig. 5. Typical stakeholder groups in a large IT implementation project [7]

5.2. Communication tools

Communication tools used during large IT implementations programs are tailored to the specific project and information requirements of the identified stakeholder groups.

Table 1. The most frequently used communication tools [6]

Tool	Effectiveness	Coverage
1 to 1 meetings	High	Low
Workshops	High	Medium
Large presentations	Medium	High
Cascade briefings	Medium	High
Newsletters	Low	High
Intranet Portals	High	High
Moderated discussion groups	Medium	High
Articles	Medium	High
Video	Medium	Medium
Formal letters	Medium	High
Frequently Asked Questions (FAQ)	Low	High
Internal call centre	Low	Medium
Interactive Voice Response messages	Low	Medium
Suggestion box	Low	Low
Email	Medium	High
Notice boards	Low	Medium

Effectiveness of the communication tools depends on several factors including the quality of content provided by the tool, frequency of updates, reliability of the tool, scope of project, characteristics of the stakeholder groups.

5.3. Management dashboard

For the visualization and basic browsing activities, a standard business intelligence tools or predefined balance scorecard solutions can be used. They can present target levels and actual figures on the management dashboard. They can be available to all authorized actors of the process.

As a part of communication management solution infrastructure, the firm should also consider creation of knowledge warehouse with findings/"good practices" collected during projects that were taking place. It can be used as a source of information of successfully applied communication methods and tools for different kind of IT engagements. They should be coherent with company culture and its readiness for change (planning and execution).

6. Conclusions

The scope of communication management in large IT implementation programs covers areas from support to the communication within the project team through managing stakeholder perceptions and knowledge management to monitoring and assessment of stakeholder feedback. The objectives of communication management

are linked with business and program/project objectives and are focused around building acceptance for the implemented solution among stakeholders, empowerment of end users to effectively work with the newly implemented solution and development of an effective communication platform for a successful IT implementation.

The communication management process starts with stakeholder analysis which is critical for the success of all communication activities and is followed by planning, infrastructure development, execution of communication activities and ends with feedback.

The success of IT implementation depends strongly on the involvement of executive and senior management of the company in communication activities which usually are at the responsibility and supervision of the program/project manager. However to ensure the success, communication management has to be at the responsibility of a dedicated person who has the appropriate authorization and position in the program/project and as well as in the organization.

With development of electronic media platforms, the set of communication tools and channels is very wide. A portfolio of tools and channels has to be tailored to the specific needs of stakeholders, scope of the implementation as well as to the depth of the change process conducted in the program/project.

References

1. Kerzner H., *Advanced project management,* John Wiley & Sons, Inc. (2004).
2. Weill P., Ross J. W., *IT Governance,* Harvard Business Press (2004).
3. *A Guide to the Project Management Body of Knowledge (PMBOK Guides),* Project Management Institute, 3rd edition, (2000).
4. Reich B. H., Benbasat I. *Measuring the linkage between business and information technology objectives,* Strategic Information Systems, 2nd edition, Butterworth-Heinemann (1999), pp. 295-329.
5. Applegate L. M., McFarlan W. F., McKenney J. L. *Corporate Information Systems Management,* 5th edition, McGraw-Hill (2002).
6. Cegielski R. W., Chudziak J., Meyer J., *Communication management process in IT projects,* Prokom internal report (2006).
7. Allen C. Amason at al, *Human Side of Project Leadership,* Project Management Institute (2007).

Proceedings of the International Multiconference on
Computer Science and Information Technology pp. 799–808
ISSN 1896-7094
© 2007 PIPS

Shared Ontologies to Increase Systems Interoperatibiliy in University Institutions

Richard Gil, Ana María Borges, and Leonardo Contreras

Department of Processes and Systems,
Simón Bolívar University, Caracas, Venezuela
{rgil,aborges,leocon}@usb.ve

Abstract. The increasing demand of quality information has forced that the methodological resources of system engineering come perfecting gradually. The practical materialization of diverse concepts and relations, are requiring that the semantic technology and ontological engineering obtain all their advance and potential. The University Institutions must be at the top of these findings. It is the purpose of this paper to show a first version of ontological integration of some subsystem models that interact inwards and outside of this type of institutions. These subsystems were selected intentionally of previous studies. The Protégé-PROMPT tool will be used as resource for the integration of some experimental ontology of such subsystems, using as methodological reference a framework that estimates an approach of shared terminology to be able to integrate in a compatible ontology. As result a method which allows obtaining a shared ontology is proposed.

Key words: Interoperability, Ontologies Integration, Shared Ontology, Protégé-PROMPT, University Institutions, Methodology.

1 Introduction

Obtain interoperability and systems integration is obvious an ambitious goal, an extended technical literature on this subject demonstrates the extent of it. Numerous papers have being published from the theory to the practice-experimental [1–3]. These papers give many current contributions in the scope of the ontologies. Most of them trend to smooth the way to implement more and better diverse information systems toward the desired integration. In this sense, and looking for methodological resources to approach viable solutions in our specific cases of study—University Institutions—it is required to emphasize the efforts of investigators to implement and to describe tools toward integration [4, 1, 5, 6]. Different authors use different technologies, among them, mathematical or algebraic formalization, algorithms, network-neuronal, rules of inference, heuristic, analysis of graphs, etc., that have enriched the existing alternatives to approach some of the problem linked with the heterogeneity of systems. Adopting these

technologies appropriately, under a concept of "semantic protocol", it can make flexible the heterogeneous systems of such organizations, taking care of their own dynamic of its internal changes (structural or of processes) reacting in prospective as response to the external and continuous changes of a globalize world. In addition with the mentioned "desirable protocol" it would be allowed to conserve the necessary stability for the planned interchange between the constituent subsystems that always will be influenced by the continuous adjustments in the same trajectory (natural evolution) of the original dominion.

For this study, variants to a similar methodology to obtain interoperability proposed by Stuckenschmidth y van Harmelen [7] will be made. It is originally support on a frame of reference and methods to create shared ontologies. In this research we take three ontologies of subsystems of University Institutions that are being developing now, to ontologically integrated using a tool of most outstanding of technical Literature [8, 9], as Protégé-PROMPT. This methodological variant, cradle in this appropriate experimental area of the case of study, will allow to validate a practical way for the possibility of increasing interoperability between different dominions for diverse systems.

2 Systems Interoperability

Several previous studies as the one by Obrst [10] clarifies better the terms of systems integration and interoperability. System integration is more related to the source of data; systems interoperability with software applications and their levels of aggregation (Fig. 3, page 368 cited, shows the dimensions of interoperability and integration).

By the increasing necessity of semantic interoperability intra and inter systems and also within it and with other organizations, in this research it is considered pertinent to focus on recent tendencies of systems integration, known as Cooperative Information Systems suggested by Antonios and van Harmelen [11]:

"Cooperative Information Systems are servicing a diverse mix of demands characterized by "Content-Community-Commerce". These demands are originating in current trends for "off-the-shelf", software solutions, such as enterprise resource planning and e-commerce systems".

The main challenges and outstanding potential of application in the cooperative information systems are:

1. Knowledge Management (organized validated, consulted knowledge, etc.).
2. Electronic commerce Business-to-Client type (B2C, based on "intelligent agents").
3. Electronic commerce Business-to-Business type (B2B, abstraction of dominions using "agents").
4. Customized agents (agreed to own interests and preferences, and memorized by "agents").
5. Technology of Web semantic:
   a. Explicit Meta-data: to capture data meaning.

b. Ontologies (classes, relations, restrictions, logic relations and subgroup relations).

c. Logic (to solve logics of predicate and rules to support validation and consistency).

d. Agents (they work independently and with prediction to support decisions).

All these trends will be addressed in the sub-dominions of the subsystems considered to give a prospective character to the proposals.

Many factors affect the application interoperability, since not only it tries itself to solve the increasing of the organization internal information requirements, but that is desired to be able to integrate it with external information of other organizations (B2B, Business-to-Government (B2G)), or with those of the different users and/or clients (B2C, Government-to-Client (G2C)). The integration must be dynamic, flexible and transparent (intelligent agents) and in addition it will be customized (personal agents), based on common or determined rules (semantic logics and restrictions).

Vernardat [12] emphasizes the following:

"The systems integration consists of arming the different parts of a system guaranteeing at the same time the compatibility of the assembly as well as the good operation of the complete system.

... the interoperability of the systems is an average one to obtain integration, based on a weak connection between the parts of the system, but seated in the capacity of the parts to communicate among them to accede and to resort to its functionalities."

"... The central paper of the models (based on ontologies) in the development of the interoperability capacities influences remarkably in obtaining the quality, since the processes of engineering of the interoperability of applications, are the platform of flows of the models."

It is for that reason that this proposal try to approach ontological integration as a methodological strategy that allows to obtain system interoperability [13, 10], and to experiment with specific cases of dominions of subsystems oriented to satisfy information services at University Institutions [14–16].

3 Integration Using Ontologies

Review of the appropriate interpretation of the term ontological integration and its strategies or methods to obtain it, such as alignment, merging, articulation, mapping, fusion, integration, morphism and others, will be tried to adopt a definition that clarify the meaning of it.

Kalfoglou and Schrlemmer [5] affirm that usually S (of O) is a mathematical structure, that can usually consist of a set of concepts (classes) keeping certain order and symbols whose arguments are defined by the concepts and their hierarchies; in addition they define a total and partial ontological transformation of the following form:

"A total ontological mapping from $O_1 = (S_1, A_1)$ to $O_2 = (S_2, A_2)$ is a morphism $f : S_1 \rightarrow S_2$ of ontological signature, such that $A_2| = f(A_1)$, i.e. all interpretations that satisfy O_2's axioms also satisfy O_1's translated axioms."

"In order to accommodate a weaker notion of ontology mapping, we will say that there is a partial ontology mapping from $O_1 = (S_1, A_1)$ to $O_2 = (S_2, A_2)$ if there exists a sub-ontology $O'_1 = (S'_1, A'_1)$ ($S'_1 \subseteq S_1$ and $A'_1 \subseteq A_1$) such that there is a total mapping from O'_1 to O_2."

There are several ontologies integration "domains". They can be classified according to the traditional approach, as well as the emergent approach. From the traditional approach the ontologies evolutions is classified by the emergent approach as the share information between pairs (P2P), the schemes integration as the intelligent communication agents, the catalogue integration as the composition of Web services, and the data integration as the answers to questions on the Web (query).

There are other classifications which besides to include the techniques of ontologies match based on schemes, that allow to group the techniques arranged by the level of the match technique from syntactic to semantic through the structural. They considered the desegregation level, from the ontology-element-level to the structure-or morphology-level.

The Problem of the ontological mapping can be because in the phase of conceptualization, the semantics of the dominion will be encoded in different ways [17]. The potential problems include:

- Different ontologies may use different names to represent the same concept, or the same name to represent different concept.
- Different ontologies may use different values to represent the same concept.
- The same concept may be represented at the class level in one ontology and at the instance level in another one.
- Different ontologies may use different structural representations of concepts.

The problems involved in mapping ontologies have been tackled in three ways:

- Manual approach: similar concepts are identified by human experts and mappings by hand.
- Semi-automated approach: to automate some processes of the mapping to attend the user.
- Total-automated approach: to try to complete all the process without human intervention.

One of more commonly used is the semi-automated, and is in this category where Anchor-PROMPT is placed. This is the approach use in this research [17]:

"...this one take as input pairs of related terms ("anchors") defined by the user, and then performs the remainder of the mapping process automatically: from the set of previously defined anchors, new pairs of semantically close terms are produced by analyzing the paths between the anchors in the corresponding ontologies".

A model to represent efficiency and effectiveness phases [18] can facilitate the compression of the phases of the different match approaches.

Abels, Haak and Hahn identified based on the literature, that the methods used by ontology integration approaches are [19]: text similarity; keyword extraction; language based methods; identification of word relations; type similarity / domains and ranges; class inheritance analysis; structural analysis / taxonomic structure; data interpretation and analysis of key properties, and graph- mapping.

4 Proposal of Methodological Variant for Integration

Noy and Musen [20] expose that "given the diversity in the tasks that the tools supports, their modes of interaction, the input data they rely on, it is impossible to compare the tools to one another and to find one or even several measure to identify 'the best' tool".

Therefore, the authors suggest that the evaluation must be user-oriented. "A user may ask either what is the best tool for his tasks or weather a particular tool is good enough for his task" [20]. Therefore they recommend for the evaluation pragmatic criteria such as:

- Input requirements: elements from the source ontologies do the tool use.
- Level of user interaction: batch or interactive mode, partial or final results.
- Types of output: what is the result of working with the tool?
- Content of output: which elements of the source ontologies are correlated in the output".

In this sense and searching of an operative, practical and structured answer, a methodological framework was taken from Stuckenschmidth and van Harmelen [7]. They presuppose an approach of shared terminology to be able to integrate in a compatible ontology (Fig. 2.2, page 37 cited).

As far as the three alternative ways suggested by these authors of how to employ ontologies for the integration, in this research we use the approach of multiple or local ontologies. We use it because it is try "to provide a new ontology source that allows to relate the other technologies" [7] (Fig. 2.1, page 32 cited).

The authors of the framework make a formal definition of shared terminology, shared ontology and source ontology that facilitates the understanding of each one of the resources required to derive the desirable shared ontology, these are [7]:

"A shared terminology is a tuple ⟨W,l⟩, where W is a set of words and $l : WxW \rightarrow$ {syno,hyper,hypo} is a partial function from the set of all pairs of terms into a set of identifiers specifying whether the first term in a synonym, a hypernym or hyponym of the second."

"A shared ontology is a tuple ⟨ST,T,R,A⟩, where $ST = \langle W_L, l_L \rangle$ is a shared terminology. T is a basic set of terms, R is a set of relations $R \subseteq TxT$ and A is a set of axioms of the form $T_i \sqsubseteq T_j$ if the following conditions hold:

$T \subseteq W_l$,

For each pair of words (W_i, W_j),
If $l((W_i, W_j)) =$ hyper then $W_j \sqsubseteq W_i$ is in A
If $l((W_i, W_j)) =$ hypo then $W_i \sqsubseteq W_j$ is in A
If $l((W_i, W_j)) =$ syno then $W_i \sqsubseteq W_j$ and $W_j \sqsubseteq W_i$ are in A"

"A source ontology is a tuple $\langle$S,C,d$\rangle$, where $S = \langle ST_S, T_S, R_S, A_S\rangle$ is a shared ontology, C is a set of class names not from the set of terms in S, L is a terminological language and d is a function that assigns expressions δ_i to class names C_i in C such that:

δ_i only refers to relations in R_S,

L is defined over T_S"

At the methodology proposed by such authors, the process steps are enumerated (Fig. 4.1, page 69 cited) as fallow:

1. Finding common concepts: common or semantically translated concepts.

2. Definition of properties: attributes that describe the concepts.

3. Finding property values: ranges and level of resolution of variables (values possible).

4. Adapt ontology: use and "invent" variations when specific terms are required.

5. Refine definitions: refining the concepts, the "bridges terms" and the vocabularies.

5 Applying the Approach

This study contemplates the integration of three ontological models (based on OWL) of a proposal for a University Institutions System [21]. First, two ontologies that were developed by others were integrate and have been taken from Internet. Both were developed in English language and deals with University dominions associate to the Administration—Management University Model. After that we have integrated the ontology shared resulting with the third ontology of the Company / Organization Model previously developed in Spanish language [22]. The proposed methods intend to gain some experience with integration using partial ontological models for futures universitary case or another practical applications.

As far as the methodological variant proposed in this paper, based on the original one developed by Stuckenschmidth and Harmelen [7], two important adjustments are made. One is related to information sources, obvious conditioned by the dominion of the application (University Institutions) and the other, somewhat more important, by the characteristics and scope of the model tool: Protégé-PROMPT.

5.1 Knowledge Sources:

In order to obtain the "more possible standardized" shared ontology, it will require to sharpen the vocabulary and the conceptualization. To reach those are used: Main Upper level ontologies, domain ontologies, dictionaries and thesauri with the typical terms of the dominion and additionally linguistic thesauri [23, 7].

a) Domain Ontologies:

a.1) UpperCyc Ontology. (`http://www.cyc.com`) This ontology was developed by CyCorp Corporation. The Cyc knowledge base was constructed in English language. The Cyc knowledge base (KB) is a formalized representation of a vast quantity of fundamental human knowledge: facts, rules of thumb, and heuristics for reasoning about the objects and events of everyday life. The Cyc KB is divided into many (currently thousands of) "microtheories", each of which is essentially a bundle of assertions that share a common set of assumptions; some "microtheories" are focused on a particular knowledge domain, a particular level of detail, a particular interval in time, etc. At the present time, the Cyc KB contains nearly two hundred thousand terms and several dozen hand-entered assertions about/involving each term. New assertions are continually added to the Cyc knowledge base by human knowledge enterers. Additionally, term-denoting functions allow for the automatic creation of millions of non-atomic terms. For this work the version 1.0.2.Win32 of the OpenCyc tool for Windows had been used.

a.2) Suggested Upper Merged Ontology (SUMO). (`http://suo.ieee.org`) This ontology is being created as part of the IEEE Standard Upper Ontology Working Group. The goal of this Working Group is to develop a standard upper ontology that will promote data interoperability, information search and retrieval, automated inference, and natural language processing. The SUMO has been translated into various representation formats, but the language of development is a variant of KIF. SUMO is a collection of approximately 1000 well-defined and well-documented concepts, interconnected into semantic network and accompanied by a number of axioms. The axioms mostly reflect common-sense notions that are generally recognized among the concepts. SUMO is intended as a domain-independent substrate for designing domain ontologies.For this work SUMO155 had been used.

a.3) SWAT Projects—The Lehigh University (LUBM). The Lehigh University Benchmark is developed to facilitate the evaluation of semantic Web repositories in a standard and systematic way
(`http://swat.cse.lehigh.edu/projects/lubm`). The benchmark is intended to evaluate the performance of those repositories with respect to extensional queries over a large data set that commits to a single realistic ontology. It consists of university domain ontology, customizable and repeatable synthetic data, a set of test queries, and several performance metrics. They are currently conducting experiments to determine the tradeoff space for various implementations of OWL repositories.

b) Domains and Linguistic thesauri:

b.1) WordNet. Is a large lexical database of English, developed by The Cognitive Science Laboratory of Princeton University
(`http://wordnet.princeton.edu`). Nouns, verbs, adjectives and adverbs are grouped into sets of cognitive synonyms (synsets), each expressing a distinct

concept. Synsets are interlinked by means of conceptual-semantic and lexical relations. Version 2.1 for the English language, used in this work, contains more than 120 thousand words and near 100 thousand concepts. It is organized in sets interrelated of synonymous (synset) and antonyms (antset), hypernymy and hyponymy (subclass-of and superclass-of), meronymy and holonymy (part-of and has-a).

b.2) Multilingual Extended WordNet. The words networks are structured of form similar to WordNet (`http://gplsi.dlsi.ua.es/wsd/ext_index.html`). Use a multi languages index that allows the searches of words meaning, the comparison of synset and its interrelations through different languages.

6 The Subsystems: Ontological models

Solution domain (University Institution) attempt a general model previously proposed [21]. The subsystems that conforms the semantic model are five, but those emphasized down (3 and 4) underlying are the worked ones for integration during this experimental approach, the original were: 1) Academic—Educational, 2) Production—Extension. 3) Company / Organization 4) Administration—Management University 5) Research and Development.

6.1 Specific Semantic models selected:

- 3) Company / Organization Model: Management model for company of service technologies (TICs).

– Sale and distribution of technology services (Paradigm ITIL).
– Construction Strategy: incremental using by referring upper ontology.
– Application scenario: human communication and access to data via transforming ontologies.

- 4) University Administration—Management Model: Management model e-learning and Government-e from existing ontologies and reference models taken from technical Literature.

– Architecture and Government-e supported in Web semantic.
– construction strategy: ontologies from others authors [24–26].
– Application scenario: access to data via shared ontologies and transforming ontologies.

7 Methodological Proposal Results

A method proposal is developed to obtain a shared ontology the study cases using Protégé-PROMPT, and the diverse sources of knowledge mentioned in 5.1. The shared ontology is derivate in four recursive phases trying to improve the definition of concepts and terminology (towards a Lightweight ontology):

1. Administration—Management University two-generic ontology integration (using PROMPT).

2. Upper ontologies comparison and aggregation (using UpperCyc, SUMO)

3. Terms and linguistic structure validation (using Extended Wordnet).

4. Company / Organization ontology merging (using PROMPT).

5. Terms and linguistic structure validation (using Extended Wordnet, Spanish).

6. Terms comparison and aggregation from a specific domain ontology (using LUBM).

8 Conclusions and recommendations

1. Integrate subsystems using ontologies is facilitated if versions of shared ontologies are achieved. In our case of study this strategy was feasible because the sources and terminology are "convergent enough", they can be specified in the same language (OWL) and the referring models are somewhat generic.

2. The integration strategy seems to be generalized to the other sub dominions of this study case, by the similarity in the strategy of development of the specific ontologies, which would be verified in later researches.

3. The methodological framework that was adapted as far as tools and knowledge sources, seems to be appropriate to the domain nature for University Institutions in addition to consider too, the diversities of languages sources (Spanish and English), where the ontologies could be conceptualize. This strategy must be a valid option for terminological integration of similar domains specified with sources multi-languages, characteristic of a globalize world.

4. Later research could be allowing to assuring the adaptability and changes possibility of the shared ontologies derived, will be the challenge of next studies and experimentations.

References

1. Wache H., Vögele T., Visser U., Stuckenschmidt H., Schuster G., Neumann H., Hübner S.: *Ontology-based Integration of Information—A Survey of Existing Approaches,* International Joint Conference of Artificial Intelligence (IJCAI 2001), Workshop: Ontologies and Information Sharing (2001).
2. Noy N.: *Semantic Integration: A Survey of Ontology-Based Approaches,* SIGMOD Record, Vol. 33, No. 4, 2004.
3. Buccella A., Cechich A., Brisaboa N.: *Ontology-Based Data Integration Methods: a Framework for Comparison,* Revista Colombiana de Computación, Vol. 6, No. 1 (2005).
4. Shvaiko P., Euzenat J.: *A Survey of Schema-based Matching Approaches,* Journal on Data Semantics (2005).
5. Kalfoglou Y., Schorlemmer M.: *Ontology Mapping: the State of the Art,* The Knowledge Engineering Review, Vol. 18:1, Cambridge University Press, UK (2003) 1–31.
6. Noy N., Stuckenschmidt H.: *Ontology Alignment: an Annotated Bibliography,* Dagstuhl Seminar Proceedings (2003).

7. Stuckenschmidt H., van Harmelen, F.: *Information Sharing on the Semantic Web,* Springer-Verlag, Berlin Heidelberg New York (2005).
8. Noy N., Musen M.: *PROMPT: Algorithm and Tool for Automated Ontology Merging and Alignment,* Proceedings of the 17th National Conference on Artificial Intelligence (AAAI 2000) (2000).
9. Noy N., Musen M.: *Using PROMPT Ontology-comparison Tools in the EON Ontology Alignment Contest,* Workshop on Evaluation of Ontology-based Tools (EON2004) at the Third International Conference on the Semantic Web (ISWC-2004), Hiroshima, Japan (2004).
10. Obrst L.: *Ontologies for Semantically Interoperable Systems,* Conference on Information and Knowledge Management (CIKM 2003), New Orleans, Louisiana, EE. UU. (2003).
11. Antonious G., van Harmelen F.: *A Semantic Web Premier,* The MIT Press (2004).
12. Vernardat F.: *Modelos Meta y Modelos para la Integración y la Interoperatividad de las Aplicaciones de Empresas de Producción,* PHD-Tesis, French (2006).
13. Kalfoglou Y., Schorlemmer M., Uschold M., Sheth A., Staab S.: *Semantic Interoperability and Integration,* Dagstuhl Seminar 04391 (2004).
14. López de Vergara J., Villagrá V., Berrocal J.: *An Ontology-based Method to Merge and Map Management Information Models,* Proceedings of the HP Openview University Association Tenth Plenary Workshop, Geneva, Switzerland (2003).
15. Park J., Ram S.: *Information Systems Interoperability: What Lies Beneath?* ACM Transactions on Information Systems, Vol. 22:4 (2004) 595–632.
16. Ding Y., Fensel D.: *Ontology Library Systems: the Key to Successful Ontology Reuse,* Proceedings of the First Semantic Web Working Symposium, Vrije Universiteit Amsterdan, Netherland (2001).
17. Villa R., Wilson R., Crestani F.: *Ontology Mapping by Concept Similarity,* Proceedings of the International Conference on Digital (2004).
18. Ehrig M, Staab S.: *Efficiency of Ontology Mapping Approaches,* International Workshop on Semantic Intelligent Middleware (2004).
19. Abels S., Haak L., Hahn A.: *Identification of Common Methods Used for Ontology Integration Tasks,* Proceedings of the First International Workshop on Information System, Bremen, Germany (2005).
20. Noy N., Musen M.: *Evaluating Ontology-mapping Tools: Requirements and Experience,* Workshop on Evaluation of Ontology Tools at EKAW, (2002).
21. Gil R., Contreras L., Ferrer J., Rivas C.: *Estrategia Ontológica para Abordar Heterogeneidad de Dominios en Instituciones Universitarias,* Conferencia Iberoamericana en Sistemas, Cibernética e Informática (CISCI 2007), Orlando, Florida, EE.UU. (2007).
22. Cumare L., Gil R.: *Bases para Desarrollar un Modelo Ontológico que Optimice Niveles de Servicio de Tecnología de Información Basados en las Mejores Prácticas de ITIL,* Caso de Estudio. Conferencia Iberoamericana en Sistemas, Cibernética e Informática (CISCI 2007), Orlando, Florida, EE.UU. (2007).
23. Gómez-Pérez A., Fernando-López, M. Corcho, O.: *Ontology Engineering,* Springer Springer-Verlag, London (2004).
24. Alani H.: *Ontology Construction from Online Ontologies,* Proceedings of the 15th International Conference on World, Edinburgh, Scotland (2006).
25. Taboada M., Martinez D., Mira J.: *Experiences in Reusing Knowledge Sources using Protégé and PROMPT,* International Journal of Human-Computer Studies 62 (2005) 597–618.
26. Noy N., Musen M.: *The PROMPT Suite: Interactive Tools for Ontology Merging and Mapping,* International Journal of Human-Computer Studies, Elsevier (2003).

International Workshop on Real Time Software (RTS'07)

International Workshop on Real Time Software will be held within the framework of the *International Multiconference on Computer Science and Information Systems*, and will be co-located with the XXIII Fall Meeting of Polish Information Processing Society.

Proliferation of computers interfacing with real world and controlling their environment requires careful investigation of approaches related to the specification, design, implementation, testing, and use of modern computer systems. Timing constraints, dependability, fault-tolerance, interfacing with the environment, reliability and safety constitute integral components of the software development process. Appropriate education of engineers developing such systems, working in interdisciplinary teams and in a global environment is of paramount importance.

In addition to traditional papers, we plan to organize a *round table discussion forum* on safety critical aspects of the education/training. We are soliciting brief one-page position papers on education and training of engineers developing dependable software intensive systems - presenting the views of academia and industry (in the submission clearly identity the "position paper" for the engineering education round table discussion). The accepted position papers will be a base for 10 minutes presentation followed by a discussion.

The workshop is planned around three main focus areas:

- Real-Time Control
- Safety, Reliability, and Dependability
- Real-Time Education

Traditional Papers topics include but are not limited to:

- Real-time system development
- Scheduling
- Safety
- Reliability
- Dependability
- Fault-tolerance
- Feedback control real-time scheduling
- Hardware-software co-design
- Standards and certification
- Control software
- Robotics and UAV
- Software development tools
- Model-based development
- Automatic code generation
- Real-time systems education
- Related engineering curricula
- Laboratory infrastructure
- Internet-based support

International Programme Committee

- Karl-Erik Årzén, Lund University, Sweden
- Mikhail Auguston , Naval Postgraduate School, USA
- Tuna Balkan, Middle East Technical University, Turkey
- Manfred Broy, Munich Technical University, Germany
- Matja~ Colnarič, University of Maribor, Slovenia
- Alfons Crespo, Universidad Politécnica de Valencia, Spain
- Albertas Čaplinskas, Institute of Mathematics and Informatics, Lithuania
- Jindřich Černohorský, VSB Technical University Ostrava, Czech Republic
- Karol Dobrovodský, Slovak Academy of Sciences, Slovakia
- Frank Golatowski, University of Rostock, Germany
- Luís Gomes, Universidade Nova de Lisboa, Portugal
- Vladimir Hahanov , KNURE, Kharkov, Ukraine
- Wolfgang Halang, FernUniversität Hagen, Germany
- Mats Heimdahl, University of Minnessota, USA
- Thomas Hilburn, Embry Riddle Aeronautical University, USA
- Michael Hinchey , Loyola College of Maryland, USA
- Janusz Kacprzyk , Polish Academy of Sciences, Poland
- Wolfgang Kastner, Vienna University of Technology, Austria
- Jan van Katwijk , Delft University of Technology, The Netherlands
- Phil Laplante, Penn State University, USA
- Tiberiu S. Letia , Technical University of Cluj-Napoca, Romania
- György Lipovszki, UTE, Budapest, Hungary
- Jacek Malec, Lund University, Sweden
- Gilles Motet, INSA-Toulouse, France
- Leo Mõtus, Estonian Academy of Sciences, Estonia
- Simin Nadjm-Tehrani, Linköping University, Sweden
- Libero Nigro, Università della Calabria, Italy
- Rauf Sadykhov , BSUIR, Belarus
- Bo I. Sandén, Colorado Technical University, USA
- Vilém Srovnal , V` B Technical University Ostrava, Czech Republic
- Tomasz Szmuc , AGH University of Science and Technology, Poland
- Miroslav ` véda, Brno University of Technology, Czech Republic
- Jean-Marc Thiriet, Laboratoire d'Automatique de Grenoble, France
- Leszek Trybus, Politechnika Rzeszowska, Poland
- Andrzej Turnau , AGH University of Science and Technology, Poland
- Shmuel Tyszberowicz , Tel-Aviv University, Israel
- Tullio Vardanega, Università di Padova, Italy
- Dieter Zöbel, University Koblenz-Landau, Germany

Organizing Committee

Andrew J. Kornecki (Chairman), Embry Riddle Aeronautical University, USA
Wojciech Grega, AGH University of Science and Technology, Poland
Janusz Zalewski, Florida Gulf Coast University, USA

Proceedings of the International Multiconference on Computer Science and Information Technology pp. 811–820
ISSN 1896-7094
© 2007 PIPS

Interactive Real-Time Control Labs with TrueTime and Easy Java Simulations

Gonzalo Farias[1], Karl-Erik Årzén[2], and Anton Cervin[2]

[1] Departamento de Informática y Automática, UNED, Spain,
gfarias@bec.uned.es
[2] Department of Automatic Control, Lund Institute of Technology, Sweden,
{karlerik, anton}@control.lth.se

Abstract. This paper presents the development of interactive real-time control labs using TrueTime and Easy Java Simulations. TrueTime is a freeware Matlab/Simulink based tool to simulate real-time control systems, and Easy Java Simulations allows rapid creation of interactive simulations in Java. Authors can use TrueTime to develop the simulation of a real-time control system, and then move to Easy Java Simulations to link the system and create the graphical user interface which provides the visualization and user interaction. The combination of these tools brings together the best of them.

1 Introduction

Control education has to adapt to the new scenario that the information technologies provide. In this context interactive virtual labs take advantage of these new possibilities and improve the understanding of the system behavior [1, 2]. The interactivity should allow the user to simultaneously visualize the evolution of the system, and its response on-the-fly to any change introduced by the user. This immediate observation of the gradient of change of the system as response to user interaction is what really helps the student get useful practical insight into control system fundamentals. TrueTime is a MATLAB/Simulink based tool [3–5], which facilitates co-simulation of controller task execution in real-time kernels, network transmissions, and continuous plant dynamics. The tasks are processes that control continuous-time plants modeled as ordinary Simulink blocks. However, models created with Simulink suffer from a certain lack of interactivity in the sense described. A typical instructor would face difficulties if (s)he had to develop, using only Simulink, virtual labs with graphical and interactive capabilities.This is where Easy Java Simulations (Ejs) comes in handy. Ejs is a software tool designed to create simulations in Java with high-level graphical capabilities and with an increased degree of interactivity [6]. The paper is organized as follows. In Section 2 both tools TrueTime and Ejs are introduced. In Section 3 main aspects of the integration between TrueTime and Ejs are commented. Section 4 shows three examples of this approach. Finally, Section 5 presents the main conclusions and further work.

2 TrueTime And Easy Java Simulations

In this Section both TrueTime and Easy java Simulation (Ejs) are described briefly. TrueTime is a Matlab/Simulink based tool to simulate real-time systems, and Ejs is an authoring tool to create simulations in Java. Since Ejs has a link to reuse Simulink models, authors can create interactive real-time control labs combining both tools.

2.1 A Brief Overview Of TrueTime

TrueTime is a Matlab/Simulink-based simulator for networked and embedded control systems that has been developed at Lund University since 1999. The simulator software consists of a Simulink block library (see Fig. 1) and a collection of MEX files. The kernel block simulates a real-time kernel executing user-defined tasks and interrupt handlers. The various network blocks allow nodes (kernel blocks) to communicate over simulated wired or wireless networks. TrueTime can be downloaded from `http://www.control.lth.se/truetime/`.

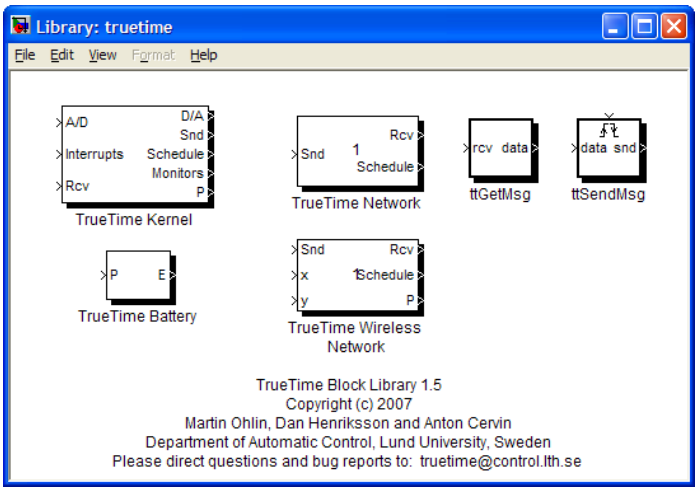


Fig. 1. The TrueTime 1.5 block library

To create real-time simulations TrueTime provides mainly two kinds of blocks, TrueTime Kernel and TrueTime Network. The TrueTime Kernel block simulates a computer node with a generic real-time kernel, A/D and D/A converters, and network interfaces. The block is configured via an initialization script. The script may be parameterized and the programmer may create objects such as tasks, timers, interrupt handlers, etc., representing the software executing in the computer node. The TrueTime Kernel block supports various pre-emptive scheduling

algorithms such as fixed-priority scheduling and earliest-deadline-first scheduling. It is also possible to specify a custom scheduling policy. The TrueTime Network block and the TrueTimeWireless Network block simulate the physical layer and the medium-access layer of various local-area networks. The types of networks supported are CSMA/CD (Ethernet), CSMA/AMP (CAN), Round Robin (Token Bus), FDMA, TDMA (TTP), Switched Ethernet, WLAN (802.11b), and ZigBee (802.15.4). The blocks only simulate the medium access (the scheduling), possible collisions or interference, and the point-to-point/ broadcast transmissions.

2.2 A Brief Overview of Easy Java Simulations

Ejs is an authoring tool designed for rapid creation of interactive simulations in Java. Ejs is different from most other authoring tools in that Ejs is not designed to make life easier for professional programmers, but has been conceived by science teachers, for science teachers and students. That is, for people who are more interested in the content of the simulation, and much less in the technical aspects needed to build the simulation. Ejs can be downloaded from http://www.um.es/fem/Ejs/.

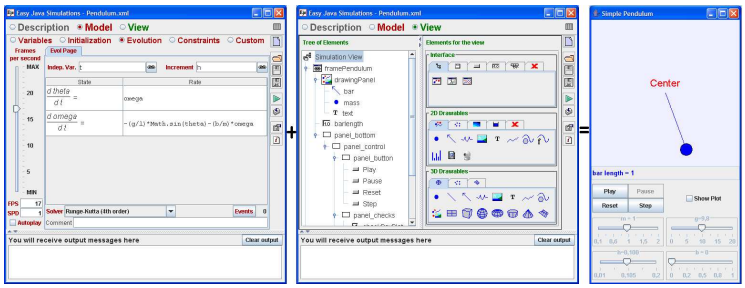


Fig. 2. Ejs uses the model and the view to create the simulations

Ejs structures a simulation in two main parts, the model and the view (see Fig. 2). The model can be described by means of Ordinary Differential Equations, Java code or external applications. The view provides the visualization of the simulated system, either in a realistic form or using one or several data graphs, and the user interface elements required for user interaction. These view elements can be chosen from a set of predefined components to build a tree-like structure. Both model and view need to be interconnected. Any change in the model state must be immediately reflected by the view in order to keep a dynamic and on-the-fly visualization of the system.

3 Integration TrueTime and Easy Java Simulations

Ejs has an special link with Matlab/Simulink. In this case Ejs is used to create the graphical user interface and the TrueTime model to provide the system behavior.

3.1 Improving the Link

The connection with Simulink models allows users develop in an easy and fast way interactive simulations. The procedure, described in [7, 8], is quite simple. It basically consists in connecting Ejs' variables to signals (input, output or parameters) of the blocks in the Simulink model. Ejs also provides a set of predefined methods that allow the users read and write variables in the Matlab workspace, for instance to read the variable *myMatlabVar* from Matlab workspace the next instruction could be used: *myEjsVar=_ external.getDouble("myMatlabVar");*

These methods are similar (same signature) to functions defined in the Matlab Engine Library. In this way the integration between TrueTime and Ejs is very direct, but to improve the performance of the initial results it was needed to consider two features of TrueTime simulations, i.e., zero crossings evaluations and scheduler data.

The first aspect is taken into account because the simulation with TrueTime involves a lot of zero-crossing functions, which produces slow, though good enough, simulations using the TrueTime Simulink blocks. However the simulations in the Ejs-TrueTime combination are even much slower. To solve this obstacle a new way to link Ejs and Simulink was created. Until now the updating of the Ejs' variables was done in every step of the Simulink solver, but with the new link the variables are updating at fixed time intervals, which involves a faster and "smooth" simulation. In the Fig. 3 more details are presented. To indicate this link the user just needs to specify the time interval in the External File option in Ejs (see Fig. 4).

The second feature considered was the schedule output signals generated by TrueTime.These signals are rather important to understand the performance of the real-time system, so it is necessary to catch all samples from schedule data. However, Ejs has a mechanism to avoid waiting a long time for a Matlab variable, so sometimes a few samples from schedulers were lost and the plot of these signals was not good. Hence the new method *_ external.setWaitForEver(true)* was added in Ejs to wait until the sample is caught.

3.2 Creating the Interactive Simulations

After the improvement phase, the linking between TrueTime models and Ejs was direct, in this situation just two main activities were done, selecting the signals and connecting the global variables (see Fig. 4). The procedure to select the signals from TrueTime model is the same as in any other Simulink model, and is simple because the link between Ejs variables and Simulink signals is done by a single mouse click. The procedure to connect global variables and

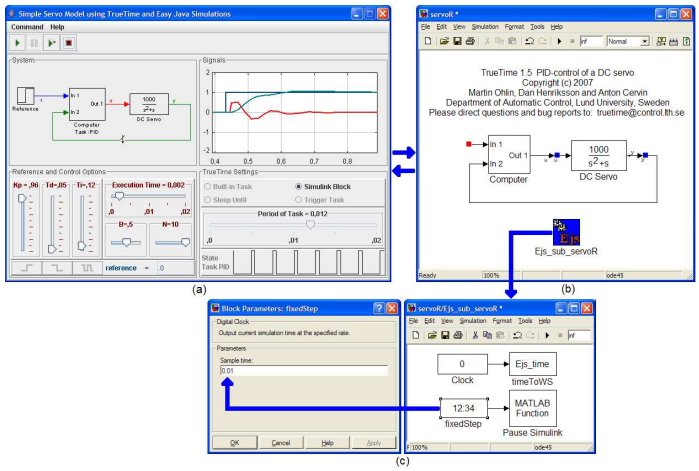


Fig. 3. Ejs simulations using a TrueTime model. The graphical user interface is shown in (a), the Simulink model is presented in (b), in (c) the submodel Ejs_sub_servoR shows the blocks to speed up the simulation. Note that the fixed time interval is equals to 0.01

Ejs variables requires more work, but it is also simple. In order to affect any variable in the TrueTime code functions, it just needed to declare them as global and use the read and write predefined methods in Ejs to set or get the values. An example of a modified code function is presented in Listing 1.1. Finally the method *_external.step()* is used in the Evolution section to simulate the Simulink model.

Listing 1.1. pidcode1 function modified to set a link between Ejs and TrueTime. Commented lines are original lines of the function

```
function [exectimeAux,data]=pidcode1(seg,data_) % Ejs
% function [exectime,data]=pidcode1(seg,data)
global data exectime; % Ejs
switch seg,
case 1,
  r = ttAnalogIn(data.rChan); % Read reference
  y = ttAnalogIn(data.yChan); % Read process output
  data = pidcalc(data, r, y); % Calculate PID action
  exectimeAux=exectime; % exectime = 0.002*rand;
case 2,
  ttAnalogOut(data.uChan, data.u); % Control Signal
  exectimeAux = -1; % exectime = -1;
end
```

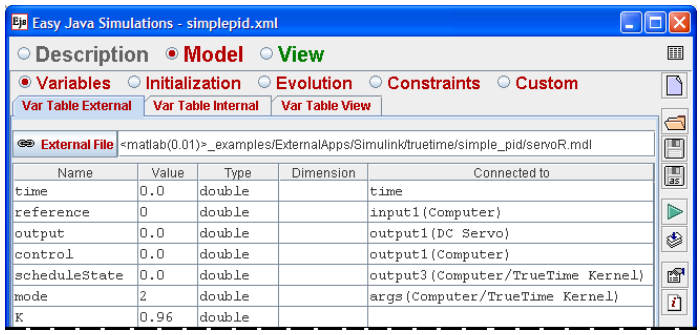


Fig. 4. Setting a link between Ejs and a TrueTime model, note the fixed time interval updating is used and the Ejs' variables are connected to inputs or outputs of Simulink's blocks

4 Examples

In this section three examples using the TrueTime-Ejs integration will be shown. The first example is an introduction to the TrueTime simulation environment. The second one is an extension of the first one. Finally, the third example uses TrueTime Network block to show a distributed control application.

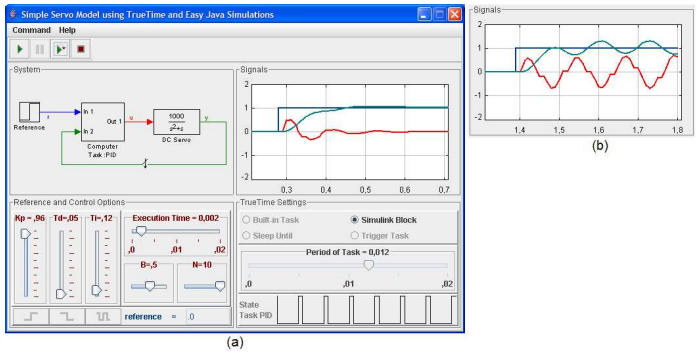


Fig. 5. (a) Graphical User Interface of the first example. Note that it is possible to change the control parameters on-the-fly using the sliders. (b) The Signals when Execution Time is incremented from 2[ms] to 9[ms]

4.1 First Example: The Simple PID Servo Controller

The example considers simple PID control of a DC-servo process. The process is controlled by a controller task implemented in a TrueTime kernel block. Four



Fig. 6. Graphical user interface of the second example. Figures: (a) is the main view, (b) show histograms of input-output latencies or response time for three tasks, (c) is the dialogCode where the user can read and write the code of PID computation. Note the schedule data of the red task and its control performance

different modes of implementation of the controller task are provided: Built-in Task, Simulink Block, Sleep Until and Trigger Task. The graphical user interface of the example is shown in Fig. 5. The user can modify different parameters like: reference type, control settings, execution time of the controller and the mode of the implementation of the tasks. In the Fig. 5b the performance of the controller is shown when the execution time change from 2[ms] to 9[ms].

4.2 Second Example: The Three Servo Controllers

This example extends the simple PID control to the case of three PID-tasks running concurrently on the same CPU controlling three different servo systems. The effect of the scheduling policy on the global control performance is demonstrated. The graphical user interface is shown in Fig. 6. Main view has the same appearance than the previous example. There is also an auxiliary dialog to show histograms of input-output latencies or response time of three tasks, and there is another dialog to enable the user to modify the code for calculate the PID action.

In the main view the user can modify the parameters for three controllers as in the first example, but also can add some jitter to the execution time. Note also the scheduler data for the tasks is presented in this window. The policies are chosen from radio buttons Rate Monotonic and Earliest Deadline First. Note

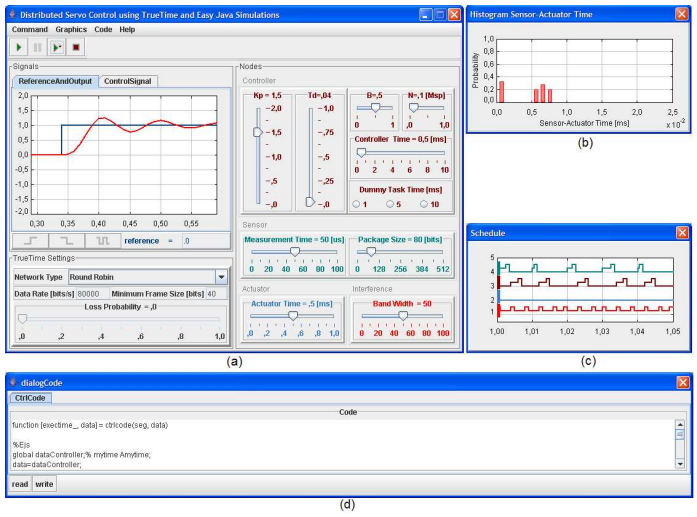


Fig. 7. Graphical user interface of the third example. Figures: (a) is the main view, (b) show histograms of sensor-actuator time, (c) is the scheduler data for the four nodes, (d) is the dialogCode where the user can read and write the code of controller

in Fig. 6b the input-output latencies for three task, since the rate monotonic policy was selected the task 1 (red) has the lowest priority and the performance is not good. The dialogCode can be used for instance to modify the algorithm to compute the PID controller.

4.3 Third Example: Distributed Servo Control

This example simulates distributed control of a DC-servo. The example contains four computer nodes, each represented by a TrueTime kernel block. A time-driven sensor node samples the process periodically and sends the samples over the network to the controller node. The control task in this node calculates the control signal and sends the result to the actuator node, where it is subsequently actuated. The simulation also involves an interfering node sending disturbing traffic over the network, and a disturbing high-priority task executing in the controller node. The Graphical User Interface is shown in Fig. 7. The main view allows the user to modify parameters of the network and nodes. There are also three auxiliary dialogs to show an histogram of end-to-end latency, to modify the code for the controller node, and to show the scheduler data for the four tasks.

In the main view the user can modify control parameters and also add a dummy disturbing high-priority task with different execution times. In the same view, but in the section Sensor the user can modify the measurement time and the package size. This last parameter is important to see the effect of the size of

the package in the control performance. The section BandWidth allows the user to increase bandwidth used by the interference node. The network parameters can be modified in the TrueTime Settings section.

5 Conclusions

Interactivity is crucial aspect in virtual labs that are to be used for pedagogical purposes in the field of control engineering.

In the context of simulation of real-time systems, TrueTime is a freeware that provides functionalities that simulates control models under resources constraints. However from a teaching point of view, and since True-Time is a Matlab/Simulink based tool, to create an interactive simulations using Matlab features could demand hard work. That's why Ejs was used.

Ejs is also a freeware tool designed to create quickly interactive simulations in Java. Besides, Ejs has a direct link with Matlab/Simulink models, and is not difficult to get a first version of the interactive lab after a few minutes. However it was necessary to improve the link in order to get more robust and faster simulations. That finally involved new features that made possible the acceleration of the simulation and prevented the loss of scheduler data.

Three examples of TrueTime models were developed. These allow the user to modify parameters like execution time, jitter, and see how the whole system is affected.

The creation of this kind of labs is not difficult, and probably in order to get a final version a couple of days is needed. However most of the time the designer will not worry about Java programming details, but about finding new functionalities to add to the lab.

Further work could involve further development of web-based labs or the creation of a link between Ejs and Scilab version of TrueTime.

References

1. Heck B. S. (editor): *Special report: Future directions in control education: IEEE Control Systems Magazine,* Vol. 19, No. 5, (1999) 35–58.
2. Dormido S.: *Control learning: Present and future: IFAC Annual Control Reviews,* Vol. 28, (2004), 115–136.
3. Ohlin M., Henriksson D., Cervin A.: *TrueTime 1.5 Reference Manual.: Manual,* Department of Automatic Control, Lund University, Sweden, (2007).
4. Cervin A., Henriksson D., Lincoln B., Eker J. and Årzén K.: *How does control timing affect performance?* IEEE Control Systems Magazine 23(3), 16–30,(2003).
5. Andersson M., Henriksson D., Cervin A. and Årzén K.: *Simulation of wireless networked control systems,* In Proceedings of the 44th IEEE Conference on Decision and Control and European Control Conference ECC (2005). Seville, Spain.
6. Esquembre F.: *Easy Java Simulations: A software tool to create scientific simulations in Java,* Comp. Phys. Comm. 156, (2004), 199–204.
7. Sánchez J., Dormido S., Esquembre F.: *The learning of control concepts using interactive tools,* Computer Applications in Engineering Education, Vol. 13, No 1, (2005) 84–98.

8. Dormido S., Esquembre F., Farias G., Sánchez J.: *Adding interactivity to existing Simulink models using Easy Java Simulations,* In Proceedings of the 44th IEEE Conference on Decision and Control and European Control Conference ECC (2005). Seville, Spain.

Proceedings of the International Multiconference on
Computer Science and Information Technology pp. 821–835
ISSN 1896-7094
© 2007 PIPS

Modelling and Schedulability Analysis of Real-time Sequence Patterns using Time Petri Nets and Uppaal

Angelo Furfaro and Libero Nigro

Laboratorio di Ingegneria del Software,
Dipartimento di Elettronica Informatica e Sistemistica,
Università della Calabria, 87036 Rende(CS) - Italy
Email: a.furfaro@deis.unical.it, l.nigro@unical.it
Web: http://www.lis.deis.unical.it

Abstract. This paper proposes an original approach to the schedulability analysis of real-time systems specified by Time Petri Nets (TPNs). The focus is on sequence patterns of transition firings (execution tasks). A TPN model is first translated in the Timed Automata terms of the popular Uppaal tool. Then schedulability properties of tasks are verified through reachability analysis. The approach is efficient and scalable. The paper demonstrates the concrete application of the approach through examples. Finally, conclusions are drawn together with an indication of on-going and future work.

Keywords: real-time systems, time Petri nets, timed automata, Uppaal, schedulability, firing sequence patterns

1 Introduction

Time Petri Nets (TPNs) [16] have been proven to be a very convenient tool for specifying timing constraints in time-dependent systems [6, 22, 10]. TPNs associate time pairs with transitions instead of a single delay as in timed Petri nets [18], thus TPNs are more general than timed Petri nets [4]. TPNs support formal analysis through an adaptation [4, 6] of the well-known reachability analysis technique of Petri nets [17]. A reachability graph represents the complete dynamic behaviour of a TPN based on the interleaving semantics. Each node of the graph is a *state class*. Edges are labelled by transition firings. A state class holds a *marking* and a *firing domain* reflecting timing constraints about *when* the state class is possibly reached in the time domain. Examples of concrete tools permitting the enumeration of the state classes of TPNs and enabling reachability analysis are Tina [5, 20] and Romeo [12, 19].

As pointed out in [22], although schedulability is closely related to reachability, it has more specific concerns with transition sequences rather than markings or states. In particular, an end-to-end delay of a task execution, that is an important issue in time critical systems, cannot directly be derived from firing domains and state classes.

In [22] a method is proposed for making schedulability analysis for a subclass of TPNs referred to as "well-structured" (WS) TPNs. In a WS TPN any firing sequence leading from the initial marking M_0 to a given reachable marking M_n, can be built by composing firing sequences taken from a finite set of elementary sequences including basic loops (see later in this paper for an example).

This paper proposes an original approach to schedulability analysis of sequence patterns of real-time systems abstracted by TPNs. First a TPN model is translated into Timed Automata (TA) [2] in the context of the UPPAAL tool [3, 21]. Then TCTL model checking on the achieved TA product system is used, possibly with the help of *decoration* means [15]. Translation is assisted by a customization of the TPN DESIGNER toolbox [7] which supports various kind of time-extended Petri nets, e.g. GSPN-like. TPN DESIGNER allows one to (i) graphically draw a TPN model (ii) simulate the model for preliminary behavioural/timing checking (iii) generate an equivalent TA representation of the model to be used for exhaustive verification in UPPAAL. The translation rests on a single TA template associated with generic TPN transition. The translation is similar to that described in [8] but was independently achieved. Motivations for choosing UPPAAL instead of existing special-case TPN tools are related to the desire of exploiting a popular tool which ensures an efficient and scalable approach.

The paper is structured as follows. Section 2 introduces TPN modelling concepts and describes a real-time model example together with its firing sequences (execution tasks) schedulability aspects. Section 3 summarises the translation process from TPNs to TA/UPPAAL and applies it to the chosen model example. A *decorator* automaton is proposed which enables task schedulability analysis. More general handling of sequence patterns in which the first transition can occur multiple times is then discussed. Section 4 gives some information about the effectiveness of the proposed approach and translation tool. Finally, conclusions are presented along with an out look to on-going and future work.

2 TPN Modelling and Schedulability Concerns

2.1 TPN Concepts

In this work Time Petri Nets are assumed augmented by inhibitor arcs. A TPN is a tuple $TPN = (P, T, B, F, I_{nh}, M_0, I)$ where P and T are non empty finite disjoint sets respectively of places and transitions of the underlying Petri net [17]; B is the backward incidence function: $B : T \times P \to \mathbb{Z}$, with $\mathbb{Z}$ the set of integers; F is the forward incidence function: $F : T \times P \to \mathbb{Z}$; I_{nh} is a set of inhibitor arcs: $I_{nh} \subseteq P \times T$; M_0 is the initial marking: $M_0 : P \to \mathbb{N}$, i.e. the non negative number of tokens assigned to places; I is the static firing interval function, $I : T \to \mathbb{N} \times (\mathbb{N} \cup \{\infty\})$. $B(t, p) = w > 0$ if there is an arc with weight w from input place p to transition t. $B(t, p) = 0$ for a non existing input arc (t, p) or for an inhibitor arc $(p, t) \in I_{nh}$. An inhibitor arc is graphically represented by a dot terminated line. $F(t, p) = w > 0$ if there exists an arc from transition t to output place p with positive weight w. $F(t, p) = 0$ for a non existing output arc

(t, p). The set of input and inhibitor places of t is said its *preset* (denoted ${}^\bullet t$). The set of output places constitutes the transition *postset* (denoted $t^\bullet$). I associates with each transition t a *dense* firing interval whose bounds are supposed to be specified by non negative integers: $I(t) = [a, b]$ with $0 \leq a \leq b$, b can be ∞. Bound a is said the (static) *earliest firing time* (EFT) of t, b the (static) *latest firing time* of t (LFT).

A transition t is said to be enabled in marking M, denoted by $M[t >$, iff: $\forall p \in P\ M(p) \geq B(t, p)$ if $(p, t) \notin I_{nh}$, $M(p) = 0$ if $(p, t) \in I_{nh}$. Let τ be an instant in time when transition t is enabled. Provided t is continuously enabled, t cannot fire before $\tau + a$ but *must* fire (*strong firing model*) before or at $\tau + b$, unless it is disabled by the firing of another (conflicting) transition. Transition firing semantics is best understood by using the *server* analogy and by thinking about of an hidden *timer* (*clock*) associated with the transition. As soon as the transition gets enabled, the timer is reset and the underlying server starts working (transition under firing). Let θ be the elapsed time since the enabling. A *time-strict* transition ($b < \infty$) can complete its firing when $a \leq \theta \leq b$, whereas for a non time-strict transition $a \leq \theta < b$. Whether a transition can complete its own firing depends in general on the firing condition of all the other transitions in the model.

When the transition completes its firing, tokens are removed from ${}^\bullet t$ and new tokens are generated into $t^\bullet$ as in classical Petri nets. Let M_{before} be the net marking when t fires. The firing of t transforms M_{before} in M_{after}, denoted by $M_{before}[t > M_{after}$, by an instantaneous and atomic process in two phases:

1. *token withdrawal*
   $\forall p \in P$, $M'(p) = M_{before}(p) - B(t, p)$
2. *token deposit*
   $\forall p \in P$, $M_{after}(p) = M'(p) + F(t, p)$

M' represents the intermediate marking generated after token withdrawal. A transition t_p is said *persistent* to the firing completion of transition t if t_p is enabled in M_{before}, and maintains its enabling condition also in M' and in M_{after}. A non persistent transition can lose its enabling either at M' or at M_{after}. Following the token deposit phase one or more transitions can become enabled. They are said *newly enabled* transitions. After its own firing, if t is still enabled, it is considered a newly enabled transition (*single server semantics*).

2.2 A Modelling Example

In the following an assembly subsystem of a flexible manufacturing system (FMS), adapted from [22], is modelled and its schedulability concerns highlighted. An FMS is a real-time system composed of a number of computer controlled tools and automated material handling, assembly and storage systems that operate in an integrated way under the control of host computers. FMS design is challenging in terms of control and scheduling, due to growing demand for high performance and flexibility despite the presence of interlocking factors of concurrency, deadline-driven activities and real-time decision making.

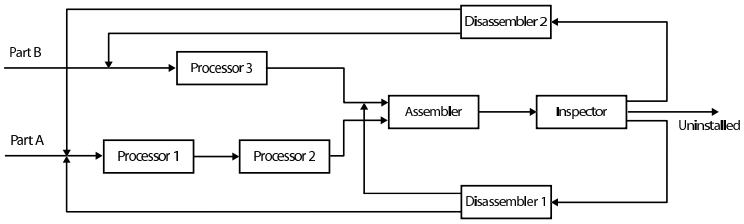


Fig. 1. An FMS assembly subsystem

In the chosen example (see Fig. 1) there are three processors, one assembler, one inspector and two disassemblers. The system receives two type of raw parts (A and B) in input. After processing the input parts, A-part and B-part are assembled in a final product. A-parts are processed in sequence by processor 1 and processor 2. B-parts are processed by processor 3. The inspector is responsible for quality control assessment of assembled products. If an assembled product satisfies quality requirements, it is delivered as a finished product. Otherwise the product is disassembled either by disassembler 1 or 2, depending upon their status. Disassembler 1 extracts A-parts which are sent back to processors 1 and 2, while B-parts are instead sent directly to the assembler. Disassembler 2 generates A-parts which are sent back to processors 1 and 2 and B-parts which are input to processor 3. All components in Fig. 1 have certain timing constraints imposed upon them. According to timing constraints, it is necessary to check different assembling schedules. Whether an A-part and B-part are assembled within a given time, mostly depend on the analysis of the following three cases:

1. neither A-part nor B-part has any quality problem
2. there is no problem with B-part, but A-part cannot pass quality control for m times ($m > 0$)
3. A-part and B-part cannot pass quality control respectively for m and n times ($m > 0$, $n > 0$).

Fig. 2 shows a TPN model of the assembly subsystem along with the assumed timing constraints. Transition t_0 models the reception of both A-part and B-part. Transitions t_1, t_2 and t_3 model respectively processor 1, 2 and 3. Transition t_4 represents the assembler, whereas t_5 denotes the inspector. Transition t_6 expresses unloading of a finished product, which can go in input to another subsystem etc. Transition t_7 models the disassembler 1, whereas transition t_8 models the disassembler 2. The initial marking of the model in Fig. 2 mirrors A-part and B-part are already available for processing. Analysis of case 1 above, means checking the following transition sequences: $t_0\sigma t_4 t_5 t_6$ where σ, due to the concurrent paths $t_1 t_2$ and t_3, can be: $\sigma = t_1 t_2 t_3$ or $t_1 t_3 t_2$ or $t_3 t_1 t_2$. Checking case 2 requires analysing the following transition sequences:

$t_0\sigma t_4 t_5 \sigma_1^m t_6$, where $\sigma_1 = t_7 t_1 t_2 t_4 t_5$.

Finally, checking case 3 means analysing sequences (roughly) expressed by:

$t_0\sigma t_4 t_5 \sigma_1^m \sigma_2^n t_6$, where $\sigma_2 = t_8 \sigma t_4 t_5$.

It is worth noting that repetitions of subsequences (loops) σ_1 or σ_2 can be interleaved. Places p_{10} and p_{11} serve the purposes of controlling respectively the occurrences of loops σ_1 and σ_2. Setting the marking of p_{10} and p_{11} to 0, forces straight sequences $t_0\sigma t_4 t_5 t_6$ to occur. In the general case, p_{10} will have m tokens as its initial marking, whereas p_{11} n tokens.

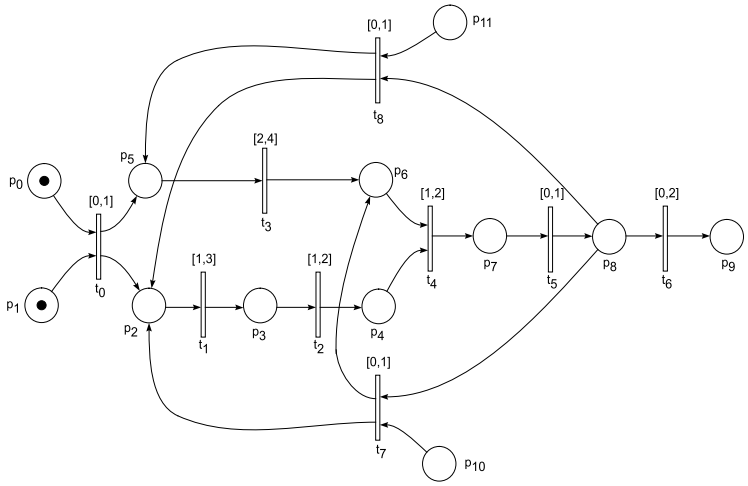


Fig. 2. TPN model of system in Fig. 1

Model in Fig. 2 is an example of a well-structured TPN. Any transition sequence leading to final marking $M_n = \{p_9\}$ can be composed by concatenating basic subsequences as mentioned above. Schedulability analysis of task executions involves both *behavioural* and *timing* verification. Behavioural analysis concerns assessing the functional requirement that each task is effectively schedulable, i.e. it *can* occur. Timing analysis means determining the time span of the entire task and of each belonging transition firing event.

3 Uppaal Translation and Schedulability Analysis

3.1 Mapping TPNs on to Uppaal

Uppaal [3, 21] makes it possible to verify systems modelled as networks of timed automata (TA) [2] extended with integer variables, structured data types (e.g. arrays) and channel synchronization (CSP-like *rendezvous*).

In the approach advocated in this work the use of TPN Designer [7, 9] is combined with that of Uppaal. In particular, first a TPN model is graphically drawn in TPN Designer, then it is pre-checked in simulation; finally, the model is translated in Uppaal as a product of as many TA as there are transitions in the TPN model. Fig. 3 shows the unique proposed template automaton which

reproduces semantics of a TPN transition. The automaton purposely exploits features of latest UPPAAL 4.0.6 version, in particular C data structures, function declarations and loop constructs.

Transition automaton has three parameters: transition unique ID, clock x and a broadcast fire channel for signalling completion of transition firing. Let T and P denote respectively the set of transitions and the set of places in a model, PRE and POST the (maximal) set of input places and output places of any transition. The translation generates a compact representation of the backward $B_{|T|\times|PRE|}$ and forward $F_{|T|\times|POST|}$ incidence matrices (realized as constant data structures) of a TPN model, its marking vector $M_{|P|}$ and the time interval vector $I_{|T|\times 2}$ (constant data structure). An Info struct is defined which contains an index in M to select a place, and the weight of an arc linking the place to a transition or vice versa. An index value of -1 is used for a non existing arc. Each element of B or F is an Info value. A non strict time interval like $[a, \infty]$ is represented in I as $[a, -1]$. Global functions bool enabled(const int ID), void withdraw(const int ID) and void deposit(const int ID) respectively check transition enabling, withdraw tokens from input places and generate tokens to output places; they refer to a particular transition by its ID received as a parameter.

Model bootstrapping is achieved by a first synchronization through the broadcast end_fire channel, which is launched by the Starter automaton shown in Fig. 4. The synchronization allows transitions enabled in the initial marking of the TPN model to move to Firing/U_Firing location. Not enabled transitions remain in their Disabled initial location. After model bootstrapping, Starter will take no part in the subsequent evolution of the model.

An enabled transition starts firing by resetting its clock x and moving to Firing if its time interval is strict, or to U_Firing if its latest firing time is ∞. An enabled transition can complete its firing as soon as its clock goes beyond its earliest firing time. A time strict transition is obliged to complete its firing at its latest firing time, provided it is still enabled, by the invariant x <= I[ID][1] attached to Firing. Transition firing, though, is non deterministic among transitions which can complete their firing at current time. Firing completion follows exactly the instantaneous and atomic process in two steps described in section 2.1, which involves committed locations Withdraw and Deposit in which time is not allowed to pass. The transition first removes tokens from its input places and signals firing completion through the fire broadcast channel. Then it moves to the Withdraw location where it sends (*first step*) a broadcast synchronization signal over the end_fire broadcast channel. This signal forces all other transitions, disabled or under firing, to re-evaluate their enabling status. This is crucial for proper management of conflicts. A no longer enabled transition moves from Firing/U_Firing to Disabled and resets its clock. The firing process continues by generating tokens in the output places and then by sending (see Deposit outgoing edges in Fig. 3) a second broadcast synchronization through end_fire (*second step*) which permits detection of newly enabled transitions and non persistent transitions which lose their enabling due to inhibitor arcs. The

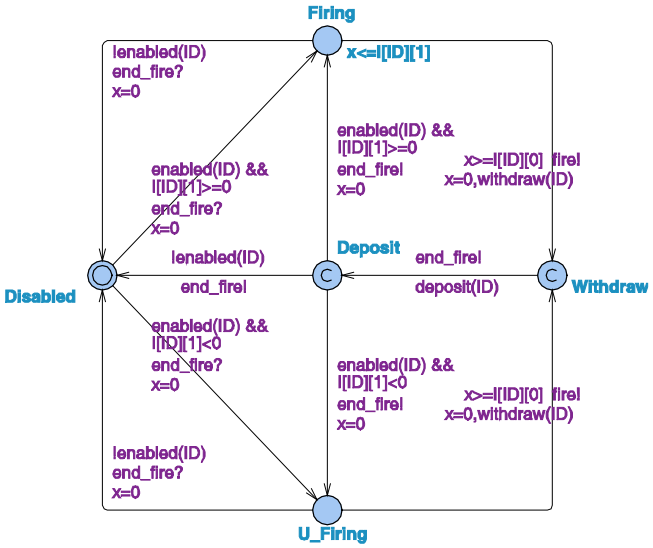


Fig. 3. Transition automaton



Fig. 4. Starter automaton

Transition automaton follows the *single server* firing policy: after its own firing, a still enabled transition is regarded as a newly enabled one and its clock reset.

A TPN model is transformed into the parallel composition of $|T|$ instances of the automaton in Fig. 3 corresponding to the $|T|$ transitions of the source model, and one instance of the Starter automaton. Some *decoration* [15] processes can possibly be added to help schedulability analysis (see later in this paper). Translation correctness can be formally proved by showing that the semantics of the translated UPPAAL model is *timed bisimilar* to that of the original TPN model, by preliminarily mapping both representations on timed transition systems. The formal proof is similar to the one described in [8]. The equivalence of a TPN model and its corresponding UPPAAL model, enables TCTL model checking activities [1] which can directly be interpreted at the TPN level. As for the approach of [8], actually only bounded TPN models can be efficiently verified (checking both safety and bounded liveness properties) using the proposed translation process.

3.2 Schedulability analysis of the FMS model

Being well-structured, the TPN model of Fig. 2 can be analyzed in UPPAAL with the assistance of the Decorator automaton shown in Fig. 5. The Decorator has

paths corresponding to the various transition sequences admitted by the model. The Decorator has distinct locations for each transition firing of a task execution. Loops in Fig. 5 directly correspond to sequence loops in Fig. 2. The Decorator relies on the global fire array of $|T|$ broadcast channels. At system configuration time, each element of the fire vector is associated with a distinct transition.

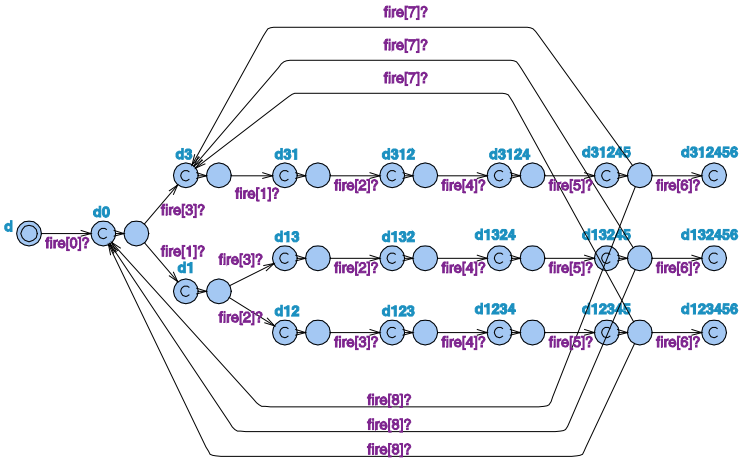


Fig. 5. Decorator automaton for the TPN model in Fig. 2

Verification experiments were separately carried out by checking the three cases in section 2.2. For case 1, the initial marking of places p_{10} and p_{11} (actually the values of $M[10]$ and $M[11]$, M being the marking vector) was set to 0. Behavioural analysis was accomplished by issuing to the UPPAAL verifier "existential" queries like the following:

```
E<> Decorator.d312456
```

which checks that effectively a state of the state graph exists in which the location d312456 of the Decorator is reached. Similar queries can be used for locations d132456 and d123456 which mark other tasks termination. All these queries were found satisfied, which in turn ensured each involved transition is effectively schedulable, i.e. it can fires according to its assigned time interval during task execution. The query

```
Decorator.d --> Decorator.d312456 || Decorator.d132456
                                  || Decorator.d123456   (1)
```

checks that one of three tasks is always terminated, i.e. that from the initial location Decorator.d it always follows that the Decorator ends in one of its final locations.

After behavioural tests, attention was focussed on the timing aspects of tasks. To this purposes, a global clock `z` (another decoration feature) was introduced and separately checked on the committed locations of the various transition sequences. For example, the time span of the sequence $t_0t_3t_1t_2t_4t_5t_6$ was determined by using the two *invariantly* queries (to be verified in *every* state of the state graph):

```
A[] Decorator.d312456 imply z >= lower_bound        (2)
A[] Decorator.d312456 imply z <= upper_bound        (3)
```

where `lower_bound` is actually the maximum integer which makes true query (1), and the `upper_bound` is the minimum integer which satisfies query (2). The time span of the chosen task which was found to be $[4, 11]$. In a similar way, it was possible to establish the time span of each transition firing (see Table 1). Values in Table 1 fix a few wrong calculations reported in [22]. It should be noted that timing information are relative to the time horizon (`z=0`) when the initial marking of p_0 and p_1 is established.

Similar queries were used for checking cases 2. and 3. of section 2.2. For demonstration purposes, the time span of the model tasks was determined when $m = 3$ (initial marking of $M[10] = 3$) and $n = 2$ (initial marking of $M[11] = 2$). For brevity, only the time span of the three tasks, i.e. the possible firing times of locations `d312456`, `d132456` and `d123456` of Fig. 5, are reported in Table 2. They serve the purposes of answering questions about deadline fulfilment of the manufacturing/production system under loop repetitions. For example, the firing time interval of task $t_0t_1t_2t_3t_4t_5t_6$ was checked by the two queries:

```
A[] Decorator.d123456 imply z <= 55                 (4)
A[] Decorator.d123456 && M[10]==0 && M[11]==0 imply z >= 18   (5)
```

which were found true only for the shown values. Query (4) which determines the lower bound of task termination, ensures that loop repetitions effectively exhausted.

Obviously, verification can also answer querying the time span of individual transition firings, e.g. when the marking $M[10]$ and $M[11]$ become 0, or in intermediate scenarios.

3.3 Sequence patterns and schedulability analysis

In a more general case, the first transition can occur multiple times in the sequence pattern σ whose schedulability has to be investigated. The decorator may

Table 1. Firing times of straight task executions

task/transition	t_0	t_1	t_2	t_3	t_4	t_5	t_6
$t_0t_3t_1t_2t_4t_5t_6$	[0,1]	[2,4]	[3,6]	[2,4]	[4,8]	[4,9]	[4,11]
$t_0t_1t_3t_2t_4t_5t_6$	[0,1]	[1,4]	[2,6]	[2,5]	[3,8]	[3,9]	[3,11]
$t_0t_1t_2t_3t_4t_5t_6$	[0,1]	[1,4]	[2,5]	[2,5]	[3,7]	[3,8]	[3,10]

Table 2. Time span of tasks when $m = 3$ and $n = 2$

task/termination	t_6
$t_0t_3t_1t_2t_4t_5t_6$	[18,56]
$t_0t_1t_3t_2t_4t_5t_6$	[18,56]
$t_0t_1t_2t_3t_4t_5t_6$	[18,55]

find itself in a situation where it has already recognized a prefix of the sequence and now detects an unexpected transition. It may happen that the concatenation of this prefix with the unexpected transition constitutes a sequence that has a suffix which is a prefix of σ. For example, suppose that $\sigma = t_0t_1t_0t_2t_3$ and that the following sequence $\omega = t_0t_1t_0t_1t_0t_2t_3$ occurs. When the decorator automaton meets the second occurrence of t_1, it knows that the partial sequence $t_0t_1t_0t_1$ is not a prefix of σ, however it must not restart the recognition of σ from scratch because t_0t_1, which is a suffix of the encountered part of ω, is also a prefix of σ. In this situation the decorator must restart its operation from the point it met the first occurrence of t_1. This is a common issue in pattern matching problems and efficient algorithms for dealing with these situations are well-known in the literature, e.g. the Knuth-Morris-Pratt algorithm [13]. The decorator may be easily adapted so as to be able to recognize such type of sequences, thus enabling schedulability analysis and the evaluation of the time span of sequences (tasks). When the first transition of a sequence pattern may appear $n > 1$ times, the usage of one clock, which is reset each time the decorator detects the start transition, is not enough for the correct computation of the time span of the sequence. In these cases it is necessary to resort to a clock queue. Each time an occurrence of the first transition is detected, a clock must be reset and put at the end of the queue. When an unexpected transition is encountered, a number of clocks, which is equal to the discarded occurrences of the first transition, must be removed from the head of the queue. At the time the last transition gets matched, the duration of the sequence pattern corresponds to the value of the clock at the head of the queue. Obviously, the maximum length of the queue is equal to n. For this reason the queue can be easily implemented as a circular array of n clocks. From the sequence pattern it is possible to design a non deterministic automaton (NDA) for accepting sequences containing the pattern, which has a number of states equal to the length of the pattern plus one. For example, the NDA depicted in Fig. 6 corresponds to the sequence pattern $\sigma = t_0t_1t_0t_2t_3$.

Although the construction of a deterministic automaton corresponding to a non deterministic one is always possible, in the general case the number of states of the resulting automaton is exponential in the number of states of the original automaton. In the case of NDA arising in sequence-matching, though, the construction of the equivalent deterministic automaton can be done efficiently and the resulting automaton has the same number of states as the original one [14]. Fig. 7 is a TPN model of an assembly subsystem where two raw parts A and B are concurrently processed by Processor 1 and Processor 2, respectively modelled by transitions t_0 and t_2. The processing of part A is simpler than that

of part B and usually terminates before. After Processor 1 completes its job, the production system waits for a certain amount of time for the part B to be ready. If the processing of part B delays too much, part A is sent again to Processor 1 for a refinement. Part A may be refined a maximum number of times before part B is ready. When both parts are ready, they get assembled together (firing of transition t_3). The timing constraint which governs the refinement of part A is modelled by transition t_1. Token generation in place p_4 corresponds to the maximum allowed number of refinements of part A.

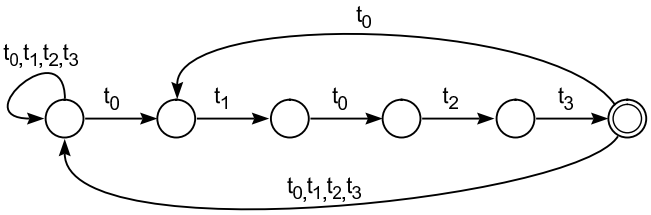


Fig. 6. Non deterministic automaton accepting $\sigma = t_0t_1t_0t_2t_3$

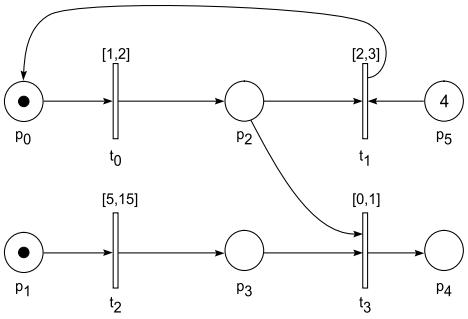


Fig. 7. A FMS/TPN model which admits sequence pattern $\sigma = t_0t_1t_0t_2t_3$

A firing sequence that contains the pattern $\sigma = t_0t_1t_0t_2t_3$ happens when the refinement of part A occurs at least once before the two parts are assembled and the last refinement completes before part B is ready. In order to evaluate the time span of such a sequence, a decorator automaton based on the deterministic automaton corresponding to the NDA of Fig. 6 has to built. Fig. 8 shows in a simplified way the decorator TA for the problem at hand. Since the first transition (t_0) occurs twice in σ, an array of two clocks must be introduced. As one can see from Fig. 8, an integer variable s is used to store the index corresponding to the head of the clock queue. When transition t_1 is detected and the decorator finds itself in location d010, only one clock must be discarded and this is imple-

mented by modular increment of variable s. In all the other locations when the sequence restarts, all the clocks must be discarded.

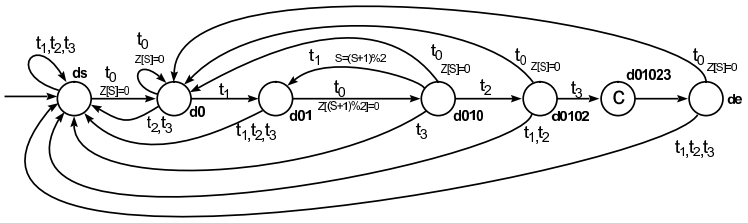


Fig. 8. Decorator for handling sequence $\sigma = t_0t_1t_0t_2t_3$

According to the timing constraints reported in Fig. 7, the sequence pattern $\sigma = t_0t_1t_0t_2t_3$ may not occur. This is witnessed by the fact that the following property is found not satisfied by the model checker:

```
A[] M[4]==1 imply (Decorator.d01023||Decorator.de).
```

However, the following query

```
E<> M[4]==1 && (Decorator.d01023||Decorator.de)
```

returns true mirroring the fact that the pattern σ can really occur. The time span of σ can be computed by respectively finding the maximum value of `k1` and the minimum value of `k2` which satisfy the following properties:

```
A[] (M[4]==1 && Decorator.d01023) imply z[s] >= k1
A[] (M[4]==1 && Decorator.d01023) imply z[s] <= k2
```

For the TPN model in Fig. 7 the time span of σ is the interval $[3, 8]$.

4 About the Approach

The approach described in this paper is capable of exploiting the compact data structures and efficient algorithms of Uppaal. A critical factor is the number of clocks required by a translated TPN model. Indeed, the verification process is exponential in the number of clocks. However, although each transition is associated with a clock, it is the number of simultaneously active clocks which concretely affects space/time complexity of the model checker. In other words, it is the concurrency degree of the TPN model, i.e. the number of simultaneous enabled transitions, which dictates the number of active clocks and then the efficiency of model verification.

The tool combination TPN Designer-Uppaal was compared to similar combination Romeo-Uppaal proposed in [8], so as to figure out some performance index when the tools are applied to "large" models. The same models used in [8] for comparing Romeo-Uppaal with the direct use of Tina and Romeo alone, were experimented with TPN Designer-Uppaal. Gros1, Gros2 and Gros3 in Table 3 are benchmark models contained in Romeo distribution v2.6.3.

Particularly for the large models, the approach of this paper outperformed considerably that based on Romeo-Uppaal. After an analysis of unexpected not so good behaviour of Romeo-Uppaal it soon emerged that a poor management of partial order, i.e. of non determinism during the disabling process of transitions, was the cause of bad performance and not scalability of translation from Romeo to Uppaal.

Table 3 reports wall clock time and memory requirements of a few experiments for the three cases: Romeo-Uppaal original, Romeo-Uppaal modified (partial-order problem fixed by us), TPN Designer-Uppaal. Performance indexes rely on the memtime facility. All the experiments were carried out on a Linux platform running kernel 2.6.20, on a Pentium IV 3.4GHz, 1GB RAM. For each model, verifyta was launched with the query `A[] !deadlock` whose satisfaction forces the generation of the whole state graph. The production cell model [10] was also chosen because it exhibits a good concurrency degree. The model was simply scaled by replicating the base model.

Table 3. Tools comparison experimental data (Uppaal version 4.0.6)

	Romeo-Uppaal		Romeo-Uppaal-mod		TPN Designer-Uppaal	
	Time (sec.)	Memory (kB)	Time (sec.)	Memory (kB)	Time (sec.)	Memory (kB)
Gros1	363.44	168520	18.89	40848	17.41	40344
Gros2	2297.33	798488	31.96	42016	27.85	41212
Gros3	-	out of memory	46.43	42992	39.70	43040
ProdCell 1 instance	1.98	48460	0.11	2608	0.10	2608
ProdCell 2 instances	-	out of memory	5.48	39448	6.70	39448
ProdCell 3 instances	-	out of memory	1618.92	335108	1579.53	197776

The experiments confirmed that the approach based on TPN Designer-Uppaal is of practical use. The tool behaves similarly to Romeo-Uppaal modified but seems to scale up a little bit better. All of this can depend on the fact that broadcast synchronizations during completion of a transition firing are embedded in the automaton in Fig. 3 whereas they are part of a separate Supervisor automaton in [8]. Scenarios where both tools are difficult to apply are in any case out of scope of both Tina and Romeo which rest on "off-line" generation of the time reachability graph. For very complex models, where exhaustive verification is almost impossible, simulation techniques like those described in [10] can be applied to estimate timing properties.

5 Conclusions

This paper proposes an original approach to schedulability analysis of real-time systems specified by Time Petri Nets (TPNs). The approach is based on a translation of a TPN model on to the popular UPPAAL tool, assisted by the TPN DESIGNER toolbox [7]. The approach, similarly to the method described in [22], permits one to investigate behavioural and timing properties of transition sequences (execution tasks). In addition, this paper shows how the approach can be generalized to the handling of sequence patterns in which the first transition can occur multiple times and where (possibly) the recognition process requires to be restarted while ensuring timing analysis. On-going work is geared at

- extending TPN DESIGNER so as to offer TCTL queries directly at the TPN level, thus allowing the modeller to reason in TPN terms and hiding the underlying use of the UPPAAL model checker engine
- applying the approach to the timing analysis of event scenarios in real-time actor systems, e.g. [11]. The idea is to model message exchanges of an actor computation by TPNs and then to investigate event timing constraints so as to guide the runtime behaviour of the scheduler.

References

1. R. Alur, C. Courcoubetis, and D. Dill. Model-checking for real-time systems. In *Proc. of Fifth Annual IEEE Symposium on Logic in Computer Science (LICS'90)*, pages 414–425, 4-7 June 1990.
2. R. Alur and D. L. Dill. A theory of timed automata. *Theoretical Computer Science*, 126(2):183–235, Apr. 1994.
3. G. Behrmann, A. David, and K. G. Larsen. *Formal Methods for the Design of Real-Time Systems*, chapter A Tutorial on UPPAAL, pages 200–236. Number 3185 in LNCS. Springer, 2004.
4. B. Berthomieu and M. Diaz. Modeling and verification of time dependent systems using time Petri nets. *IEEE Transactions on Software Engineering*, 17(3):259–273, March 1991.
5. B. Berthomieu, P.-O. Ribet, and F. Vernadat. The tool TINA—Construction of abstract state spaces for petri nets and time Petri nets. *International Journal of Production Research*, 42(16):2741–2756, 2004.
6. G. Bucci and E. Vicario. Compositional validation of time-critical systems using communicating time Petri nets. *IEEE Transactions on Software Engineering*, 21(12):969–992, Dec. 1995.
7. L. Carullo, A. Furfaro, L. Nigro, and F. Pupo. Modelling and simulation of complex systems using TPN Designer. *Simulation Modelling Practice and Theory*, 11(7-8):503–532, Dec. 2003.
8. F. Cassez and O. H. Roux. Structural translation from time Petri nets to timed automata. *Journal of Systems and Software*, 79(10):1456–1468, Oct. 2006.
9. F. Cicirelli, A. Furfaro, and L. Nigro. An approach to protocol modeling and validation. In *Proc. of 39th Annual Simulation Symposium (ANSS'06)*, pages 261–268, 2-6 April 2006.

10. F. Cicirelli, A. Furfaro, and L. Nigro. Distributed simulation of modular time Petri nets: An approach and a case study exploiting temporal uncertainty. *Real-Time Systems*, 35(2):153–179, Feb. 2007.
11. G. Fortino, L. Nigro, F. Pupo, and D. Spezzano. Super actors for real time. In *Proc. of 6th IEEE CS Workshop on Object-oriented Real-Time Dependable Systems (WORDS'01)*, pages 142–149, 2001.
12. G. Gardey, D. Lime, M. Magnin, and O. H. Roux. Romeo: A tool for analyzing time Petri nets. In *Proc. of 17th Int. Conf. on Computer Aided Verification (CAV'05)*, number 3576 in LNCS, pages 418–423. Springer, 2005.
13. D. E. Knuth, J. H. Morris, and V. R. Pratt. Fast pattern matching in strings. *SIAM Journal on Computing*, 6(2):323–350, 1977.
14. H. R. Lewis and C. H. Papadimitriou. *Elements of the Theory of Computation*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 1997.
15. M. Lindahl, P. Pettersson, and W. Yi. Formal design and analysis of a gear controller. *International Journal on Software Tools for Technology Transfer (STTT)*, 3(3):353–368, Aug. 2001.
16. P. Merlin and D. Farber. Recoverability of communication protocols–implications of a theoretical study. *IEEE Transactions on Communications*, 24(9):1036–1043, September 1976.
17. T. Murata. Petri nets: Properties, analysis and applications. *Proceedings of the IEEE*, 77(4):541–580, April 1989.
18. C. Ramchandani. *Analysis of asynchronous concurrent systems by timed Petri nets*. PhD thesis, MIT, Cambridge, MA, 1974.
19. ROMEO - A tool for time Petri net analysis. `http://romeo.rts-software.org/`.
20. TINA - TIme petri Net Analyzer. `http://www.laas.fr/tina/`.
21. UPPAAL. `http://www.uppaal.com`.
22. D. Xu, X. He, and Y. Deng. Compositional schedulability analysis of real-time systems using time Petri nets. *IEEE Transactions on Software Engineering*, 28(10):984–996, Oct. 2002.

Proceedings of the International Multiconference on Computer Science and Information Technology pp. 837–847
ISSN 1896-7094
© 2007 PIPS

Improving Dependability of Automation for Free Electron Laser FLASH

Bogusław Kosęda[1], Tomasz Szmuc[2], Wojciech Cichalewski[1]

[1] Technical University of Łódz,
Department of Microelectronics and Computer Science,
Al. Politechniki 11, 90-924 Łódz, POLAND
[2] AGH University of Science and Technology, Department of Automatics
Al. Mickiewicza 30, 30-059 Kraków, POLAND

Abstract. Free-electron laser FLASH (260-meter-long machine) [1] is a pilot facility for the forthcoming XFEL [1] (3 km) and ILC [2](≃35 km) projects. Along with growth of the experiment, service and maintenance are becoming so complex that certain degree of automation seems to be inevitable. The main purpose of the automation software is to facilitate operators with computer-aided supervision of several hardware/software subsystems. The efforts presented in this contribution concern elaboration of general framework for designing and development of automation software for the FLASH. The toolkit facilitates specification, implementation, testing and formal verification. The ultimate goal of the framework is to systematize the way of automation software development and to improve its dependability. At present usefulness of the tools is being evaluated by testing the automation software for single RF-power station of the FLASH.

0.1 Safety-Critical Aspects of Application

The peculiarity of application implies number of requirements typical for safety-critical applications. One of the most important feature of customer-oriented facility as FLASH is machine uptime. It entails automation software liveness. Moreover, the software deals with expensive hardware installations, which are now and then serviced by technical personnel. Under this circumstances assertion of various safety properties is evident.

0.2 Insufficiencies of Previous Frameworks

Several attempts to automate certain subsystems of the FLASH have been performed at the DESY [4–7]. All of them utilized the DOOCS [3] Finite State Machine [8, 4] toolkit or Stateflow [9].

Authors' practice reveals that successful applications of simple automation schemes are feasible but design of statemachines for larger subsystems turns out to be tedious and error-prone. The problem becomes particularly evident when specification evolves and design has to be updated. Then, even well elaborated statemachine becomes a mixture of complex expert's knowledge and tricky

endeavors which make it work. Both aforementioned toolkits offer merely the implementation tools. They do not facilitate stages of specification testing and verification.

1 Proposed Approach

To address requirements of application domain, several mechanisms borrowed from expert-systems field have been used. Proposed software consists of two execution engines supplied with the specification in the domain-specific language. The planner engine assembles plans to drive the subsystem automatically towards desired operation mode. The exception handler is designed to deal with possibly complex exceptional situations, which may be exposed by driven hardware. Its role is to fix known operation glitches and perform conflict resolution in the case of multiple exceptions.

2 The Architecture

Single installation of the automation software consists of two runtime automation engines and two specification files. Cooperation of the engines is realized by dedicated cooperation protocol.

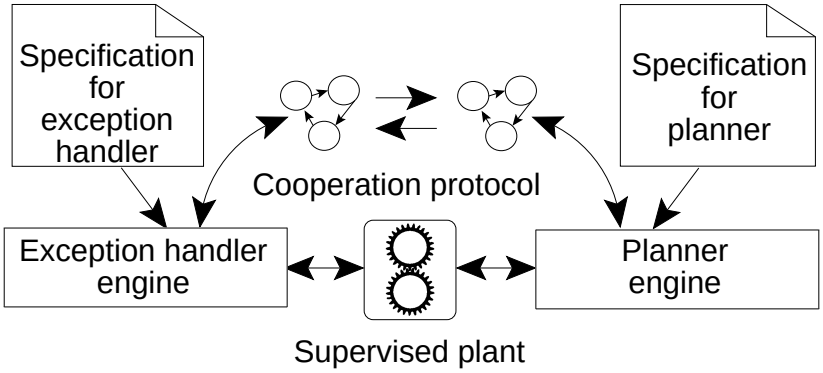


Fig. 1. Single installation of the automation software.

2.1 Planner Engine

Its role is to automate routine operation procedures usually performed by the operators. It consists of specification language interpreter, state estimator, planner and plan executor. State estimator retrieves current status of supervised accelerator subsystem. Planner synthesizes a sequence of procedures bringing the

system from active state to the state satisfying specification of target operation mode. Plan executor takes care for executing a single procedure. The specification for the planner engine is comprised of constructs presented by the grammar from Fig. 2. A state space of a finite state description is represented by set of system variables (`<qvariable>`) with significantly reduced domains. Physical signals readouts are introduced to the specification by means of `<observable>`. Mapping between the model and hardware readouts is accomplished by definition of system variables domains. Possible model state transformations are expressed by means of atomic operations (`<procedure>`). Their specification consists of precondition, postcondition, reference to the executable code and estimate of execution time. Procedure is permissible only if its precondition evaluates to true. Postcondition becomes fulfilled after its successful execution. Execution time helps in estimation whether the procedure is still in progress or has presumably failed. Since every automated operation is performed on purpose, there is a way to specify possible goals of automation. For these purpose there exists a construct `<op_mode>`. It specifies a valuation of subset of system variables which must hold for the operation mode to be active. Specification can be augmented with definitions of formal properties of the model (`<formal_prop>`). The only usage scenario of the planner engine is to configure target mode and let the software bring the subsystem there. This process executes in cycles. Every cycle the state estimator guesses the status of supervised device, planner fins the sequence of atomic procedures driving the system into target operation mode and plan executor performs first procedure from the plan. After reaching the target operation mode, planner engine restricts itself to monitoring. In the case of single procedure failure several scenarios depending on plan executor setup may happen. At present there are two setups possible. First repeats failed procedure while the second tries to find and execute alternative procedure.

2.2 Handler of Exceptional Events

Exception handler recognizes operation glitches and if possible executes appropriate remedy procedures. If exception cannot be dealt with automatically, it stops the automation software and warns the operators. In the case of multiple exceptions it must choose the most suitable remedy procedure.

Its specification language is designed for definition of exceptional situations. They are described by means of conditions defined in terms of monitored DOOCS properties[3]. There are distinguished three categories of the exceptions. Permanent faults, temporary interrupts and warnings. Faults cause permanent break in machine operation. Interrupts are temporal glitches which can be automatically dealt with. Warnings provide information about possibly approaching operation problems.

Above classification was introduced to facilitate conflict resolution in the case of multiple exceptions occurrence. If a fault occurs, automation software is permanently suspended and appropriate message is sent to operators' console.

[3] Corresponding grammar may be found in Fig. 2

```
***************** COMMON PART OF THE LANGUAGE *****************
<specification> ::=     {def <definition> ";"}
<definition>    ::=     <rep_exceptions> | <exception> | <observable> |
                        <procedure> | <q_variable> | <op_mode> | <formal_prop>

<observable>    ::=     observable <obs_name> of type <obs_type>
                        taken from <doocs_addr>
<obs_type>      ::=     bool | int | float | string

<condition>     ::=     <relation> <junction> <relation>
<junction>      ::=     & | "|"
<relation>      ::=     <obs_name> <operator> <number> |
                        <obs_name> <bitop> integer <operator> integer
<operator>      ::=     == | != | < | > | <= | >=
<bitop>         ::=     bitor | bitand | bitxor
<number>        ::=     integer | float

<proc_name>, <obs_name>, <q_name>, <q_val>, <op_name>, <p_name>,
<doocs_addr>, <description>, <message> ::= string


******************** PLANNER-SPECIFIC PART ********************
<q_variable>    ::=     qvariable <q_name> <q_domain>
<q_domain>      ::=     <q_value> {"," <q_domain>}
<q_value>       ::=     value <q_val> if <condition>

<op_mode>       ::=     opmode <op_name> active when <state>
<state>         ::=     <state> { <junction> <state> }
<state>         ::=     <qrelation>
<qrelation>     ::=     <q_name> == <q_val> | <q_name> != <q_val>

<procedure>     ::=     procedure <proc_name> description: <description>
                trigger: <doocs_addr> allowed <condition_type> <condition>
                        postcondition: <condition> cost: integer
<condition_type>::=     unless: | when:

<formal_prop>   ::=     specification <p_name> <p_type> <state>
<p_type>        ::=     always | never | possible


*************** EXCEPTION HANDLER-SPECIFIC PART ***************
<rep_exceptions>::=     report <exc_type> to <doocs_addr>
<exc_type>      ::=     fault | warning | interrupt
<exception>     ::=     <interrupt> | <fault> | <warning>

<interrupt>     ::=     interrupt <description> holds if <condition>
                        report message <message> execute procedure <proc_name>
<fault>         ::=     fault <description> holds if <condition>
                        report message <message>
<warning>       ::=     warning <description> holds if <condition>
                        report message <message>
```

Fig. 2. Grammar defining syntax of the specification language for both the planner and the exception handler.

Occurrence of an interrupt in case of lack of faults entails execution of suitable remedial procedure.

Conflict resolution between interrupts is based on calculation of subsumption relation. More strictly specified interrupts have precedence before more general ones. The algorithm for deciding whether one exception subsumes another utilizes two constraint solvers. The *clp/bounds* [16] and the *clpqr* [16]. The idea of calculating the relation is fairly simple. If one assumes two exceptions E_1 and E_2 which conditions *cnd_1* and *cnd_2*. The algorithm reports the subsumption if there exist three valuations V_1, V_2, V_3 of variables (hardware readouts) in *cnd_1* and *cnd_2* meeting one of the following statements.

$$E_1 \textbf{ subsumes } E_2 \; iff$$
$$(cnd_1(V_1) \; \wedge \; cnd_2(V_1)) \; \wedge \; (cnd_1(V_2) \; \wedge \; \neg cnd_2(V_2))$$
$$\wedge \; \neg(\neg cnd_1(V_3) \; \wedge \; cnd_2(V_3))$$

$$E_2 \textbf{ subsumes } E_1 \; iff$$
$$(cnd_1(V_1) \; \wedge \; cnd_2(V_1)) \; \wedge \; (\neg cnd_1(V_2) \; \wedge \; cnd_2(V_2))$$
$$\wedge \; \neg(cnd_1(V_3) \; \wedge \; \neg cnd_2(V_3))$$

When above conflict resolution methods fail, the order of appearance in the specification file decides which exception is handled first.

2.3 Cooperation Scenarios

Both the runtime engines perform complementary tasks. Since they share the same hardware equipment, they must obey certain rules of cooperation. For this purpose a protocol orchestrating their collaboration has been designed. General diagram of the cooperation protocol design is presented in Fig. 3. Table 1 explains the interfaces presented in the diagram. Figures 4 and 5 present the design of the cooperation protocol in the form of Harel's statecharts [19]. [4]. Table 2 provides descriptions of the states from the Fig. 4 and 5. Poorly designed cooperation protocol might cause automation software to hang. Therefore it had to be verified for the deadlock [18] and livelock [18] freedom. The SPIN [14] model checker was used for this purpose. Protocol design presented in Fig. 3, 4 and 5 was modeled in the PROMELA[5] language. Then the model has been checked for the existence of deadlocks and livelocks.

It turned out that all non-progressive cycles [14] found in the model did not cause starvation (livelock). They have been marked as progressive by inserting a progress labels depicted as numbered bullets in 4 and 5. After inserting the labels into the model no invalid endstates [14] (deadlocks) have been found. Besides, the model has been verified to conform to its functional requirements presented in Fig. 6.

[4] It is assumed that they are interpreted according to the operational semantics described in the "Stateflow User's Guide" [9]

[5] A modeling language of the SPIN model checker

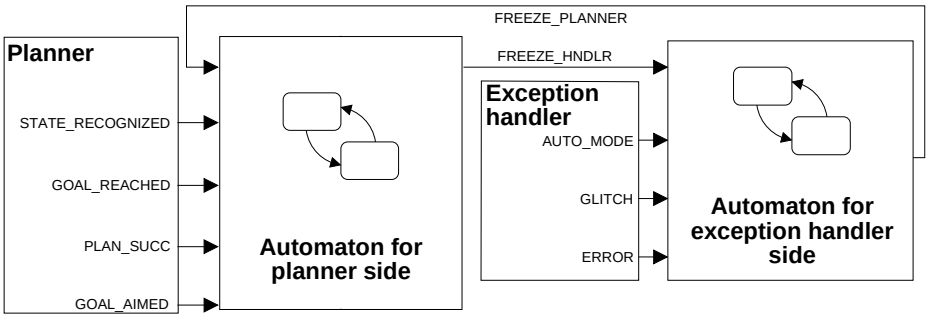


Fig. 3. General scheme of communication between the planner and the exception handler.

Table 1. Explanation of the data supplied to and interchanged between the parts of the protocol from Fig. 3

Stimulus name	Description
STATE_RECOGNIZED	State of supervised plant fits in the state space defined by the specification
GOAL_REACHED	A state of the target operation mode has been reached
PLAN_SUCC	A path to one of the target states has been found
GOAL_AIMED	Target operation mode has been specified
AUTO_MODE	The software is permitted to supervise the plant
GLITCH	Exception handler reports an operation glitch
ERROR	Exception handler reports a permanent fault
FREEZE_PLANNER	Suspend planner
FREEZE_HNDLR	Suspend exception handler

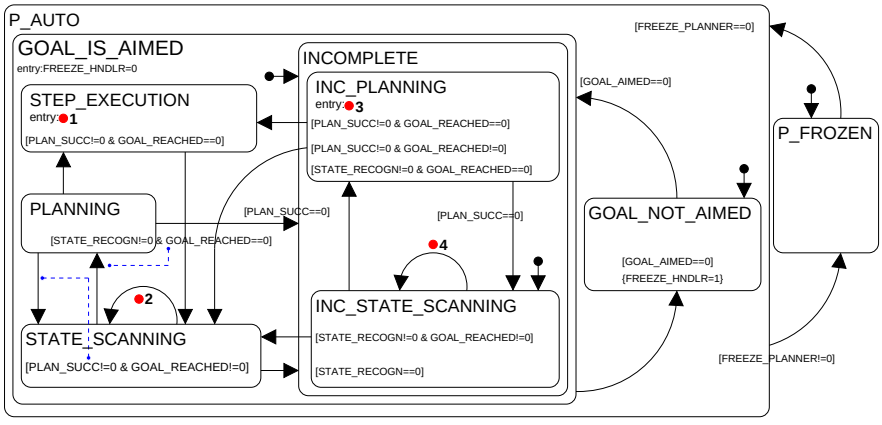


Fig. 4. Design of the communication protocol for the planner.

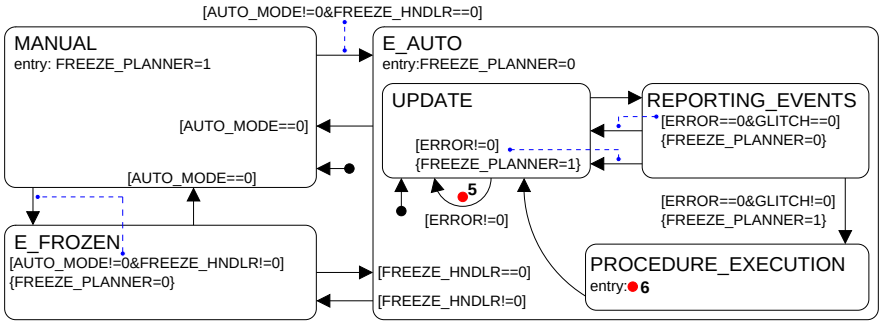


Fig. 5. Design of the communication protocol for the exception handler.

Table 2. Explanation of state names from the Fig. 4 and 5

State name	Description
P_AUTO	Planner is permitted to supervise the plant
P_FROZEN	Planner is suspended
GOAL_IS_AIMED	Target operation mode has been specified
GOAL_NOT_AIMED	Target operation mode is not specified
STEP_EXECUTION	Planner executes single step of the plan
PLANNING	Planning in progress
STATE_SCANNING	Planner performs state recognition
INCOMPLETE	Planner is incomplete
INC_PLANNING	Planning procedure failed
INC_STATE_SCANNING	State of the plant is unknown
E_MANUAL	Both engines are suspended
E_FROZEN	Exception handler is suspended
AUTO	Both engines are permitted to supervise the plant
UPDATE	Exception detection in progress
REPORTING_EVENTS	All exceptions are being reported to the operator
PROCEDURE_EXECUTION	Exception handling procedure in progress

1. $FREEZE_PLANNER \rightarrow \diamond\neg FREEZE_PLANNER$
2. $FREEZE_HNDLR \rightarrow \diamond\neg FREEZE_HNDLR$
3. $GLITCH \rightarrow \diamond\neg P_FROZEN$
4. $ERROR \rightarrow \diamond\neg P_FROZEN$
5. $MANUAL \rightarrow \diamond\neg P_FROZEN$
6. $\neg GOAL_AIMED \rightarrow \diamond GOAL_NOT_AIMED \wedge E_FROZEN$

Fig. 6. Properties of the cooperation protocol which prove its responsiveness and deadlock freedom.

3 Integrated Formal Verification and Testing

In this project, automated formal verification is realized by model checking [12]. The NuSMV [11] is a model checker used to verify formal properties included in the specification for the planner. Dedicated converter translates the model encoded in the specification language to the equivalent model expressed in the NuSMVs input language. Definitions of formal properties which need to be fulfilled by the model are expressed in the Computation Tree Logic (CTL) [12]. Fragmentary example of the NuSMV input specification can be seen in Fig. 7.

```
MODULE systems_state
 VAR
        FORCE_MANUAL_MODE:          {FALSE,TRUE};
        MODULATOR_STATUS:           {LOCKED_FOR_5_MIN,ERROR,OFF,ON};
 ASSIGN
        next(FORCE_MANUAL_MODE):= FORCE_MANUAL_MODE;
        next(MODULATOR_STATUS)  := MODULATOR_STATUS;

MODULE switch_to_manual(st)
ASSIGN next(st.FORCE_MANUAL_MODE):=case
        st.FORCE_MANUAL_MODE = TRUE : FALSE;
        1                   : st.FORCE_MANUAL_MODE;
                                        esac;
MODULE switch_to_auto(st)
 ASSIGN next(st.FORCE_MANUAL_MODE):=case
        st.FORCE_MANUAL_MODE = FALSE : TRUE;
        1                   : st.FORCE_MANUAL_MODE;
                                         esac;
MODULE main
 VAR
        state                       : process systems_state;
        proc_switch_to_manual       : process switch_to_manual(state);
        proc_switch_to_auto         : process switch_to_auto(state);
FAIRNESS
        running

-- reachability of MODULATOR READY
SPEC EF ( state.FORCE_MANUAL_MODE = FALSE
        & state.KLY_INTERLOCK_STATUS = ALL_GREEN
        & state.MOD_INTERLOCK_STATUS = ALL_GREEN
        & state.MODULATOR_STATUS = ON)
```

Fig. 7. Fragmentary specification for the planner automatically translated to the NuSMV input language. The model is an asynchronous statemachine. Its state is described by symbolic variables defined in the module *system_state* and each transition is represented by corresponding module (e.g. *switch_to_manual*). Module *main* instantiates all the processes. Model execution consists in sequential execution of nondeterministically chosen processes.

4 Testing Environment

To facilitate the process of automation design, dedicated software has been implemented. The toolbox allows to simulate continuous or step-by-step execution

of the planner engine. It provides the interface to display and simulate the system state and integrates automatic formal verification. Some elements of the toolbox can be seen in the Fig. 7, 8, 9.

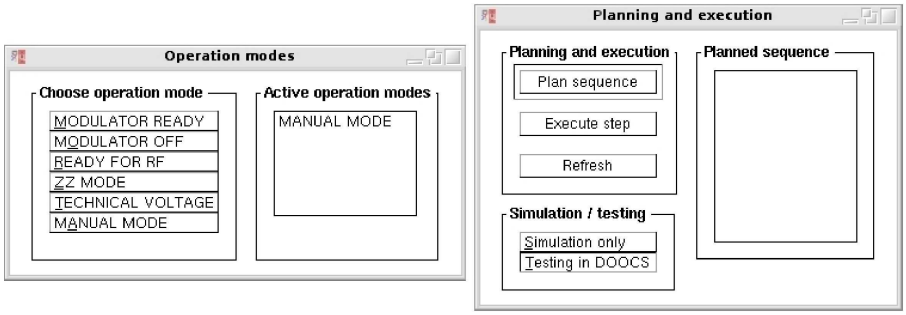


Fig. 8. The interfaces for choosing the target operation and step-by-step simulation.

5 Proof of Concept

Usability of the framework has been evaluated by implementation of supervision software for RF-power station subsystem. This installation is responsible for

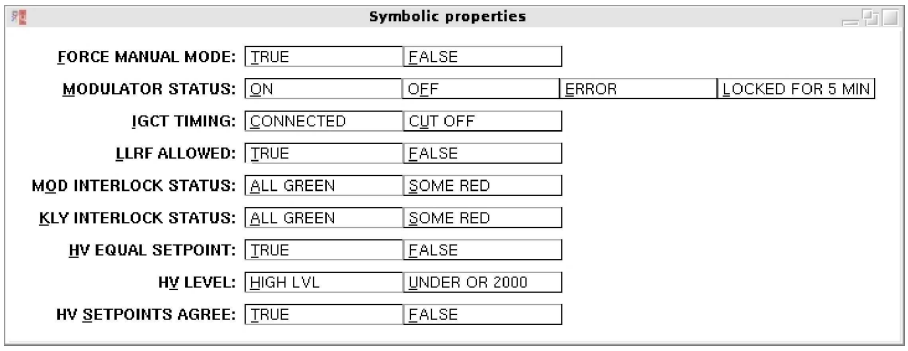


Fig. 9. RF-power station in the "MANUAL_MODE" operation mode.

supplying cavities with energy necessary for particle acceleration. Despite the whole RF-power station is quite complex, it has simple operation scenarios. Six operation modes have been specified. They are presented in Fig. 8. The system state was described by nine system variables depicted in Fig. 9. The exception handler was supplied with the specification of the following exceptions.

- Personal interlock active (personal safety, permanent fault)
- RF-leakage detected (personal safety, permanent fault)
- Unrecoverable modulator fault (permanent fault)
- Modulator power supplier switch is off (human assistance needed)
- Only RF-inhibit activated (remote restart possible)
- General modulator problem (remote restart possible)
- IGCT stack overheated (hardware safety, wait till temperature drops)

The software has been used for several maintenance days for driving the one RF-power station. Both planned and exception handler proved to work correctly.

References

1. Aghababyan A., Altarelli M., et al.: *XFEL The European X-Ray Free-Electron Laser Technical Design Report,* ISBN 3-935702-17-5 (2006).
2. Committee on Elementary Particle Physics in the 21st Century, National Research Council: *Revealing the Hidden Nature of Space and Time: Charting the Course for Elementary Particle Physics,* ISBN: 0-309-66039-4 (2006) 93–100.
3. Hensler O., Rehlich K.: *DOOCS: a Distributed Object Oriented Control System,* Proceedings of XV Workshop on Charged Particle Accelerators, Protvino (1996).
4. Ayvazyan V., Rehlich K., Simrock S., Sturm N.: *Finite State Machine Implementation to Automate RF Operation at the TESLA Test Facility,* Proceedings of the Particle Accelerator Conference, Chicago (2001).
5. Koseda B., Cichalewski W.: *Design and Implementation of Finite State Machine for RF Power Station,* Proceedings of the 12th International Conference Mixed Design of Integrated Circuits and Systems, Cracow, Poland (2005).
6. Koseda B., Cichalewski W.: *Improvements of Expert System for RF-Power Stations,* Proceedings of the 13th International Conference Mixed Design of Integrated Circuits and Systems, Gdynia, Poland (2006).
7. Brandt A., Cichalewski W., Koseda B., Simrock S.: *Automation of low level RF control operation for the VUV-FEL at DESY and future accelerators,* Proceedings of SPIE, Photonics Applications in Industry and Research IV **5948** (2005).
8. Wagner F.: *Modeling Software with Finite State Machines: A Practical Approach,* ISBN 0-8493-8086-3 (2006).
9. MathWorks Inc.: *Stateflow and Stateflow Coder User's Guide.*
10. Brandt A. et al.: *General automation of LLRF control for superconducting accelerators,* Physica C **441** (2006) 263-267
11. Cimatti A., Clarke E., et al.: *NuSMV 2: An OpenSource Tool for Symbolic Model Checking,* Proceeding of International Conference on Computer-Aided Verification, Copenhagen, Denmark (2002).
12. Huth M., Ryan M.: *Logic In Computer Science, Modelling and Reasoning about Systems,* ISBN 0-521-54310-X (2004).
13. Manna Z., Pnueli A.: *The Temporal Logic of Reactive and Concurrent Systems: Specification,* ISBN-10: 0-387-97664-7 (1992).
14. Holzmann G.: *SPIN Model Checker, The: Primer and Reference Manual,* ISBN: 0-321-22862-6 (2004).
15. Wielemaker J.: *An overview of the SWI-Prolog Programming Environment,* Proceedings of the 13th International Workshop on Logic Programming Environments, Katholieke Universiteit Leuven (2003) 1–16.

16. Wielemaker J.: *SWI-Prolog 5.6 Reference Manual* `http://www.swi-prolog.org`
17. Møller R.: *Efficient BDD-Based Planning for Non-Deterministic, Fault-Tolerant, and Adversarial Domains,* PhD thesis, Computer Science Dept., Carnegie Mellon University, CMU-CS-03-139, (2003).
18. *Free On-Line Dictionary of Computing,* `http://foldoc.org/`
19. Harel D.: *Statecharts: A Visual Formalism for Complex Systems,* Science of Computer Programming **8** (1987) 231–274.

Proceedings of the International Multiconference on
Computer Science and Information Technology, pp. 849 – 858
ISSN 1896-7094
© 2007 PIPS

Task jitter measurement under RTLinux operating system

Pavel Moryc, Jindřich Černohorský

Technical University of Ostrava
Faculty of Electrical Engineering and Computer Science
Department of Measurement and Control
Centre for Applied Cybernetics
Ostrava, Czech Republic
pavel.moryc@mittalsteel.com
jindrich.cernohorsky@vsb.cz

Abstract. This paper deals with real-time task jitter measurement under RTLinux operating system. In the first part, it describes methods and tools developed to measure jitter in the RTLinux environment. In the second part, it is focused on discussion of results, obtained on PC hardware, and their interpretation.

1. Introduction: Applicability of Linux in Real-Time Systems

In a real-time system there is a conflict between periodic and aperiodic tasks. Aperiodicity naturally stems from noise, disturbances, delays, and all other unpredictable phenomena in the real world. A real-time operating system should not enforce strict rules that nature cannot meet, but rather provide resources that help to smooth the conflicts. Basic approaches are preemptivity, buffers, and priority rules. However, as a result of preemptivity, a high-priority task requiring a resource may be blocked to wait for medium-priority tasks that do not hold this resource. This problem is called priority inversion.

Linux is neither intended, nor designed to support real-time tasks. It is a general-purpose operating system, implementing full range of API functions covered by the POSIX-1003.1 specification. A kernel providing such large scope of API services cannot meet demands of preemptivity and low latency, required in most technology control systems.

RTLinux kernel implements a Hardware Abstraction Layer inserted between hardware and the Linux kernel. Essentially, it creates a virtual machine, that controls the Linux kernel timer interrupt. The RTLinux can switch between the Linux kernel and other tasks [1, 2], thus making possible to solve conflicts between real-time tasks and the Linux kernel. The Hardware Abstraction Layer (HAL) controlling the system is realized in the Linux kernel space. The system as a whole is simple and fast, but barriers between real-time task and the non-real-time Linux kernel are thin, and as a result of that, the real-time part may easily get out of stability margins required.

Jitter is a variable deviation from ideal timing event. Scheduling jitter is the delay between the time when task shall be started, and the time when the task is being started. Similarly, interrupt latency is a delay between the interrupt is triggered and the time when the Interrupt Service Routine is being started. The interrupt latency varies, and therefore, it is a jitter. Jitters result from physical phenomena in hardware (noise), from concurrent task processing (realized either in hardware or in software), and from passing the code through different branches (each conditional instruction is a potential jitter source). Kernel latency is not stable, but composed from various phenomena, most of them (if not all) showing jitters.

RT-Linux is designed as a module to the Linux kernel, and therefore, it could be reasonably supposed, the RTLinux kernel can suffer from jitters inherited to the Linux kernel. Hence, thoughtful testing is of prime importance.

2. Existing measurement methods

Throughout the time, many jitter measuring methods were developed and published [3, 4]. Periodic task is characterized by its starting time of execution and by the length of execution, while aperiodic task is characterized by its latency. Interrupt latency is defined as the time from generating the interrupt request to (the start of) its service routine execution.

Both theory and experience requires, that a system shall be tested under load. In a technology control system, the load is caused by the specific application requirements. But, whenever results with more general validity are needed, a more general load shall be applied. As a model load, heavy network or disk load is often used. Proctor [3] has performed evaluation of these load types influence on RTLinux operation. Unfortunately, [3] does not include specifications of used hardware.

Published methods of RTLinux evaluation are either concentrated on common techniques of jitter and latency measurement [3] or focused on measuring latencies of basic RTLinux resources [4].

However, above mentioned papers present specific approaches and methods rather than general application studies. The applicability of obtained results to a different design seems to be uncertain.

3. Designed measurement methods

This work presents an approach based on a generalized application program. It is intended as a measuring method tightly connected with application study. Therefore, it can be reasonably assumed, that results will be useful for general engineering practice.

3.1 Generalized data acquisition program

RTLinux in tight connection with Linux is intended for use in systems, that allow both real-time and non-real-time tasks to coexist, and it is typically applied as an interface between dedicated real-time and non-real-time IT levels. As a representative case of this class, a generalized data acquisition program has been chosen, supplied with diagnostic timestamp outputs.

The application, named RT-golem, contains and integrates resources, which make possible to apply a defined workload to the system, as well as to measure how the load task is executed on the system. It implements both periodic and aperiodic tasks.
The RT-golem is a part of an overall architecture presented in Figure 1. The architecture contains and integrates resources, which make possible to apply a defined workload to the system, as well as to measure how the load task is executed on the system. The RT-golem includes

- periodic task, which is controlled by RTLinux scheduler,
- aperiodic task (the Interrupt Service Routine, which is installed instead the default RTLinux ISR routine).

During initial tests performed with the RT-golem it was observed, that excessive IRQ requests can disrupt system operation. For that reason, the RT-golem was strengthened with overload protection, so it can sustain arbitrary input IRQ rates. Based on the test results, a saturation method of measuring interrupt latency was designed, as shown in Figure 2. Interrupts are triggered by a periodic signal supplied from external generator. The incoming interrupt rate is boosted, till the time between two successive ISR routine starts decreases. When the time stops decreasing, the IRQ rate reaches its saturation point. The minimum time between two successive ISR starts equals to the interrupt latency. It consists of latency times caused both by hardware and software resources.

Since the saturation imposes the maximum IRQ rate load on the measured system it is capable to accept, the method is expected to provide comparable results across various hardware platforms.

RT-golem application has been further modified, so it could be loaded more than once. This creates a possibility to load the system by more periodic tasks, each with different priority, period, time of execution and diagnostic timestamps output. It has evolved to a configurable and flexible simulation tool.

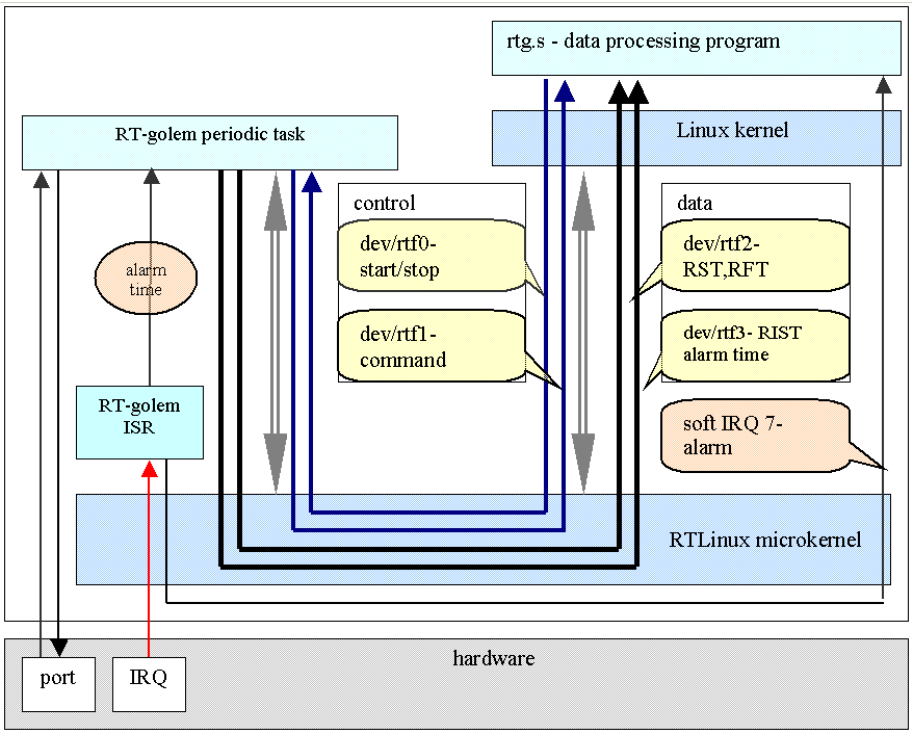


Fig. 1. RT-golem operation

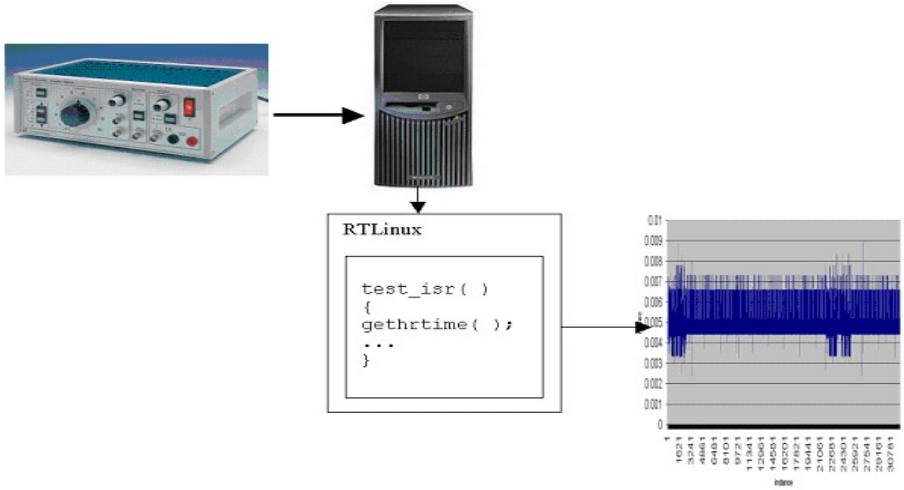


Fig. 2 . ISR latency saturation method

3.2 Advanced measurement tool: RT-golem

Analysing the measurement results and RTLinux resources, it has been recognized, that a measurement tool, which encompasses the whole range of typical RTLinux resources and provides a deeper insight, is needed. Based on this analysis, the following important RTLinux characteristics have been identified:

- precision of the scheduler (measured as task starting time jitter),
- interrupt latency time,
- execution time of typically used API services,
- pipe write and read operations,
- shared memory write and read operations,
- thread switching time
- I/O port read and write access time.

The I/O access is also included, because it characterizes hardware, and presents the basic method of communicating with both sensors and actuators.

The generalized application was substantially redesigned to form an advanced measurement tool. The advanced version of RT-golem consists of a periodic task, and an interrupt service routine. The periodic task includes two threads. It is possible to set priority and period of both threads, as well as to disable one or more parts of the task. This way, it is possible to balance the workload, that the RT-golem imposes on the system.

4. Experimental setup

A set of comparison measurements has been performed. In particular, the effects of different load on different test systems have been measured, as follows:

- no load,
- load with copying files
  (while [true]; do cp /bin/bash ${f}; done),
  /bin/bash is ca. 70kB in length,
- load of 15 RT-golem 5.1 tasks
  on two test systems,
- PC Dell GX 280,
- PC no name.

The source code of the RTLinux scheduler contains a comment [5] recommending that this scheduler should not be used for more than 10 tasks. For verification of this recommendation, an experiment was designed, where the system is heavily loaded by fifteen RT-golem tasks, and jitters of RT-golem test task are measured. The fifteen tasks have been configured as maximum acceptable load for the system, that is, the highest load, at which the Linux kernel yet does not start reporting lost timer interrupts.

PC Dell is a workstation designed for graphical applications, while PC noname is a low-cost, low-end personal computer. Linux kernel has been configured to use only 64 MB of RAM memory. Test system configurations are presented in Table 1.

Table 1 . Test system details

PC DELL GX 280

CPU	Intel P4 3.0 GHz, 1 MB L2 cache
RAM	1024 MB
HDD	SAMSUNG SV0842D, SATA, 75GB
	WDC WD800JD-75JNC0, 8 GB, ATA-66

PC no name

CPU	Intel P4 2.4GHz, 32 K of L1 cache
mainboard	MSI 865 PE Neo2-P
RAM	256 MB
HDD	Seagate Barracuda ST380011A 80 GB ATA-100
	Maxtor WDC WD100EB-00BHF0 10 GB ATA-66

5. Experimental results

Because of limited space, only a handful of results can be presented. The first series of graphs, presented in Figures 3 through 6, show the task instance starting (or finishing) time, while the second series of graphs (presented in Figures 7 and 8) shows the statistical data. The task instance starting time is calculated from the previous task instance starting time. This means, the starting time delay impacts two adjacent values. First, the difference between the correct and delayed instance is longer, which causes the spike up on the graph, and then, the difference between the delayed and next correct instance is shorter, which causes the spike down. If both spikes are symmetrical, the second value is okay. Finishing time is calculated from task instance starting time.

Spikes on the relative starting time graphs below oscillate around 1 msec, because they show scheduling jitter, that means, a difference of the actual relative starting time from the nominal value, which is 1msec.

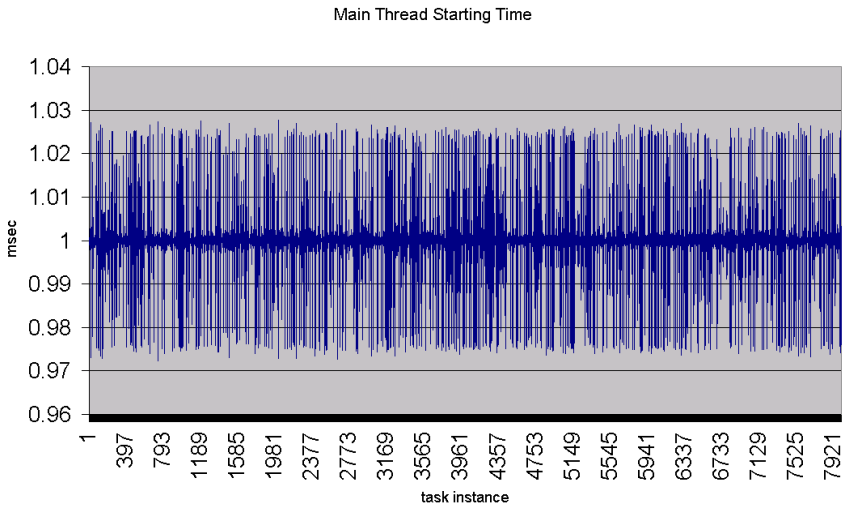


Fig. 3. Periodic task starting time, PC Dell, loaded with copying files

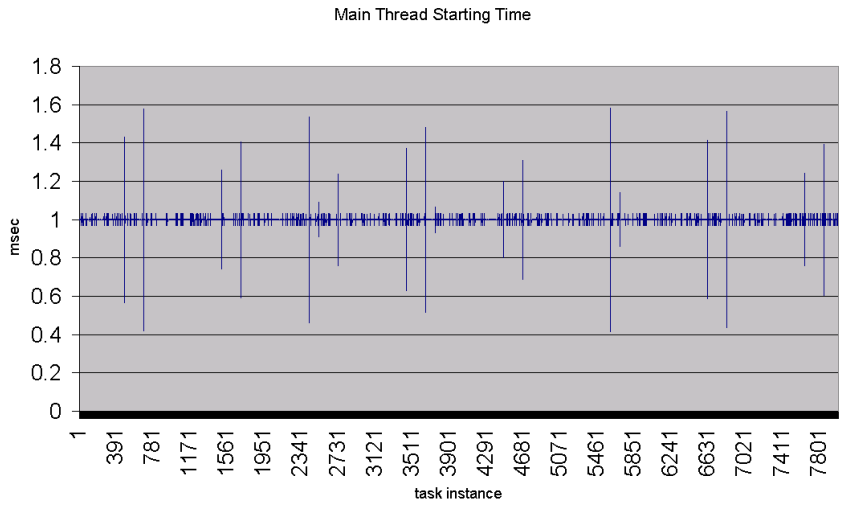


Fig. 4. Periodic task starting time, PC no name, loaded with copying files

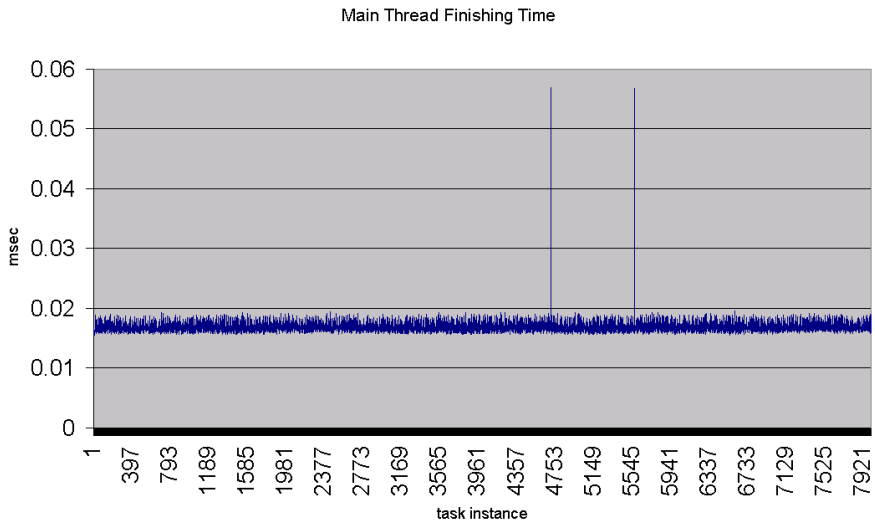


Fig. 5. Periodic task finishing time, PC Dell, loaded with copying files

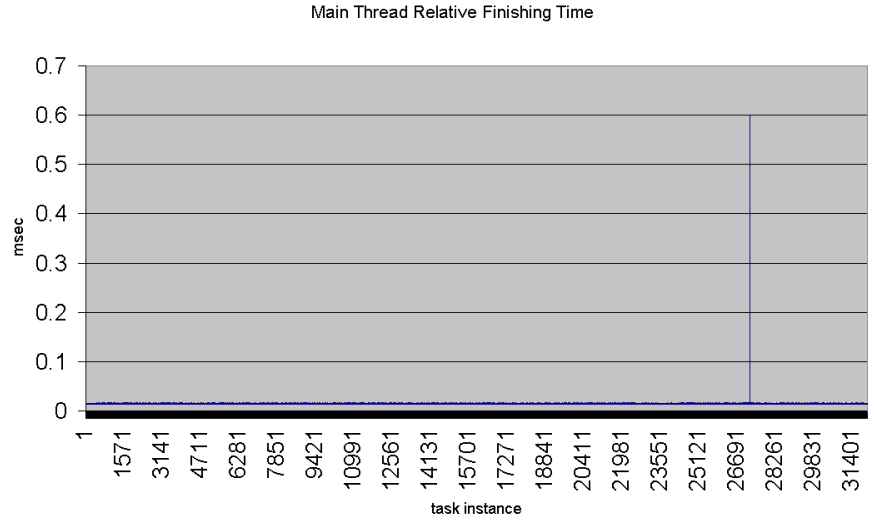
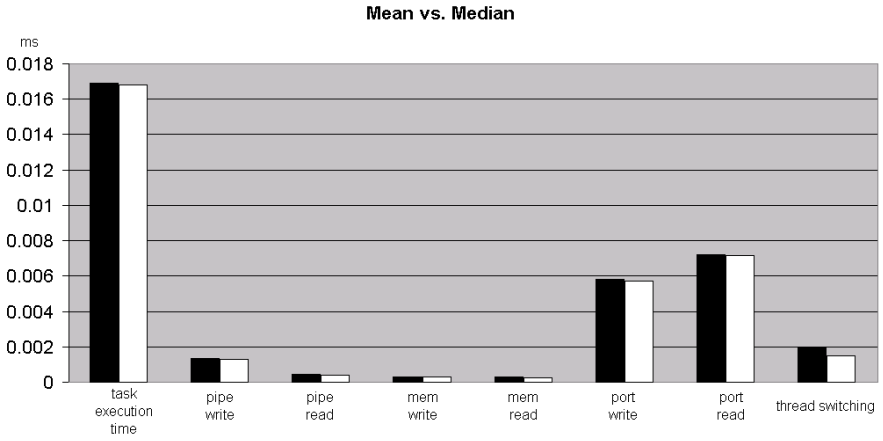


Fig. 6. Periodic task finishing time, PC no name, loaded with copying files

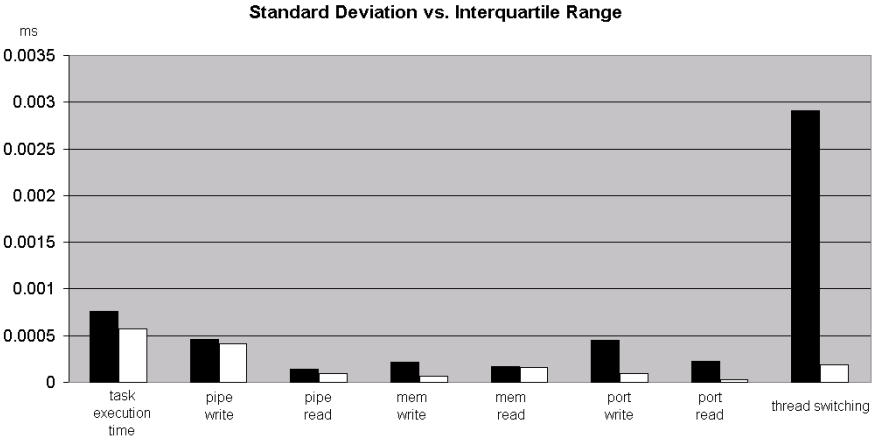
The graphs show that the RT-golem runs smoother on the Dell PC (Figures 3 and 5), than on the no name PC (Figures 4 and 6). To evaluate the outlying values, figures 7 and 8 show the mean vs. median comparison, as well as standard deviation vs. interquartile range comparison. It stems from definitions of mean and standard deviation, that they are more impacted by outlying values than median and interquartile range.



dark: mean

light: median

Fig. 7. Execution time means vs. medians. PC Dell, loaded with copying files



dark: standard deviation

light: interquartile range

Fig. 8. Execution time means vs. medians. PC Dell, loaded with copying files

6. Conclusion and future work

From the results one can conclude that the heavy hard disk operation [3] imposes more load on the system than the real-time tasks load. As the load resulting from hard disk operation is quite common in the system according to POSIX 1003-13 PSE 54 profile, it can be concluded, that

- the scheduler manages to handle more tasks than is was presumed by its authors,
- the Linux kernel operation significantly influences the real-time task jitters.

There is a question, whether the real-time characteristics of the system (the measured jitter spikes) could be smoother, if the Linux kernel were ported on a CPU designed for real time.

The hardware is built as a layered structure of basic hardware resources (disk, memory, processor registers, etc.), and following advanced means (instruction queues and priority rules). The advanced resources are basically the same as the resources used in operating system. These higher level hardware resources can be seen as a hardware implementation of the operating system resources.

A CPU designed for real time (DSP) has different architecture than a CPU designed for general purpose application. It is unlikely, that it could optimally support a full range POSIX 1003-1 compliant kernel.

Based on performed RTLinux and Linux kernel analysis, as well as on measured results, it can be reasonably concluded that for the RTLinux/Linux operating system, a general-purpose hardware is the optimal hardware platform.

Measurements performed at the level of a real-time task often provide valuable information on task jitters, but only little information on underlying causes. Therefore, it could be useful to create a small and simple HAL layer (module) in the Linux kernel, which intercepts timer interrupt and possibly other hardware means for a moment, and quickly gets and taps the diagnostic information needed. Another possible idea is, that such tool could be integrated into lower (architecture dependent) layer of the RTLinux HAL.

Acknowledgement

This work was supported by the Ministry of Education of the Czech Republic under Project 1M0567.

References

1. FSM Labs Inc.,*Getting Started with RT Linux,* 2001.
2. I. Ripoll, et al.: *WP1: RTOS State of the Art Analysis: Deliverable D1.1: RTOS Analysis,* OCERA, 2002.
3. Proc. SPIE Vol. 4563, Sensors and Controls for Intelligent Manufacturing II, Peter E. Orban, Ed., pp. 10-16, 2001.
4. C. Dougan, Z. Mwaikambo: *Lies, Misdirection and Real-time Measurements,* `http://www.rtl.com` 2004.
5. RTLinux v.3.1 source code, /usr/src/rtlinux-3.1/schedulers/rtl-sched.c

Proceedings of the International Multiconference on
Computer Science and Information Technology pp. 859–869
ISSN 1896-7094
© 2007 PIPS

Real-Time Communication for Large Scale Distributed Control Systems

Mariusz Postol

Department of Computer Network, Technical University of Lodz Poland
postol@zsk.p.lodz.pl

Abstract. Because of their scale, complexity and requirement of expandability, Large Scale Distributed Control Systems (LSDCS) are usually created in a multistep integration process. To succeed, it has to be governed by well-defined information architecture, appropriate communication infrastructure and the supervisory role of the time notion taken into consideration from the very beginning of the design stage. Mutual influence of the architecture and underlying communication is discussed in the paper and a novel systematic design methodology is proposed to greatly reduce the complexity. A dedicated communication component is proposed in this approach. Functionality and scheduling algorithms offered by these components enable to satisfy all the defined prerequisites of the real-time distributed control and design the robust system in a systematic and uniform way. The presented case study proves that the solution not only allows the real-time process requirements to be met, but also is a platform for multi-enterprise collaboration.

1 Introduction

The defining characteristic of a networked control system (NCS) is having one or more control loops closed via a serial communication channel. A distributed control system means that the feedback is provided from a supervisory control system to isolated islands of automation and all of them deal with the same real-time process.

Large scale emphasizes three main features of the discussed type of systems [1, 2, 6, 7]. The first two are especially big responsibility and intrinsic complexity. The third one is a large geographical area the system is spread over. An example is described in Sect. 6.

The first installations of LSDS appeared just after microprocessor technology had started rocketing [1]. In spite of no objections concerning possible improvements to the process management and safety, the number of this kind of applications is incomparably smaller than others. The main reason is the complexity of information architecture determined by time relations and a communication infrastructure.

The article shows that it is possible to reduce complexity and provide a systematic uniform design pattern using the proposed modified relation model

where an additional component provides communication support for the implementation of association between system components. The solution enables normalization of information architecture for all applications and making it independent of the underlying communication links structure.

2 Prerequisites

A very important feature of the real-time systems is their intrinsic reliability [1, 6, 7]. Reliability is crucial because a failure of the computer system could result in an economic disaster or even loss of human lives. Unfortunately, there are no appropriate means to guarantee full reliability, but it can be significantly increased by using well-designed principles and approved common solutions during the system life cycle.

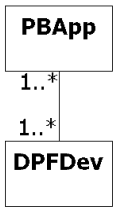


Fig. 1. Basic model

2.1 Structure

Typical simplified hierarchical information architecture [2] for the process industry involves the following levels:

1. Process and Business Management (PBM);
2. Field Management (FM);
3. Controlled real-time process;

The Field Management level consists of smart digital plant-floor devices responsible for control of real-time processes and access data from the plant floor and integrate it into management and business systems. Control and management subsystems located in the upper layer are responsible for controlling the process as a larger whole to gain additional benefits from its optimization. For further discussion it is assumed, that from the PBM layer point of view the FM layer is a sole virtual representation of the underlying layer providing additional functionality and an interface to the upper layer. It means that PBM can reach the target function and achieve full system controllability using the interface provided by the FM layer.

Both layers (1, 2) have a composite character. Therefore, while modeling, we have to distinguish association between components rather than the layers themselves. For applications (PBApp) making up subsystems on the PBM layer and for digital plant-floor devices (DPFDev) on the FM layer many to many association between them must be assigned (Fig. 1).

The previous discussion leads to a very simple design model which becomes incredibly complicated after having been transformed into the implementation diagram (Fig. 2). In this model, we can distinguish vertical and horizontal links responsible for an appropriate inter-layer and intra-layer relationship.

If there is any common part along the link between a device and an upper layer application (port, medium, repeater, etc.) additional precautions must be taken to preserve the integration of transferred data.

Basing on the above discussion, the following prerequisites can be defined to find a systematic design approach:

1. The number of links has to be reduced to keep the complication level of the architecture manageable.
2. Mutually exclusive access to common resources must be guaranteed to deploy architecture presented in the Fig. 1

2.2 Communication

In real world, we need underlying communication to instantiate association, i.e. create a link between objects (devices and applications). Although from the information architecture modeling and design point of view this communication can be considered transparent, its availability and reliability is crucial for the final result. Assuming transparency, it simplifies the problem to a great extent, provided that the assumption is valid.

We need a memory/processing engine unit to instantiate an object and we need a medium to instantiate a link between nodes. But it is insufficient. To transfer data over the medium, we have to use selected protocols controlling access to the medium and data transfer. The protocol is implemented as a component which needs processing power. Medium control and processing consume power. Additionally, protocol and medium often limit bandwidth and medium access. Any of these requirements can cause that the above assumption and, in consequence, this approach becomes unreal. Therefore, we need to look for a compromise between an unacceptable complexity and unreal assumption.

Because of the scale and area of the LSDCS, we cannot even make an assumption that the system is to be designed on a platform of one medium [10]. To transfer the data, we need a medium, but to use the medium, we need to engage an infrastructure: a *technology* (GSM, satellite, ISDN, etc.) governed by technical standards and an *organization* governed by regulations, procedures, practice, etc. A platform optimal today may be useless for future because technology is progressing rapidly and economical standing of organizations may fluctuate.

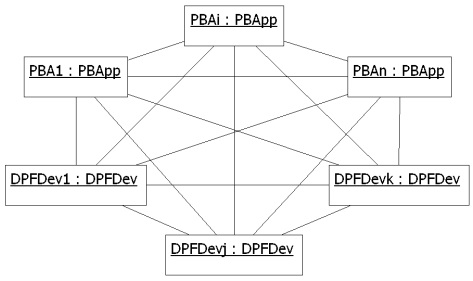


Fig. 2. Basic model implementation

Lack of common medium coverage of the whole area requires engaging simultaneously many communication infrastructures, and dealing with a multidimensional communication network—the next point of freedom. The main advantage

of using many infrastructures is the possibility of improving robustness of the system by providing communication redundancy [11] in overlapping areas.

We need to provide vertical (interlayer) and horizontal (intralayer) communication to follow two-layer information architecture as described above. To make two devices interoperable, both have to use the same (a vendor specific or standard compliant) protocol. Relying on vendor specific solutions limits future system expandability and, therefore, generally it is not recommended and vendors usually offer a standard protocol for plant floor devices. Unfortunately, there are hundreds of "open standards" defined in the automation marketplace. Most of them are dedicated to FM intralayer communication, but they can be also successfully used for interlayer communication.

Traditionally, whenever a software package needed to access data from a device, a custom interface or driver had to be written. To overcome this disadvantage the OPC [3, 8] was designed to bridge applications based on general purpose operating systems and process control hardware and software applications.

The following prerequisites for a systematic approach to deployment of the communication platform for the Large Scale Distributed Control Systems (LS-DCS) can be defined:

1. It has to allow to engage multidimensional networks,
2. It has to provide communication redundancy,
3. It has to conform with a widely accepted standard.

3 Modified architecture

As it was stated in Sect. 2, the main problem we are faced with is the rapidly increasing complexity with the number of elements in the system (Fig. 2). A novel architecture is proposed in Fig. 3 to reduce this complexity. An additional element (PO) was introduced between FM and PBM layer in this architecture. The main function of this element, called *process observer*, is to represent the FM layer as a whole at the boundary of the PBM layer. Two main functions can be distinguished for this element: data and state observation. *Data observer* provides an information mirror of the current values of physical signals gathered from the plant-floor devices and integrated into one homogenous address space of process variables. *State observer* is responsible for computing current values of unavailable signals using available data and a dynamic simulation model. It also integrates obtained results into the same address space. Using the state observer it is possible to improve control quality and build fault-tolerant systems.

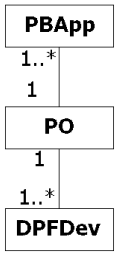


Fig. 3. Modified architecture

Additionally, it will be possible, but not necessary, to reduce the complexity once more if direct intralayer communication is replaced by indirect one through the intermediary PO element.

The approach presented above allows to virtually replace k elements by only one at the PBM layer boundary, and, in consequence, to change the many to many association to two associations of one to many. Finally, we can reduce the number of possible links from (k+n)(n+k-1)/2 to k+n and obtain homogenous architecture.

In the proposed architecture, the PO element is like a bridge connection between the process plant-floor control side and the process and business management side. For the sake of expandability and before deploying architecture presented above, it should be emphasized that the data of plant-floor devices governed by the PO should be communicated to any upper layer client application in a standard way. To provide a universal error free solution, data access has to be defined on an appropriate abstraction level as a set of well defined strongly-typed data structures and method prototypes.

4 Lower layer communication

Employing of an intermediate component as a driver for plant-floor devices is a middleware archetype used worldwide in thousands of applications. A novel lower layer communication model is proposed (Fig. 4) to use it as a consistent sole representation of a distributed real-time process at the PBM layer boundary.

Because it is so important to provide process representation at the PBM layer boundary in a standard way, it is proposed to use a widely accepted OPC [3, 8] interface implemented by *OPC_API. Cache* is an intermediate storage of real-time process data. The *Cache* reflects the latest value of the data (subject to update rate and deadband optimizations) as well as the quality and timestamp. *Controller* holds the plant-floor device data description.

Channel is used to represent independent communication threads conducted simultaneously to each other. Usually, lower layer communication requires multidimensional networks to transfer the real-time process data to/from plant-floor devices. In some cases, even with homogeneous infrastructure, like IP network, we can establish independent simultaneous connections using selected transport protocols (e.g. TCP/UDP). *Channel* concept allows defining as many simultaneous communication paths as it is necessary to meet current needs.

Segment represents a single communication path and is responsible for managing communication resources and data transfer from a group of devices that is to be accessed using the same transport connection. To assure mutually exclusive access to common resources, the *Channel* activates only one *Segment* at any time.

DataProvider is responsible for providing a stream of data to the *Segment*. To provide a consistent process data representation at boundary of the PBM layer, the process observer has to deal with a multidimensional network environment and vendor specific languages (protocols) used by the devices for data access. To meet requirements of this heterogeneous environment, the proposed solution enables to instantiate many custom *DataProviders* by a *Channel* and use them by a uniform data transfer algorithm realized by the *Segment*. Each segment

can use only one *DataProvider*, but one *DataProvider* can be used by many *Segments* associated with the same channel. Because *Segments* are activated sequentially by the corresponding *Channel,* using of *DataProvider*s is safe for common resources.

To provide polymorphism for the environment specific needs, the *DataProvider* is located outside the main software package and inherits an interface ensuring flexible management of the communication medium and transfer of the process data. This solution makes it possible to keep the core software unchanged in all applications and flexibly adapt a skeleton to the vendor or application specific needs.

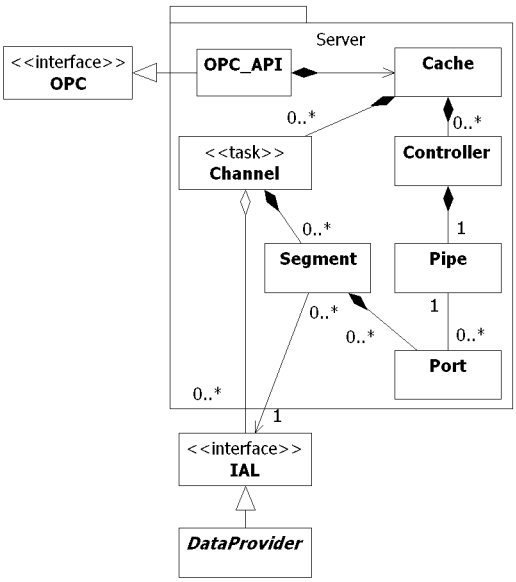


Fig. 4. Lower layer communication model

In a real environment, apart from managing communication networks and accessing plant-floor devices, monitoring and management of the recourses that make up information processing and communication infrastructure are often of the same importance as access to the real-time data. To provide a state observer or commence factory tests of any system, we need to build a simulation environment. With the data provider concept, it is possible to publish all of the mentioned types of information in the same way using the defined interface.

It is generally accepted, that redundancy is indispensable for network control systems. The redundancy of communication is the possibility of defining the excessive communication paths, which can be used to access plant-floor devices. Duplication of the communication paths can be costly, because data transfer over distributed networks is usually not for free. The crucial feature of paths redundancy is the provision of path multiplication without any necessity of transferring the same data over the network many times and controlling the backup path availability at the same time.

In the proposed model (Fig. 4), the *Pipe* concept is used to assure redundancy. *Pipe* is a collection of *Ports*, where only one is active at any time. After detecting a failure of the active path, another *Port* belonging to the same pipe is activated immediately. It represents a bidirectional device data stream. *Segment* creates *Ports* to get access to the data model description held by the *Controller* and after acquiring the process data to deposit it in the *Cache* repository. *Segment* using all associated active *Ports* builds up a data model description to schedule the

sampling operations. *Segment* uses only active *Ports* and, therefore, it is assured that the process data is transferred over the underlying network once only in spite of communication paths multiplication. Behind the scene, the *Pipe* checks availability of non-active paths periodically by changing the state of *Ports*. Using paths redundancy additional spare plant-floor devices can be used seamlessly as the next level of redundancy.

5 Optimization of communication paths utilization

The main job of the communication software is to make 'best effort' to keep the process data fresh and allow clients to access this data randomly. From the communication point of view, two independent communication environments can be distinguished (Fig. 3):

- Connecting FM level plant-floor devices to the intermediate component (PO);
- Connecting the intermediate component (PO) to PBM level clients.

Because both are used to transfer the process data, we have to address the question how these data transfer processes are related to each other. In other words, what a relation should be applied between sampling rates of individual process variables and update rate of clients connected to the server. To answer this question and finally design an appropriate sampling scheduling mechanism on the process side, we need to take the following into consideration:

- Current communication path load and its throughput,
- Current real-time process state,
- PBM level clients needs.

All of them can change in time and, therefore, it is proposed to implement the following two unique closely coupled algorithms allowing to build cost saving, flexible solutions providing process data just in time and preserving communication bandwidth:

- *Optimal Transfer Algorithm* (OTA): responsible for minimizing the difference between requirements of client individual process data update rate and current sampling rate of a process control unit.
- *Adaptive Sampling Algorithm* (ASA): responsible for adjusting the plant-floor devices sampling rate according to the current process state.

The *Segment* is responsible for keeping the process data located in the cache fresh. The scanning schedule managed by the *Segment* can be interrupted by a direct device read/write requests and disturbed by communication errors. Additionally, communication paths redundancy and adaptive sampling algorithm cause that the load of individual communication routes changes in time. Furthermore, we need to deal with "the stupid client" syndrome, where a client unconscious of the limitations requests too high update rate. Having detailed

knowledge of the communication limitations the *Segment* is able to prevent monopolization of the resources by any malicious client.

To address all these issues, the *Segment* implements a weighted fair queuing algorithm. The main job of this algorithm is to harmonize individual sampling rates with clients update rate requests to minimize the difference between both of them.

In the proposed algorithm, the individual process data descriptors are aggregated in groups. Any group has been assigned a quasi static requested sampling rate and is scheduled in the sampling queue as one item. This sampling rate is a function of client requests and limitations set in the configuration. An overload coefficient is calculated for each group, as the ratio of current to requested sampling period. To make the scanning process fair, the scheduling algorithm serves the group with the greatest overload coefficient (provided it is greater then 1) first. As the result, it makes the coefficient approximately equal for all groups assigned to the same *Segment*, i.e. the same communication path, in the steady state. Because the coefficient is monotonically increasing between the sampling instants, it prevents from causing a "starvation" disease, i.e. keeping important data fresh causes that update of less important data is not possible at all. In other words, the algorithm distributes communication bandwidth proportionally to requested sampling rate of the groups.

In some applications we can have another requirement, i.e. a request to keep bandwidth utilization below a preset limit. For example, that is the case when we use toll network. For those systems this algorithm provides additionally the possibility of controlling overload coefficient at the requested level and, in consequence, guarantees bandwidth utilization at the preset level.

Segment is responsible for keeping the cached data fresh, but it is worth stressing that "freshness" of the data depends on the process state and can drastically change in time. Therefore, to minimize the data transfer costs, the sampling rate should be adapted to the current process control needs. It is proposed to use adaptive sampling algorithm to achieve this. The *Port* is responsible for adjusting the requested sampling period of the group while updating current process values associated therewith.

6 Supervisory control of a municipal heating system

In this section, a case study of supervisory control of a metropolitan heating system is provided [4, 5, 9, 10]. The system is located in the city of Lodz – Poland and consists of heat and power plants, backbone pumping stations, backbone heat chambers and local distribution points. The heat distribution network of Lodz (750k citizens) is supplied from three plants with total thermal output of 2560MW. Their optimal utilization requires a control system to allow working on common supplying areas. As the system is distributed geographically (about 800km of pipes), safe communication between nodes (automation islands) is critical. Additionally, a great number of nodes ($\sim$ 8000) requires that the devel-

opment of an appropriate system structure is preceded by a detailed analysis of the availability, throughput, and connectivity standardization.

To meet the requirements, the architecture presented in Sect. 3 is used, where the *process observer* is implemented as an OPC server [4]. It has to manage data transfer using some independent networks: VHF, GPRS, leased and dedicated fiber optic and corporate multipurpose field segments. Additionally, because of the propagation limits several transmitting on the same bandwidth locations have to use. The biggest challenge is to synchronize transmission on one common frequency and provide an appropriate level of redundancy and throughput.

The introduction of local weather automation on a large scale has caused very great daily fluctuations of the demand for heat energy. The predominant function of the application is to control the distribution of energy streams produced by the plants (hot water with flow of up to 7000 m3/h each) to allow optimization of the resources utilization and provide an interface between the operators and the heating system. To control the heat stream distribution, selected nodes on the network are used to unmanned remote control of the chamber fittings in order to switch over the supply areas from one to another power plant and/or control pressure in selected points on the network. When controlling, all components of the system must be monitored in real time. Each of the software components provides diagnostics and statistics information via OPC tags, therefore it can be monitored by HMI and SCADA stations.

Automation islands—PLC's local controllers (network nodes)—equipped with radio transceivers are connected to the OPC server – CommServer [4]. This fully configurable OPC server implements architecture and algorithms (Sect. 3, 4, 5) so it provides a multi-protocol, multi-medium and multi-channel redundant access to the data of the plant-floor devices. To ensure short response time and effective utilization of the communication channels throughput, unique optimal scheduling algorithms have been implemented.

Such system cannot work alone. Thanks to the open and coherent infrastructure it has seamlessly been integrated with the following systems: power consumption prediction, geographical information, remote control of water main pumping stations, power plant monitoring systems. Using OPC as the core technology we have proven that the obtained structure is open, robust (because of built-in redundancy) and allows a straightforward integration process.

7 Conclusion

To satisfy the prerequisites defined in Sect. 2 a novel architecture is proposed. In this architecture, an additional element called *process observer* has been introduced between FM and PBM layer (Sect. 3, 4). The main function of that element is to represent the FM layer as a whole at the boundary of the PBM layer. Additionally, two unique closely coupled scheduling algorithms have been implemented in the proposed model (Sect. 5), namely *optimal transfer algorithm* and *adaptive sampling algorithm* to build cost saving, flexible solution providing process data just in time and preserving communication bandwidth. The

presented methodology and solutions were used as a platform to provide supervisory control of a metropolitan heating system supplied from three plants. Their optimal utilization requires controlling the load and pressure distribution. As the system is distributed geographically, safe communication between nodes is critical. Additionally, a great number of nodes requires an appropriate system structure.

The presented approach not only allows to meet the real-time distributed control requirements (Sect. 2), but also is a platform for multi-enterprise collaboration. It has been implemented using the OPC standard [3], [8] standard that permits a consistent method of accessing field data from plant-floor devices. Originally based on Microsoft's DCOM technology, currently new harmonized standard Unified Architecture (UA) has been just proposed. Because UA uses the concept of Web Services based on top of a proved industry chosen standard, such as XML technology it can be considered as vendor and platform independent. Using of this standard provides the prospect of seamless, truly open and easy enterprise-wide communications between devices and systems, from plant floor to MIS (Management Information System) and beyond. Feature research and implementation of the presented approach should be focused on migration towards UA standard.

References

1. Arendt D. Postol M.: *Real-time Multiprogramming System for Mine Control Center,* Microprocessors and Microsystems, 14 (1990) 39–46.
2. Bayne J. S.: *Automation and control in large-scale interactive systems,* Object-Oriented Real-Time Distributed Computing, (ISORC 2002). Proceedings. Fifth IEEE International Symposium on, (2002) 3–10.
3. Box D.: *Essential COM,* Addison Wesley, 1998.
4. *CAS: CommServer—Redundant, Multi-Protocol, Multi-Channel OPC Server For Large Scale Distributed Systems,* doc: PR214001BEN, (2005), `http://www.cas.eu`
5. *Remote Control of Lodz Agglomeration Heat Distribution System,* CAS (2004) `http://www.cas.eu`
6. Hayashi H.; Takabayashi Y.; Tsuji H.; Oka M.: *Rapidly increasing application of Intranet technologies for SCADA (supervisory control and data acquisition system),* Transmission and Distribution Conference and Exhibition 2002: Asia Pacific. IEEE/PES Volume 1, (2002) 6-10.
7. Nof S. Y., Morel G., Monostori L., Molina A., Filip F.: *From plant and logistics control to multi-enterprise collaboration,* Annual Reviews in Control 30 (2006) 55–68.
8. OPC Foundation: *OPC Overview,* Industry Standard Specification, (1998).
9. Postol M., Lipowska-Nadolska E.: *Using Monitor Concept to Maintain Concurrent Processes in Some Industrial Applications. Part II—Applications,* Using computers in elctrotechice (ZKwE 2002) Proceedings. VII Symposium on, Pozna, (2002) 429-432 (in polish).
10. Postol M.: *Remote Control Of Lodz Agglomeration Heat Distribution System,* CAS, Doc: O-03300101, (2003) `http://www.cas.eu`

11. Shono T.; Sekiguchi K.; Tanaka T.; Katayarna S.: *A remote supervisory system for a power system protection and control unit applying mobile agent technology,* Transmission and Distribution Conference and Exhibition: Asia Pacific. IEEE/PES Volume 1 (2002) 148–153.

Proceedings of the International Multiconference on
Computer Science and Information Technology pp. 871–881
ISSN 1896-7094
© 2007 PIPS

Reusing Verilog Designs in the Synchronous Language Esterel

Menachem Leuchter[1], Shmuel Tyszberowicz[2], and Yishai A. Feldman[3]

[1] The Open University, Israel
[2] Tel-Aviv University
[3] IBM Haifa Research Lab

Abstract. Veriest is an automatic translator that converts synthesizable Verilog designs into the synchronous language Esterel. The translation into a synchronous language can expose hidden flaws in the original design, including subtle race conditions. In addition, the extensive libraries of verified Verilog designs can now be reused in synchronous designs.
Verilog and Esterel have different models and features, complicating the translation. For example, Verilog has flexible data types and operators for dealing with data buses of varying widths; it also supports three-state logic, which has no equivalent in langauges not meant to describe hardware. Veriest creates functions in the hosting language (usually C) to represent concisely such features of Verilog that are not native to Esterel.

1 Introduction

Synchronous languages specify the reactions of reactive and real-time systems to their environment. They have rigorous mathematical semantics, which allows programmers to develop critical software faster and more reliably. Synchronous languages also enable validation and verification of the developed systems. They are particularly useful for designing the control of real-time embedded systems [8].

Modern hardware design is done using hardware description languages (HDLs); Verilog [6] is one of the two most popular such languages (the other being VHDL). As a result, there are many libraries of tested and verified Verilog designs available. These include many useful applications, such as communication protocols, video and audio compression algorithms, cryptographic algorithms, and more. These libraries far outnumber corresponding libraries for synchronous languages. Many of these asynchronous designs could also be useful for designers of reactive systems that use synchronous languages, if they could be translated into these languages. These could be used to synthesize synchronous designs in software as well as hardware.

Besides allowing reuse of existing designs, translating a Verilog design into a synchronous language can expose hidden flaws in the design that were not discovered by testing. Phenomena such as race conditions become obvious when stated in a synchronous language.

Such translation is complicated by the difference in models between Verilog and synchronous languages. Verilog is actually composed of two main sub-languages. *Synthesizable Verilog* is the subset of Verilog that can be directly compiled into hardware.

The rest of the language is used for designing stimulus environments (which are not a part of the design) for the simulation of synthesizable Verilog designs, and contains such features as the generation of random test vectors and assertions. Reusable Verilog designs are all written in synthesizable Verilog. This is a hardware-oriented language, and includes variable-length data types and flexible operators for combining and selecting parts of such buses. Verilog also supports three-state logic, in which a wire can be in an undriven (or "floating") state, enabling multiple sources of control for the wire. Such features of the language have no equivalent in the target synchronous languages.

We have implemented an automatic translator, called Veriest, that converts synthesizable Verilog designs into the synchronous language Esterel [4]. Esterel is one of the most popular synchronous programming languages, designed for programming reactive systems. Esterel can be compiled into sequential code as well as into hardware. Veriest preserves the Verilog RTL hardware semantics in the generated Esterel code.

In order to create readable Esterel programs, Veriest attempts to keep the structure of the original program as much as possible. Features of Verilog that are not directly expressible in Esterel are implemented using Esterel's support for external functions in the hosting language (usually C). Veriest defines a set of auxiliary functions, derived from the particular features and operations used in the source design. The function names follow a fixed scheme, so that their semantics can be immediately understood without reference to their implementation. In this way, implementation details are hidden while the design is still executable.

We have successfully translated a number of different Verilog designs using this tool. The generated Esterel code was about half again as long as the original code (excluding the generated C functions). The original intent was quite clear in the generated code, and it seemed to be understandable and maintainable.

1.1 The Gap

Esterel is a high-level synchronous language that supports abstraction, a variety of data types and the ability to define new ones, and interoperability with other languages. It contains special constructs for timing control, and is therefore appropriate as the target language for developing reactive systems.

In contrast, Verilog is meant for hardware design, and the synthesizable subset of Verilog lacks many of the abstraction mechanisms and data types of Esterel. It does emphasize timing, although in an asynchronous setting, and in that sense is closer to Esterel than most conventional programming languages. The translation of C/C++/Java code into Esterel is not as useful as that resulting from Verilog, since the former programs are sequential, and have no timing restrictions or concurrency. The loss in such languages of the synchronization characteristics in the design makes it less suitable for a synchronous language such as Esterel.

Some features of Verilog can be translated more-or-less directly into Esterel,[4] although care must be taken to preserve the semantics. Other features pose more difficulties. Verilog supports arbitrary-width variables, with their associated operations. For

[4] In most of this paper we refer to Esterel version 5, which was the latest version when this work was implemented [5]. We address the changes to Esterel in Section 3.1.

example, it is possible to define 5-bit variables, and arithmetical and shift operations on such variables truncate any results to 5 bits. Similarly, it is possible to define a variable consisting of 128 bits, which is beyond the capabilities of Esterel's built-in data types.

Verilog also supports flexible combinations of parts of such variables. For example, the expression
`{data_out[6:0], data_in}` represents the value containing the seven least-significant bits of `data_out` with the value of `data_in` concatenated on the right. In hardware, such operations correspond to simple wiring, without computational content. In Esterel, these have to be represented in code.

Hardware designs sometimes use *tri-state* (or *floating*) signals, which are distinct from logical high and low (or one and zero) values. The floating signals are not actively driven, so that any of a number of connected circuits can drive them in turn. This is useful in cases where one output value can result from one of several sources, each active at a different time, or when a single wire is used alternatively for input and for output. Verilog fully supports tri-state signals; for example, the expression `6'b101zzz` denotes a 6-bit value whose most-significant bits are 101, followed by three floating bits. These bits could be supplied by a different module. Esterel has no built-in support for tri-state values.

In addition, Esterel has no built-in support for arrays, a basic data type of Verilog (and many other languages). These have to be simulated in the generated code.

2 The Translation

Those constructs of Verilog that have direct counterparts in Esterel are translated separately and independently. Note that even seemingly-trivial operations such as addition are complicated by the presence of data types with flexible widths. For example, the Verilog expression `a+b` cannot be directly translated to Esterel if the original `a` and `b` are 5-bit variables, whereas their counterparts in Esterel are integers, since truncation of the sum to 5 bits needs to be added.

Unique Verilog constructs require a more detailed analysis of the Verilog source code. There are general translation solutions for each such construct, but these are sometimes overly complex. Veriest therefore attempts to recognize some frequently-used Verilog patterns, in order to translate those into more concise and efficient code.

Pure Esterel can express some Verilog features awkwardly, and others not at all. For example, it is impossible to represent in Esterel a 128-bit vector using the built-in types. Five-bit vectors *can* be represented using Esterel integers, but correctly translating operations on them to pure Esterel requires additional operations that increase code size and obscure its meaning. To achieve a full and efficient translation we use Esterel *user types* [1]. These types are considered completely abstract by Esterel itself; their implementation is given in the host language. Veriest defines a set of user types with associated operations. The types and operations generated depend on the particular source program; however, they follow common patterns, and their names completely describe their semantics. Designers who want to maintain and modify the generated Esterel code therefore do not need to read the C implementation at all.

```
module up_down_counter(data_out, carry,rst,clk,dir);
input clk,rst,dir;
output [23:0] data_out; output carry;
reg [23:0] data_out; reg carry;
always @(posedge clk or negedge rst)
begin
   if (!rst) begin data_out <= 0; carry <= 0; end
   else begin
      if (dir)
      begin
         data_out <= data_out + 1;
         if (data_out == 24'hffffff) carry <= 1;
         else carry <= 0;
      end
      else begin
         data_out <= data_out - 1;
         if (data_out == 0) carry <= 1;
         else carry <= 0;
      end
   end
end
endmodule
```

Fig. 1. A Verilog design of an up-down counter.

We start with a small yet complete example, then discuss the non-trivial issues involved in the translation one by one. The full details are available in [5].

Figure 1 contains a Verilog design of a simple 24-bit up-down counter. Its inputs are a clock, a counting direction selector line (`dir`) and a reset line (`rst`). Its outputs are a 24-bit bus that stores the current value and a carry for the next stage. The value of `data_out` is incremented or decremented at each rising clock cycle, depending on whether the value of `dir` is one or zero, respectively. Whenever `data_out` overflows or underflows, it is truncated, and `carry` is set to one for one clock cycle.

Figure 2 shows the Esterel code generated by Veriest. Both Verilog wires (transient values without memory, such as `clk` and `dir`) and registers (memory devices, such as `carry`) are translated into Esterel as signals with boolean values. A change in the variable value causes the corresponding signal to be emitted, while the value of the variable is stored in the signal's associated value. A binary value could have been represented by the presence or absence of a signal. However, the use of a signal with a value preserves two important characteristics of Verilog register variables: first, retaining the value after the signal disappears; and second, the ability to refer to the value of the signal at the previous clock tick even while a new value is assigned to it. Because of the need to refer to previous values, all signals are generated with appropriate initializations.

The Verilog design uses *procedural assignments*, which have a triggering condition. In this case, the triggering condition is `always @(posedge clk or negedge rst)`, which means that this section of code is activated on a positive edge of the `clk` signal or a negative edge of `rst`. This is translated into an Esterel loop that waits

```
module up_down_counter:
type VEC_23_0;    % definition of new type for 23-bit register
function CONV_STRING_VEC_23_0(string):VEC_23_0;
function PLUS_VEC_23_0(VEC_23_0, VEC_23_0, integer):VEC_23_0;
function COMP_EQ_VEC_23_0(VEC_23_0, VEC_23_0): boolean;
function MINUS_VEC_23_0(VEC_23_0, VEC_23_0, integer):VEC_23_0;
input clk:=false: boolean, rst:=false: boolean, dir:=false: boolean;
output data_out:VEC_23_0, carry:=false: boolean;
loop    % procedural assignment section
 await [clk or rst];
 if (((?clk) and not pre(?clk)) or (not(?rst) and pre(?rst))) then
  if (not (?rst)) then
   emit data_out(CONV_STRING_VEC_23_0("0"));
   emit carry(false);
  else if ((?dir)) then
   emit data_out(PLUS_VEC_23_0(pre(?data_out),
                               CONV_STRING_VEC_23_0("1")), 24);
    if (COMP_EQ_VEC_23_0(pre(?data_out),
 CONV_STRING_VEC_23_0("16777215")))
     then emit carry(true);
     else emit carry(false);
    end if;
   else
    emit data_out(MINUS_VEC_23_0(pre(?data_out),
                                 CONV_STRING_VEC_23_0("1")), 24);
    if (COMP_EQ_VEC_23_0(pre(?data_out), CONV_STRING_VEC_23_0("0")))
     then emit carry(true);
     else emit carry(false);
    end if;
   end if;
  end if;
 end if
end loop
end module
```

Fig. 2. The Esterel version of the up-down counter.

for a change in `clk` or `rst`. Because these are signals with values, the meaning of the statement `await [clk or rst]` is to wait for the *existence* of either signal, regardless of their values. A signal exists when a value is emitted for it. The type of change (from false to true, denoting a positive edge, or from true to false, denoting a negative edge) is checked separately, with the following `if` statement. This is necessary because Esterel does not support the notion of positive or negative edges of signals, and is the reason for the need to reference previous values. This treatment of signals allows the Esterel compiler to generate more efficient code, which only checks the conditions when the value is assigned, rather than at each instant.

The body of the Verilog `always` statement uses a number of *nonblocking assignments*, expressed using the `<=` operator. The right-hand sides of all concurrent

(a) Verilog source

```
assign o1 = a | b;
assign b = a;
```

(b) Esterel translation

```
loop
   await [a or b]; emit o1(?a or ?b);
end loop
||
loop
   await a; emit b(?a);
end loop;
```

Fig. 3. Translating continuous assignments.

nonblocking assignments are computed before any value is changed. In the generated Esterel code, the same effect is achieved by using the previous values of the variables on the right-hand side. Thus, the two Verilog nonblocking assignments `a <= b; b <= c;` would be translated into `emit a(pre(?b)); emit b(pre(?c));` whereas the (sequential) blocking assignments `a = b; b = c;` would be translated into `emit a(?b); emit b(?c);`.

Other than these changes, the structures of the two programs are similar. The most striking change is the treatment of the 24-bit data type used for `data_out`. In Verilog, the definition `reg [23:0] data_out` causes the addition operator in `data_out + 1` to denote 24-bit addition, which truncates its result to the required length. To achieve the same result in Esterel, Veriest defines a user type called `VEC_23_0`. All the required operations on this data type are defined as user functions. This particular program uses addition and subtraction on 24-bit quantities, and compares them for equality. Veriest therefore defines the functions `PLUS_VEC_23_0`, `MINUS_VEC_23_0`, and `COMP_EQ_VEC_23_0` for these operations. The implementation of these functions in C is straightforward, and is generated automatically.

This example also shows the treatment of constants. In this example, the target host language could represent integers of up to 16 bits. The 24-bit vectors of the example are therefore represented as a struct of two integers, and there is no general way to specify constants of this type. The general solution in such cases is to use strings to encapsulate the constants, with conversion functions (like `CONV_STRING_VEC_23_0`) to create the internal representation. Of course, whenever possible, numeric constants are used.

Continuous Assignment In addition to the procedural assignments shown in the example, Verilog also supports *continuous assignment* statements of the form `assign` *var = expression*. Their semantics is that the variable always has the value of the expression; whenever the value of the expression changes, so does the value of the variable.

In order to translate continuous assignments into Esterel, they must be placed in a loop that waits on any variable involved in the expression to change. If there are multiple continuous assignments, they are translated separately into parallel Esterel loops. Figure 3 shows an example of such translation.

Variable-Width Vectors There are two possible ways of dealing with variable-width vectors. The first is to embed them into the closest built-in integer type large enough to hold them. This makes arithmetical operations relatively straightforward, but selecting

subfields and combining them to create new vectors is more difficult. The second approach is to keep each vector as a set of separate bits. This makes selection, shifting, and concatenation easier, but arithmetic becomes complex. Quite often, the same variable participates in both types of operations. Veriest takes the first approach. Each of the new types created for variable-width vectors is embedded in the next highest integer type, or in a C struct of integers in case the field is too wide. As shown above, special-purpose functions are created as necessary in order to perform arithmetical operations on the new types. Similar functions are created for selection and concatenation.

A set of concatenation functions is created when variable-width vectors are combined. For example,

```
data[7:0] = {data[6:0], a}
```

could be translated into

```
emit data(CONCAT_INT_BOOL(CONV_INT_INT(?data, 6, 0), 7, ?a,1)),
```

where the second and fourth arguments specify the number of bits to take from the preceding argument. However, Veriest recognizes this pattern as a shift, and produces the more concise

```
emit data(OR_INT_BOOL(SHIFT_L_INT_INT(?data, 7,1), ?a)).
```

The second argument to the shift function is the width of the shifted field, and the third is the shift amount.

Such heuristics are not necessary for correct translation, but they improve code readability and reduce its size. Since the goal of the translation is to generate maintainable code in the synchronous language, it is important to include such heuristics for common cases, such as shifts. For the same reason, built-in Esterel operators are used when possible. For example, Veriest will use "`=`" instead of the function `COMP_EQ_VEC_8_0`.

Arrays Verilog supports arrays of other types, and in particular arrays of registers. For example, the declaration `reg [7:0] mem[0:255]` defines an array containing 256 8-bit vectors. (This is a typical way to define memory.) Esterel, on the other hand, does not have built-in arrays. As in the case of vectors, the solution is to use user types. In this example, Veriest will create the type `ARRAY_REG_7_0_255_0`, with associated operations for reading and writing elements or ranges of arrays.

Tri-State Logic Verilog supports a floating state for bits in addition to zero and one. In Esterel, it must be simulated using an additional bit. For each variable that may have floating bits, Veriest creates another variable that has the same name with a `_Z` suffix. Each bit in the new variable has the value one exactly when the corresponding bit in the original variable is in the floating state. Veriest creates code to manage both variables together in order to represent the Verilog semantics. For example, the Verilog excerpt

```
inout [2:0] data;
data = (select) ? 3'b111 : 3'bzzz;
```

is translated into

```
inputoutput data: integer;
output data_Z: integer;
if (?select) then
   emit data(7); % set data to 3'b111
   emit data_Z(0); % set data to normal mode
else
   emit data_Z(7); % set data to tri-state mode
end if
```

In this case, the variable `data` is shared between all modules that may attempt to write to it, but `data_Z` is duplicated for each module. A module may only write to `data` if it asserts in `data_Z` that those bits it is writing to are not floating. Note that when `select` is false and all bits of `data` are set to tri-state mode in the Verilog source, there is no need to change the value of the `data` variable in Esterel, since its value is irrelevant when `data_Z` is set to all ones.

Veriest creates special comparison functions, such as `COMP_EQ_Z` and `COMP_NEQ_Z`, for comparing tri-state variables.

3 Results and Discussion

We have evaluated Veriest on four designs. These are all real and useful Verilog designs, each written by a different programmer. The example designs include the following: the up-down counter shown in Section 2; a Viterbi encoder, an error-correcting algorithm based on convolution encoding [7]; a real-time clock; and an implementation of the I^2C protocol for connecting multiple devices using only two wires. In all cases, the generated code was functionally equivalent[5] to the original. This section analyzes the results of the translation of these examples, and discusses their implications.

3.1 Maintainability

Esterel code generated by Veriest tends to be longer than the original Verilog code; the average increase in our examples was 60% (excluding the generated C functions). Generated lines also tend to be longer, due to the substitution of primitive operators such as arithmetic, selection, shifting, and concatenation, by generated functions with relatively long names. Nevertheless, most lines have a reasonable length.

There are several other factors for the increase in the total size of the code. The largest overhead in the translation comes from continuous assignments, each of which needs to be translated into a loop. This is mostly due to the need to separate the triggering condition into two parts, one in the `await` statement listing the signals to watch, and the other in the `if` statement that checks the particular triggering values. Another source for the increase in size of the generated code is the difference between the control structures in the two languages. For example, Esterel lacks a `case` or `switch` statement; Verilog's `case` statement is therefore translated into nested `if`s. A very common

[5] The transformations were designed to preseve the Verilog RTL hardware semantics. We also ran simulations of the source and target programs to check their equivalence.

construct in hardware designs is a state machine, which is often implemented as a long `case` statement. Translating that statement into an `if` results in many terminating `end if` statements, which increase code size and decrease readability.

In spite of all these factors, the increase in the total size of the code seems quite reasonable. The structure of the original design is preserved in the translated design to the best degree possible, including all module, register, and wire names. The original intent is therefore easily discernible in the generated design, making it easily maintainable.

Since the publication of the first author's thesis [5] (on which this paper is based), the Esterel language (currently at version 7) has been enhanced with new data types and operators. These include bitvectors (arrays of booleans), bitvector maps (records composed of fields and bitvector slices), encoders from unsigned expressions to bitvectors, and signal arrays. These were added to the language to support "hardware or low-level software designs" [3]. These changes remove the need to use auxiliary C functions to implement operations previously unavailable. The translation can now be even more concise and readable.

3.2 Discovering Race Conditions

Because of the asynchronous nature of the language, it is possible to write in Verilog designs that have race conditions. These are not flagged by the compiler, and may not be caught during simulation. Figure 4(a) shows a simple example of a race condition [2]. In this example, a `rst` signal sets `x` to 0 and `y` to 1. A subsequent `clk` signal causes a race condition, where the first `always` block tries to set `x` to the value of `y` while the second block simultaneously tries to set `y` to the value of `x`. The Verilog simulator will arbitrarily execute one block first. The result will be that both `x` and `y` will have the same value: 1 if the first block is executed first, and 0 if the second block executes first.

The Veriest translation of this example to Esterel is shown in Figure 4(b). Because of the synchronous semantics of Esterel, this program is illegal and the compiler rejects it with the error message "statically cyclic program, cannot be compiled by scssc."

In this simple Verilog example, the race condition is easy to spot by inspection. However, subtle race conditions in larger Verilog designs will be harder to discover. Simulation, and even hardware execution (for different reasons), may yield correct results even in the presence of race conditions, and the problem might manifest itself only under rare conditions that are difficult to duplicate. The translation to the synchronous language discovers these problems during compilation.

4 Conclusions

We have presented Veriest, a translator that can convert designs in synthesizable Verilog into the synchronous language Esterel. This makes the large body of intellectual-property hardware designs available to be incorporated into synchronous designs. As an added benefit, the translation of an asynchronous design into a synchronous language can help discover subtle timing conditions that may have escaped testing in an asynchronous setting.

```
module race(x, y, clk, rst);
   output x, y; input clk, rst; reg x, y;
   always @(posedge clk or posedge rst)
      if (rst) x = 0; else x = y;
   always @(posedge clk or posedge rst)
      if (rst) y = 1; else y = x;
endmodule
```

(a) A Verilog design with a race condition [2].

```
module race:
input clk:=false:boolean, rst:=false:boolean;
output x:=false:boolean, y:=false:boolean;
loop
   await [clk or rst];
   if (((?clk) and not pre(?clk)) or ((?rst) and not pre(?rst))) then
      if (?rst) then emit x(false); else emit x(?y); end if;
   end if
end loop
||
loop
   await [clk or rst];
   if (((?clk) and not pre(?clk)) or ((?rst) and not pre(?rst))) then
      if (?rst) then emit y(true); else emit y(?x); end if;
   end if
end loop
end module
```

(b) The translation to Esterel.

Fig. 4. A race condition.

The translation is complicated by several hardware-related features of the source language. These are solved by using user types implemented in the host language. The structure of the resulting translation is very close to that of the original, although it is somewhat more verbose. Experiments have shown that the results are still readable and maintainable in the target language.

Some Verilog designs are more appropriate than others for automatic translation to a synchronous language. Designs of large generic processing components such as CPUs and DSPs will probably not generate useful synchronous designs. The same effects can be better achieved by coding directly in Esterel. On the other hand, hardware-oriented protocols for specific purposes, such as communication protocols and compression algorithms, are good candidates for automatic conversion and are expected to yield results comparable to our example cases.

References

1. G. Berry and G. Gonthier. The Esterel synchronous programming language: Design, semantics, implementation. *Science of Computer Programming*, 19(2):87–152, 1992.
2. C. E. Cummings. Nonblocking assignments in verilog synthesis, coding styles that kill! In *Synopsis Users Group (SNUG)*, 2000.
3. Esterel Technologies. The Esterel v7 reference manual. `http://www.esterel-technologies.com/files/Esterel-Language-v7-Ref-Man.pdf`.
4. N. Halbwachs. *Synchronous Programming of Reactive Systems*. Kluwer, 1993.

5. M. Leuchter. Translating Verilog designs into the synchronous language Esterel. Master's thesis, The Open University, Israel, February 2003.
6. S. Palnitkar. *A Guide to Digital Design and Synthesis*. Prentice Hall, 1996.
7. J. G. Proakis. *Digital Communications*. McGraw-Hill, 3rd edition, 1995.
8. R. Shyamasundar and J. Aghav. Realizing real-time systems from synchronous language specifications. Real Time Systems Symposium, Work in Progress Session, Orlando, Florida, USA, 2000.

Proceedings of the International Multiconference on
Computer Science and Information Technology pp. 883–892
ISSN 1896-7094
© 2007 PIPS

Towards the Safety Verification of Real-Time Systems with the Coq Proof Assistant

Olga Tveretina

Institute for Computing and Information Sciences,
Radboud University Nijmegen,
The Netherlands
olga@cs.ru.nl

Abstract. Hybrid systems are systems involving the interaction of discrete and continuous dynamics. Hybrid systems have been used as a mathematical model for many safety critical applications. One of the most important analysis problems of hybrid systems is the reachability problem. Approaches based on predicate abstraction are widely used for the reachability analysis. They are not efficient enough because of introducing additional transitivity along the series of abstract states. In this paper we give an approach to solve this problem for some classes of hybrid systems. A verification example formalized within the Coq proof assistant is provided.

Keywords: Hybrid systems, reachability, verification, discrete abstraction

1 Introduction

Real-time embedded systems have become very important in our everyday life. Programs such as device drivers and embedded controllers must run on real-time constraints. Demand placed on the embedded systems functionality, complexity and critical nature are increasing [5].

Hybrid systems are systems when there is a significant interaction between the continuous and discrete parts and high performance specifications are to be met by the system. For example, hybrid systems arise from the interaction of discrete planning algorithms and continuous processes. The study of hybrid systems is essential in designing intelligent control systems. Examples of such systems include intelligent highway systems, air traffic management systems, computer and communication networks, smart houses and others. Another challenging application is a construction of mathematical models of biological processes.

Many of the above mentioned applications are *safety critical* and require the guarantee of safe operation. The problem of safety verification seeks for an answer to the question: is there a potentially unsafe state reachable from an initial state? Therefore, formal verifying safety properties of a hybrid system consists of building a set of reachable states and checking whether this set intersects with a set of unsafe sets. That is why, one of the most central problems in

the analysis of hybrid automata is the *reachability* problem. For systems with continuous dynamics, it is very difficult to compute the set of all states reachable from an initial set.

Abstraction is one of the complexity reduction techniques [4]. It reduces the state space of a system by mapping it to an abstract set of states that preserve the actual behavior of the system. Predicate abstraction is the most widely used form of the abstraction. Given a concrete infinite state system and a set of abstraction predicates, a finite state abstraction is produced. It is an abstraction in terms of that for every execution in the concrete system there is a corresponding execution in the abstract system. A predicate abstraction approach for the verification of hybrid systems is represented in [9]. However, the computational cost of predicate abstraction can be too high, and for large systems practically infeasible. That is why we start from a method that decomposes the state space of a system according a rectangular grid [11]. We show applicability of our method with the verification example that we have formalized within the `Coq` proof assistant. It models the gate controller of a railroad crossing.

Related work. The problem of computing reachable states of hybrid systems received a lot of attention during past several years and different reachability tools have been developed. There are two main approaches for reachability analysis: a) the methods "overapproximating" the reachable set; b) the methods based on computing "convergent approximations" to reachable sets [12]. "Overapproximating" methods compute an overapproximation of a set of reachable states. The tools such as d/dt [2] and *Checkmate* [10] represent sets as convex polyhedra and propagate these polyhedra under continuous dynamics. A second group of methods is based on computing solutions to static Hamilton-Jacobi equations and on techniques from viability theory and set valued analysis [12]. Overapproximating approaches are mostly hard for systems with non-linear dynamics. The methods in the second group are exponential in n and thus not practical for problems with large dimensions.

2 Problem description

Basically, a hybrid system is a graph, whose vertices represent continuous changes and whose edges represent discrete transitions. A system trajectory is a sequence of continuous flows and discrete jumps. A hybrid automaton combines discrete-state and continuous-state dynamics to model systems which evolve both continuously and according discrete jumps. Therefore, we use a hybrid automaton as a mathematical formalism for describing hybrid systems.

Now we formally define the notion of hybrid automaton. In the following $P(Q)$ denotes the power set (the set of all subsets) of Q, and $\mathbb{R}$ denotes the set of real numbers.

Definition 1. *A Hybrid automaton is a tuple* $\mathcal{H} = (\mathcal{L}, n, \mathcal{S}_0, \mathcal{I}, F, \mathcal{G}, \mathcal{R})$ *with the following components:*

- $\mathcal{L}$ *is a finite set of discrete locations;* $n \geq 0$ *is called the dimension of* $\mathcal{H}$. *The state space of* $\mathcal{H}$ *is* $\mathcal{S} = \mathcal{L} \times \mathbb{R}^n$. *Each state thus has the form* (l, x), *where* $l \in \mathcal{L}$ *is a discrete part, and* $x \in \mathbb{R}^n$ *is continuous;*
- $\mathcal{S}_0 \subseteq \mathcal{S}$ *is a set of initial states;*
- $\mathcal{I} : \mathcal{L} \to P(\mathbb{R}^n)$ *assigns to each location* l *an invariant set* $\mathcal{I}(l) \subseteq \mathbb{R}^n$, *which constrains the value of the continuous part while the discrete part is* l, *i.e. continuous evaluation can go on as long as* x *remains in* $\mathcal{I}(l)$.
- $F : \mathcal{S} \to \mathbb{R}^n$ *assigns to each discrete state a continuous vector field, i.e. starting from an initial state* $(l_0, x_0) \in \mathcal{S}_0$ *the continuous state* x *flows according to the differential equation* $\dot{x} = F(l_o, x)$ *and* $x(0) = x_0$. *We denote by* $\dot{x}$ *the first derivative of* x *with respect to time, i.e.* $\dot{x} = dx/dt$.
- $\mathcal{G} : \mathcal{L} \times \mathcal{L} \to P(\mathbb{R}^n)$ *describes a guard condition, i.e. if a system remains in a discrete location* l_1 *and a continuous state* x *reaches the guard* $\mathcal{G}(l_1, l_2)$ *than the discrete state may change its value to* l_2.
- $\mathcal{R} : \mathcal{L} \times \mathcal{L} \times \mathbb{R}^n \to P(\mathbb{R}^n)$ *is a reset function. It means that when a discrete state changes its value from* l_1 *to* l_2, *a continuous state gets reset to some value in* $\mathcal{R}(l_1, l_2, x) \subseteq \mathbb{R}^n$.

For simplicity, we assume that a number of discrete locations is finite and for all $l \in \mathcal{L}$, the vector field $\mathrm{F}(l, \cdot)$ is Lipschitz continuous. It guarantees that the solutions of the differential equation $\dot{x} = \mathrm{F}(l, x)$ are well-defined.

Example 1. (*Temperature control*) The temperature T of a room is controlled by a thermostat, which continuously checks the temperature and turns the heater on and off if some threshold values T_{min} and T_{max} are reached. When the heater is on, the temperature increases according to the flow $\dot{T} = kt$, when the heater is off, the temperature decreases according to the flow $\dot{T} = -kt$. The resulting hybrid system is shown in Figure 1. A variable c is used to ensure an upper bound spent in each node.

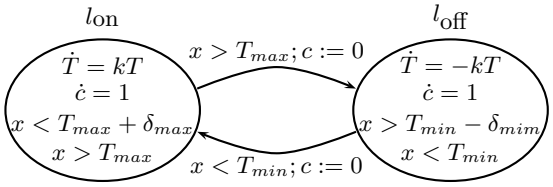


Fig. 1. Graphical representation of the thermostat hybrid automaton.

Let $\mathbb{R}_{\geq 0}$ be the set of non-negative real numbers. We denote by Φ the set of functions $f : \mathbb{R}_{\geq 0} \to \mathbb{R}^n$ and define the flow of a system in a discrete location l as $\phi_l(x, f, t)$ with an initial condition $\phi_l(x_0, f, 0) = x_0$. A system state can change in two ways: either by time evaluation or by discrete transitions. Hence, there are two kinds of transitions: a continuous transition, denoted as $\rightarrow_c$, describes

evolution of a system in a given location, and a transition, denoted as $\rightarrow_d$, describes moving from one location to another location and, possibly, changing the continuous variables according to the function $\mathcal{R}$. Then the semantics of a hybrid automaton is given by the corresponding transition system [1].

Definition 2. *Given a hybrid automaton $\mathcal{H} = (\mathcal{L}, n, \mathcal{S}_0, \mathcal{I}, F, \mathcal{G}, \mathcal{R})$, the transition system $\mathcal{T} = \{\mathcal{S}, \rightarrow, \mathcal{S}_0\}$ of $\mathcal{H}$ consists of*

- *set of states $\mathcal{S}$;*
- *the set of initial states $\mathcal{S}_0$;*
- *the transition relation $\rightarrow \subseteq \mathcal{S} \times \mathcal{S}$ between states, and it is defined as an union of two relations $\rightarrow_c$ and $\rightarrow_d$, where $\rightarrow_c \subseteq \mathcal{S} \times \mathcal{S}$ the relation describing a transition due to continuous flow and $\rightarrow_d \subseteq \mathcal{S} \times \mathcal{S}$ the relation describing a transition due to a discrete jump, i.e.*

  $(l, x) \rightarrow_c (l, y) \Leftrightarrow \exists t \geq 0, f \in \Phi : \phi_l(x, f, t) = y \wedge \forall t' \in [0, t],\ \phi_l(x, f, t') \in \mathcal{I}(l)$
  $(l, x) \rightarrow_d (l', y) \Leftrightarrow x \in \mathcal{G}(l, l') \wedge y = \mathcal{R}(l, l', x)$

  When it is necessary, we will use $(l, x) \rightarrow_c{}^t (l, y)$ instead of $(l, x) \rightsquigarrow_c (l, y)$.

Trajectories of a hybrid automaton are sequences of continuous flow and discrete steps and start from initial states.

Definition 3. (*Trajectory*) *Given a hybrid automaton $\mathcal{H} = (\mathcal{L}, n, \mathcal{S}_0, \mathcal{I}, F, \mathcal{G}, \mathcal{R})$, we say that $\tau_n : s_0 \rightarrow \cdots \rightarrow s_n$ is a trajectory of $\mathcal{H}$ if $s_0 \in \mathcal{S}_0$, and $s_i \in S$,$1 \leq i \leq n$, where n is the length of the trajectory.*

Given a hybrid automaton $\mathcal{H}$ and a set of unsafe states $\mathcal{U}$, the safety verification problem concerns with proving that all trajectories of $\mathcal{H}$ cannot enter $\mathcal{U}$. Formally it can be defined as follows.

Definition 4. (*Safety*) *We say that a hybrid automaton is safe if for all $n \geq 0$ there is no trajectory $s_0 \rightarrow \cdots \rightarrow s_n$ such that $s_i \cap \mathcal{U} \neq \emptyset$ for some $0 \leq i \leq n$.*

Our goal is to design an algorithm that, given a hybrid automaton and a set of unsafe states, decides whether the corresponding system is safe. However, this problem in is undecidable in general [6]. That is why we are interested in algorithms that terminate efficiently for problems of practical importance.

3 Reachability analysis

In this section we describe an algorithm for verifying safety of hybrid systems. Basically, it is a modification of traditional methods based on discrete abstraction. The fact that the approaches based on a discrete abstraction of a state space produce too many spurious transitions, motivated us to find another approach.

3.1 Abstract transition system

Our method is based on the approach that decomposes the continuous state space according to a n-dimensional rectangular grid as, for example, in Figure 2. Such abstractions are mostly performed in a manual manner. In general, a state space can be represented by polyhedra [1]. This is a more flexible approach but it requires an algorithm for dealing with these polyhedra. A rectangular grid is less flexible but it is simpler to implement the corresponding operations.

We assume that we have an algorithm that can produce a decomposition of a state space, as for example in [8]. We denote by χ an abstract state and by S^a a set of all abstract states.

In order to define a discrete abstraction of a given system, we have to describe the transitions between the abstract states, the set of initial abstract states S_0^a, and the set of abstract unsafe states $\mathcal{U}^a$. Given a hybrid automaton $\mathcal{H}$ and a set of abstract states, the set of initial abstract states can be computed.

Definition 5. *A hybrid automaton* $\mathcal{H} = (\mathcal{L}, n, \mathcal{S}_0, \mathcal{I}, F, \mathcal{G}, \mathcal{R})$ *and an abstract state space* S^a *generate the abstract transition system* $\mathcal{T}^a = \{\mathcal{S}^a, \rightsquigarrow, S_0^a\}$*, where*

- *the set of initial abstract states* $\mathcal{S}_0^a$*: an abstract state* $(l, \chi) \in \mathcal{S}_0^a$ *if there is* $(l, x) \in \mathcal{S}_0$ *such that* $x \in \chi$*;*
- *the abstract transition relation* $\rightsquigarrow \subseteq \mathcal{S}^a \times \mathcal{S}^a$ *is a union of two relations:* $\rightsquigarrow_c \subseteq \mathcal{S}^a \times \mathcal{S}^a$ *is a relation describing a transition due to continuous flow and* $\rightsquigarrow_d \subseteq \mathcal{S}^a \times \mathcal{S}^a$ *is a relation describing a transition due to a discrete jump, i.e.*

  $(l, \chi) \rightsquigarrow_c (l, \chi') \Leftrightarrow \exists t \geq 0, x \in \chi, x' \in \chi', f \in \Phi, l' \in \mathcal{L} : \phi_l(x, f, t) = x' \wedge x' \in \mathcal{G}(l, l') \wedge \forall t' \in [0, t],\ \phi_l(x, f, t') \in \mathcal{I}(l)$

  $(l, \chi) \rightsquigarrow_d (l', \chi') \Leftrightarrow \exists x \in \chi, x' \in \chi', : x \in \mathcal{G}(l, l') \wedge x' = \mathcal{R}(l, l', x).$

  When it is necessary, we will use $(l, \chi) \rightsquigarrow_c{}^t (l, \chi')$ *instead of* $(l, \chi) \rightsquigarrow_c (l, \chi')$*.*

Due to finiteness of the abstraction, we can check it's safety by computing all possible transitions.

The abstract transition system described here differs from the abstraction transition system as given by [1] in terms of the formulation of the relation $\rightsquigarrow_c$ defining a continuous step.

Example 2. The grid for the thermostat example is depicted in Figure 2. Each rectangular box represents an abstract state. We assume that $T^*_{min} < T_{min} < T_{max} < T^*_{max}$, where $T^*_{min} = T_{min} - \delta_{min}$ and $T^*_{max} = T_{max} + \delta_{max}$. While staying in a discrete location l_{on}, an abstract continuous step from a state (l_{on}, χ_0) to a state (l_{on}, χ_1) is possible.

Definition 6. *An abstract trajectory is defined as a sequence*

$$s^a_{(0,0)} \rightsquigarrow_c s^a_{(0,1)} \rightsquigarrow_d s^a_{(1,0)} \rightsquigarrow_c \ \cdots \ \rightsquigarrow_d s^a_{(i,0)} \rightsquigarrow_c s^a_{(i,1)} \rightsquigarrow_d \ \cdots,$$

where $s^a_{(0,0)} \in S^a_0$*.*

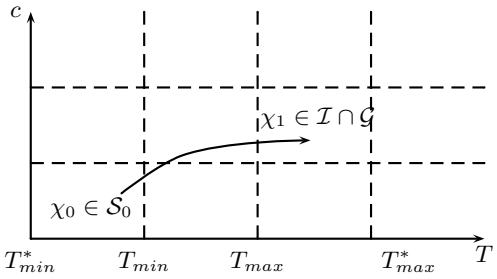


Fig. 2. An abstract continuous step from a state (l_{on}, χ_0) to a state (l_{on}, χ_1).

An abstract trajectory is closed if there are i and j such that $i < j$ and

$$s^a_{(0,0)} \rightsquigarrow_c s^a_{(0,1)} \rightsquigarrow_d s^a_{(1,0)} \rightsquigarrow_c \ \cdots \ \rightsquigarrow_d s^a_{(i,0)} \rightsquigarrow_c \ \cdots \ \rightsquigarrow_d s^a_{(j,0)},$$

where $s^a_{(i,0)} = s^a_{(j,0)}$. We denote by $\mathrm{Tr}(s^a)$ a closed abstract trajectory starting from an abstract state s^a.

Note that in general an abstract trajectory starting from some initial abstract state is not unique. Existence of a closed abstract trajectory follows from finiteness of the continuous state space partition.

Standard approaches based on predicate abstraction are less efficient than they could be, because of introducing additional transitivity along the series of abstract states, i.e. they produce too many spurious abstract transitions. It means that if for abstract states χ_1, χ_2 and χ_3 holds $\chi_1 \rightsquigarrow_c \chi_2$ and $\chi_2 \rightsquigarrow_c \chi_3$ than χ_3 is proclaimed to be reachable from χ_1, although it is not always the case. In contrast to the standard methods, our framework allows computing only transitive transitions.

3.2 Algorithm

Given a hybrid system, the reachability problem is to check whether all trajectories starting from an initial state avoid unsafe regions. If the reachability is verified for the abstraction then it holds in the concrete system. In general, this problem is undecidable. So we have aimed to have an algorithm that given a hybrid system and an abstraction of the state space the following holds: a) it terminates; b) if it terminates with an answer 'safe', then the original system is safe.

We define an "auxiliary" abstract continuous step as follows.

$(l, \chi) \rightsquigarrow^{aux} (l, \chi') \Leftrightarrow \exists x \in \chi, x' \in \chi' : (l, x) \rightarrow_c (l, x')$

The above definition is similar to the definition of an abstract continuous step of [1] but it serves a different purpose, i.e. defining the "auxiliary" set of successors ${Succ_c}^{aux}$ we avoid computing spurious abstract transitions.

${Succ_c}^{aux}((l, \chi), (l, \chi')) = \{(l, \chi'') \in \mathcal{S}^a \mid \exists t \geq 0 : (l, \chi) \rightsquigarrow_c{}^t (l, \chi') \wedge \forall t' \in]0, t[, (l, \chi) \rightsquigarrow^{aux} (l, \chi'')\}$.

Given an abstract trajectory

$$\tau^a : s^a_{(0,0)} \rightsquigarrow_c s^a_{(0,1)} \rightsquigarrow_d s^a_{(1,0)} \rightsquigarrow_c s^a_{(1,1)} \rightsquigarrow_d \ \dots \ \rightsquigarrow_d s^a_{(n,0)}$$

the set Reach(τ) is define as follows.

$$\text{Reach}(\tau) = \bigcup_{i=0}^{n} s^a_{(i,0)} \cup \bigcup_{i=0}^{n-1} {Succ_c}^{aux}(s^a_{(i,0)}, s^a_{(i,1)})$$

Lemma 1. *Given a hybrid automaton $\mathcal{H} = (\mathcal{L}, n, \mathcal{S}_0, \mathcal{I}, F, \mathcal{G}, \mathcal{R})$, for each trajectory τ there is a corresponding abstract trajectory τ^a such that for each $s \in \tau$ there is $s^a \in Reach(\tau^a)$ such that $s \in s^a$.*

Proof. We give a sketch of a proof. Suppose

$\tau : s_{(0,0)} \rightarrow_c \dots \rightarrow_c s_{(0,i_0)} \rightarrow_d s_{(1,0)} \rightarrow_c \dots \rightarrow_d s_{(n,0)} \rightarrow_c \dots \rightarrow_c s_{(n,i_n)}$

Let us construct the following abstract trajectory

$\tau^a : s^a_{(0,0)} \rightsquigarrow_c s^a_{(0,1)} \rightsquigarrow_d s^a_{(1,0)} \rightsquigarrow_c s^a_{(1,1)} \rightsquigarrow_d \dots \rightarrow_d s^a_{(n,0)}$,

where $s_{(k,0)} \in s^a_{(k,0)}$, $0 \le k \le n$ and $s_{(k,i_k)} \in s^a_{(k,i_k)}$, $0 \le k \le n-1$.

By definition, $s_{(k,j)} \in {Succ_c}^{aux}(s^a_{(k,0)}, s^a_{(k,1)})$, $0 \le k \le n-1$, $1 \le j \le i_k$.

We conclude that for each $s \in \tau$ there is $s^a \in \text{Reach}(\tau^a)$ such that $s \in s^a$.

Given an abstraction of a state space and a set of unsafe states, the set of unsafe abstract states $\mathcal{U}^a$ can be computed.

The methods based on predicate abstraction are often less efficient than they could be because of the accuracy to which the abstract reachable states are computed. The extra accuracy requires much more calls of validity checker than it is necessary. Often an *over-approximation* of abstract reachable states is sufficient to prove the verification condition. In cases when the continuous flow is represented by a monotone function in a discrete location l, we can compute $Succ^{aux}((l,\chi),(l,\chi'))$ as a convex hull of χ and χ'.

We define the set $Succ(Q)$ of successors of a set $Q \in S^a$ and an 'auxiliary' set of reachable states $\text{Reach}^{aux}(Q)$ as follows.

$$Succ(Q) = \{s \in \mathcal{S}^a \mid \exists q \in Q, s' \in S^a : q \rightsquigarrow_c s' \wedge s' \rightsquigarrow_d s\}$$

$$\text{Reach}^{aux}(Q) = \{s \in S^a \mid \exists q \in Q, p \in S^a : (q \rightsquigarrow_c p) \wedge (s \in Succ^{aux}(q,p) \cup p)\}.$$

Theorem 1. *Given a hybrid automaton $\mathcal{H} = (\mathcal{L}, n, \mathcal{S}_0, \mathcal{I}, F, \mathcal{G}, \mathcal{R})$, a finite set of abstract states S^a, and a set of abstract unsafe states $\mathcal{U}^a$, the safety of the abstract system implies the safety of $\mathcal{H}$.*

Now we define hybrid systems for which this approach can be applied efficiently. We need to define some restrictions on functions describing continuous behavior.

Algorithm 1 REACHABILITY ALGORITHM

1: $i := 0$;
2: $\text{Reach}_i := \mathcal{S}_0$;
3: $Succ_i := \mathcal{S}_0$;
4: $Succ_{i+1} := Succ(\mathcal{S}_0)$;
5: $\text{Reach}_{i+1} := \mathcal{S}_0 \cup Succ_i \cup \text{Reach}^{aux}(\mathcal{S}_0)$;
6: **while** $(\text{Reach}_i \cap \mathcal{U}^a = \emptyset \wedge \text{Reach}_{i+1} \not\subseteq \text{Reach}_i)$ **do**
7: if $\text{Reach}_{i+1} \cap \mathcal{U}^a \neq \emptyset$ return "unsafe";
8: if $\text{Reach}_{i+1} \subseteq \text{Reach}_i$ return "safe";
9: $i := i + 1$;
10: $Succ_i := Succ(Succ_{i-1})$;
11: $\text{Reach}_{i+1} := \text{Reach}_i \cup Succ_i \cup \text{Reach}^{aux}(Succ_{i-1})$;
12: **end while**

Definition 7. (*Admissible function*) *We say that a function* $f : \mathbb{R}_0 \times \mathbb{R}^m \to \mathbb{R}^n$ *for some* $m, n \in \mathbb{N}$ *is admissible if for all* $x_1, x_2, x_3 \in R^m$ *and* $t \in \mathbb{R}_0$ *there are* $\alpha_1, \alpha_2 \in [0, 1]$ *such that* $x_3 = \alpha_1 x_1 + (1 - \alpha_1)x_2$ *and* $f(t, x_3) = \alpha_2 f(t, x_1) + (1 - \alpha_2)f(t, x_2)$.

The above definition means that while time elapses, each segment of a straight line is mapped to a segment of a straight line. In the following we consider only admissible functions.

4 Modeling of hybrid systems with Coq

We give here only a very brief description of the `Coq` proof assistant, and refer for more details to [3]. `Coq` is a proof assistant for higher order typed λ-calculus, i.e. the proof in `Coq` is equivalent to the construction of a λ-term.

`Coq` is an interactive proof assistant, composed of a specification language called Galina and a proof engine, and it allows formal defining mathematical objects and helps the user with proving the properties of these objects. Basically, the use of `Coq` follows three steps: a) define the objects and axioms used, b) state a theorem, c) provide proof steps, until the proof is complete.

In `Coq`, data types are presented as inductive types. For example, natural numbers are defined as follows.

```
Inductive nat: Set:= | 0 : nat | S : nat -> nat.
```

This definition introduces a type *nat* which is itself of type *Set*.

Formalization in `Coq`. We show here how an abstraction of a hybrid automaton can be formalized in `Coq`. A partition `P` of a continuous state space is defined as a list of lists or real numbers, an abstract continuous state is defined as a list of naturals and a set of all abstract states as a list of lists of naturals.

```
Parameter P: list(list R).
Definition AbsState:= list nat.
```

```
Definition SetAbsStates:= list(list nat).
Definition Time := R.
Definition DiscState:= Set.
Definition ContState:= list R.
Parameter InitialStates: DiscState → SetAbsStates.
Parameter Invariant: DiscState → list(list nat).
Parameter Guard: DiscretState → DiscreteState →list(list nat).
Parameter Flow: Time → State → State.
Parameter Jump: DiscState → DiscState → SetAbsStates.
Parameter Reset: DiscState → DiscState → SetAbsStates
→ SetAbsStates.
```

Verification example: the gate controller of a railroad crossing. As an example we consider the gate controller of a railroad crossing from [7]. The model consists of two subsystems: a train and the gate controller. The train is required to send a signal *app* at least two minutes before it enters the crossing. The train sends a signal *out* when it leaves the crossing and it must happen no late than 5 minutes after the *app* signal (this is expressed by the guard $2 < x \leq 5$). The gate must be closed in at least 1 minute after the *app* signal is received and not later than in 2 minutes (the guard is $1 \leq y < 5$). The gate responds by opening within 1 minute after receiving the *out* signal. We have defined a product of timed automata that combines the train process and the controller process, and the resulting system is depicted in Figure 3. It has the following discrete locations: Loc1: a train is far from the gate and the gate is open; Loc2: the train is approaching the crossing and the gate is open; Loc3: the train is approaching the crossing and the gate is closed; Loc4: the train has left the crossing and the gate is closed. We want to verify that the gate is never closed more than 5 minutes.

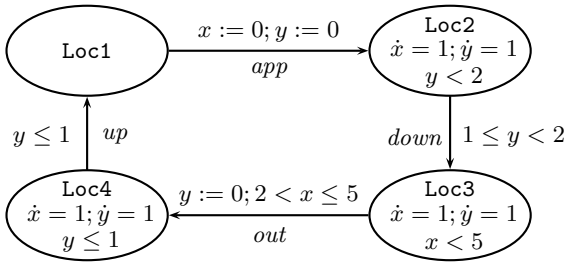


Fig. 3. A model of a train crossing problem.

We define the set of discrete states as:

`Inductive DS: Set:= | Loc1:DS | Loc2:DS | Loc3:DS | Loc4:DS`, and the partition of the continuous state space as:

```
Definition P := (0::1::2::3::4::5::6 )::(0::1::2::3::4::5::6).
```

We do not have enough of space to present all formalizations in `Coq`. We just conclude that we have been able to verify with our approach that the gate is never closed more than 5 minutes.

5 Conclusion

The results presented in the paper are twofold. On one hand, we have given a framework for reachability analysis which allows to avoid spurious transitions for some classes of hybrid systems. On the other hand, the results presented in the paper were 'mechanically' checked in `Coq` by formalizing a train crossing example. As for future work, we intend to investigate the structure of special classes, for example linear hybrid systems, for which our method is efficient, to generalize this approach within `Coq` and to apply it to larger case studies.

6 Acknowledgments

We are grateful for the support by Milad Niqui in the field of `Coq`, and for the support by Herman Geuvers and Dan Synek for the examples of hybrid systems formalizations in `Coq`.

References

1. Rajeev Alur, Thao Dang, and Franjo Ivancic. Reachability analysis of hybrid systems via predicate abstraction. *ACM transactions on embedded computing systems (TECS)*, 2004.
2. E. Asarin, T. Dang, and O. Maler. The **d/dt** tool for verification of hybrid systems. In *Computer Aided Verification: 14th International Conference*, pages 365–370. LNCS, 2002.
3. Yves Bertot and Pierre Casteran. *Interactive Theorem proving and Program Development*. Springer, 1998.
4. E. M. Clarke and O. Grumberg. Model checking and abstraction. *ACM Transactions on Programming Languages and Systems*, 1992.
5. A. R. Girard, A. S. Howell, and J. K. Hedrick. Model-driven hybrid and embedded software for automotive applications. In *Proceedings of the 2nd RTAS Workshop on Model-Driven Embedded Systems*, 2004.
6. T. A. Henzinger, P. W. Kopke, A. Puri, and P. Varaiya. What's decidable about hybrid automata? *Journal of Computer and System Sciences*, 57(1):94 – 124, August 1998.
7. T. A. Henzinger, Xavier Nicollin, Joseph Sifakis, and Sergio Yovine. Symbolic model checking for real-time systems. In *7th. Symposium of Logics in Computer Science*, 1992.
8. T. M. A. Colon and E. Uribe. Generating finite-state abstractions of reactive systems using decision procedures. In *Computer Aided Verification*, 1998.
9. A. Olivero, J. Sifakis, and S. Yovine. Using abstractions for the verification of linear hybrid systems. In *Proceedings of the sixth International Conference on Computer-Aided Verification (CAV)*, 1994.
10. B. I. Silva and B. H. Krogh. Formal verification of hybrid systems using checkmate: a case study. In *American Control Conference*, pages 1679 – 1683, 2000.
11. O. Stursberg, S. Kowalewski, I. Hoffmann, and J. Preussig. Comparing timed and hybrid automata as approximations of continuous systems. In *Hybrid Systems*, pages 361 – 377, 1997.
12. C. J. Tomplin, I. Mitchell, A. B. Bayen, and M. Oishi. Computational techniques for the verification of hybrid systems. *Proceedings of the IEEE*, 91(7), 2003.

Proceedings of the International Multiconference on
Computer Science and Information Technology pp. 893–902
ISSN 1896-7094
© 2007 PIPS

A speed classification method for real-time controlled dynamical systems

Paweł Piątek, Wojciech Grega

Department of Automatics
AGH University of Science and Technology
Al. Mickiewicza 30, 30-059 Cracow, Poland
ppi@ia.agh.edu.pl, wgr@ia.agh.edu.pl

Abstract. Traditionally, control algorithms are designed without a consideration of their real-time implementation details. The performance of a digital control system besides the sampling period depends on many variables, such as the control loop execution time, jitter, the complexity of the control algorithm etc. In this paper attention is focused on the interaction of the parameters of the scheduled tasks and on the performance of control loops closed with digital controller. A new design approach that is based on the relative speed classification of the control system have been proposed. The approach is illustrated by analysis of control systems developed for laboratory magnetic levitation process.

1 Introduction

Computer based digital controllers typically have the ability to monitor a number of discrete and analog inputs, execute complex control algorithms, and drive several outputs, all at defined, often very high speeds. In general, computer-based digital controller must detect external events and respond to them by taking appropriate actions. It is required that all above operations and calculations take place at the correct time. This imposes timing requirements on hardware and software of the computer-based control systems. More precisely, computer based digital controller must have sufficient processing power, sufficient high-speed input/output hardware peripherals and operating system, fulfilling more or less hard timing requirements and handling error conditions in a predefined way.

Limited processing power combined with a non-optimized hardware and software components introduce delays and non deterministic behavior in the real-time system. Digital control theory normally assumes evenly spaced sampling intervals and a negligible (or constant) control delay between sampling and actuation [1]. However, this can seldom be practically achieved in a real resource-constrained system. Time delays and timing variations in control loop execution degrade the control performance and may in extreme cases lead to instability.

Control theory does not very often advise how to design controllers to take that limitation into an account [2]. Usually, control algorithms are designed without consideration of their real-time implementation details. Designers usually try

to separate the real-time aspects and the dynamics of the control system. They develop controllers that guarantee all tasks deadlines under worst case load and external interrupt occurrence scenario. The design of safety-critical controllers is based on this approach. Plant can be suitably controlled, but at the cost of poor computer resource utilization. However, if sampling rate bound caused by the speed of the control computer is close to the minimal required by the plant, then the sampling rate of the control system becomes time critical. For such a system the performance of the real-time operating system is essential for correct operation of the control system.

It has been stated in the previous works (see for example [10, 12]), that integrated approaches combining two disciplines: real-time computation systems and control systems results in better quality for digital control systems. This is also true for networked or multirate systems [4, 6, 9].

This problem is analyzed in this paper. The notion of relative speed of the control system is introduced and illustrated, on the example of magnetic levitation (MagLev) real-time control. Control system design approach based on the relative speed system classification is proposed.

2 Relative speed of a digital-controlled plant

The general scheme of a digital control system is given in Fig. 1 [11, 3]. The operation of the closed-loop system can be split into three main tasks: sampling, control algorithm computation and actuation. Models and methods used by discrete-time control theory implicitly impose the timing of the tasks in the computer implementation.

The tasks are associated with the events i.e. timer event, termination of a data frame transmission, signals that data are ready to be read from the input devices, fault detection, etc. The tasks usually share the same processor and exchange data with each other.

Although a great variety of scheduling policies is available, in this work periodic task scheduling method is assumed.

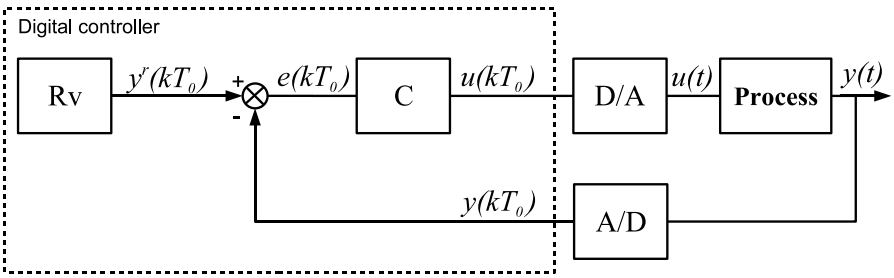


Fig. 1. The general scheme of a computer controlled system. A/D - analog to digital converter, D/A - digital to analog converter, Rv - reference value, C - controller

The following timing assumptions are made about three main tasks of a digital control loop.

1. Sampling is performed at equidistant time instants given by T_0, but some variation of are allowed.
2. The actuation is performed instantly when the control signals $u(kT_0)$ are delivered to the D/A converter.
3. The control algorithm computation is executed as soon as the input data are available.
4. The control algorithm design is based on correctly identified models of the process and the disturbances (referred as "nominal models").
5. For the nominal models it is possible to estimate maximal, admissible sampling period T_0 which would guarantee acceptable control performance. This sampling period can be estimated as:

$$T_0 \in \left[T_0^l, T_0^u\right] \tag{1}$$

where:
T_0^u—is time period for "ideal" control system, where modelling and identification errors as well as time delays and variations of time period T_0 are negligible,
T_0^l—is the sampling period guaranteeing robust operation of the control system, if it is under the influence of external and internal disturbances. This sampling/control period, besides fulfilling the Shannon theorem, can be chosen following one of various "rules of thumb" [1, 3], depending on the desired closed-loop system performance.

6. The performance of the closed-loop control system is a strictly monotonic function of T_0. Any applied sampling period $T_0 < T_0^u$ improves control performance. For $T_0 < T_0^l$ improvement is not observed.
7. The proposed control platform (processor, peripherals hardware and operating systems) are characterized by minimal (a shortest accessible) closed - loop execution time, estimated as:

$$\tau_s \in \left[\tau_s^l, \tau_s^u\right] \tag{2}$$

where:
τ_s^l—is the control loop execution time for simple control algorithms,
τ_s^u—is the control loop execution time for complex control algorithms.

The control algorithm is classified as a "simple", if the pseudocode of the controller task includes no more than 5-10 operations (loops are excluded). The examples of "simple" algorithms are: incremental PID or state feedback controller. If the pseudocode of the controller includes more than 10 operations or loops are included than algorithm is classified as "complex". The examples of "complex" control algorithms are: time-optimal, model-reference controller, predictive controller.

Fig. 2 illustrates typical timing models one can use for regularly sampled process.

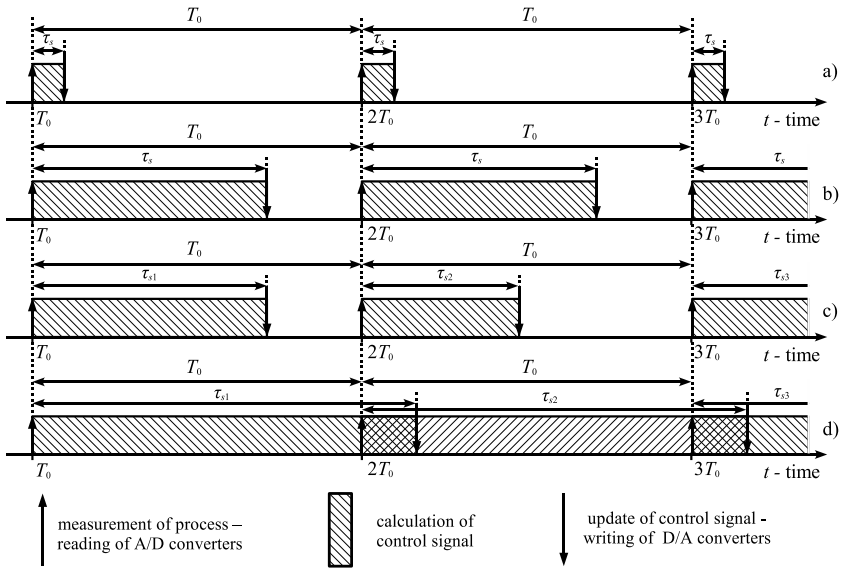


Fig. 2. Timing models that can be used for regularly sampled process

For the case a) sampling and actuation are performed at the same sampling time (time delay is zero). For this model we have: $\tau_s \ll T_0$ and so called *causality rule* is fulfilled.

Model b) from Fig. 2 is more realistic because it takes into account that control task takes time. Control loop execution time is constant and is less than sampling period: $\tau_s < T_0$. Causality rule is not fulfilled in this case. The causality rule can be fulfilled if actuation is performed at the next sampling instant, e.g. one step delay is assumed. It is so called *Strictly Proper Control Law* [5].

Model c) also takes into account that control task takes time, but time delay is variable and is less than sampling period: $\tau_s < T_0, \tau_{s1} \neq \tau_{s2}$. The causality rule can be fulfilled if the actuation is performed at the next sampling instant, i.e. one step delay is assumed.

For the constant delay it is straightforward to compensate for it during the algorithm design. For continuous-time design Smith predictor can be implemented, for discrete-time design plant model augment method can be applied [3].

For model d) time delays are longer than the sampling period. Both causality rule and real-time constraints are not fulfilled in this case.

The relative speed of control system can be characterized by the factor: $\frac{\tau_s}{T_0}$. The following classification of control system is proposed.

1. The control system will be referred as **relatively low-speed** if $\frac{\tau_s}{T_0} \ll 1$ (model from Fig. 2a).

2. The control system will be referred as **relatively medium-speed** if and $\frac{\tau_s}{T_0} \approx 1$ (model from Fig. 2b and Fig. 2c).
3. The control system will be referred as **relatively high-speed** if $\frac{\tau_s}{T_0} > 1$ (model from Fig. 2d).

It is obvious, that high-speed execution platform applied for the process described by a slow dynamic is classified as relatively low-speed control system. A digital control system will be consider as **time-critical** if it is classified as relatively high-speed or relatively medium-speed. Consistently, low-speed control loop applied for fast dynamic process will be classified as relatively high-speed control system.

3 Adapting control system parameters for time-critical system

For $\frac{\tau_s}{T_0} > 1$ the control loop is relatively high-speed and therefore becomes time-critical. It can be classified as medium- speed system, if (Fig. 3):

a) after assuming $T_0 = T_0^u$ the condition $\tau_s < T_0^u$ is fulfilled. Non-robust control is available in this case,
b) after assuming $\tau_s = \tau_s^l$ the condition $\tau_s^l < T_0$ is fulfilled. Applications of "simple" control algorithms becomes possible.

Such adaptation of control system parameters is limited and in most cases can not move the system to low-speed class. This class can be reached be changing the control platform: using the most efficient processor, better operating system, etc. (case B in Fig. 3).

4 Example: real-time control of MagLev system

A laboratory magnetic levitation system presented on Fig. 4 was used as an example of relatively high-speed plant [7, 8]. The MagLev was chosen with regard to its specific nonlinearities and fast dynamics.

The identified parameters of MagLev process were: $T_0^l = 40\mu s$, $T_0^u = 2000\mu s$. The parameters of the MagLev digital control system and its relation to the proposed relative-speed definitions are given in Table 1.

This plant, when controlled by PC or general-purpose microcontroller, can be classified as relatively medium-speed system. MagLev controlled by simple 8-bit microcontroller is classified as relatively high-speed system. It is a representative example to show that:

- usage of more efficient control system with shorter control period T_0 results in better quality of control by changing classification of plant form relatively medium-speed to relatively low-speed,

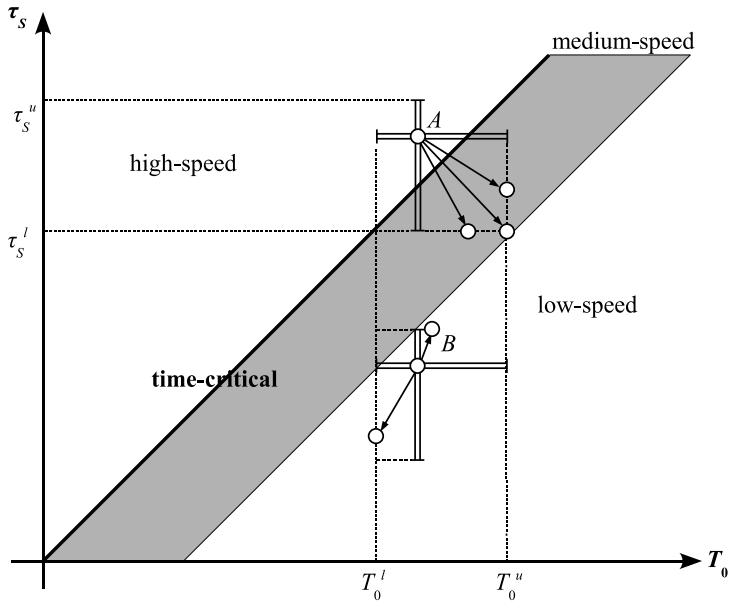


Fig. 3. Moving time critical control system (A) to the medium speed class (adapting the parameters) and to low speed class (B), by changing the control platform

– usage of more efficient control system with shorter control loop execution time τ_s results in better quality of control by better satisfying the causality rule.

Table 1. System parameters for MagLev

	PC	**microcontroller**	**FPGA**
τ_s^l	$500\mu s$	$2000\mu s$	$2\mu s$
τ_s^u	$2000\mu s$	n.a.	n.a

Experiments with MagLev controlled by two different control systems were carried out. The first one was based on PC computer and the second was based on FPGA circuits. Both of them were developed using an extention board consisting the FPGA circuit. Control algorithm for PC-based control system was calculated as a controller task by MATLAB/Simulink real-time application. The FPGA circuit was used only for signal generation for analog/digital and digital/analog converters and for providing data to the PC computer. In case of

FPGA-based control system the control algorithm was calculated directly by the FPGA circuit. The PC was used only for monitoring the process and logging the data. Both of described control systems are presented in the Fig. 4.

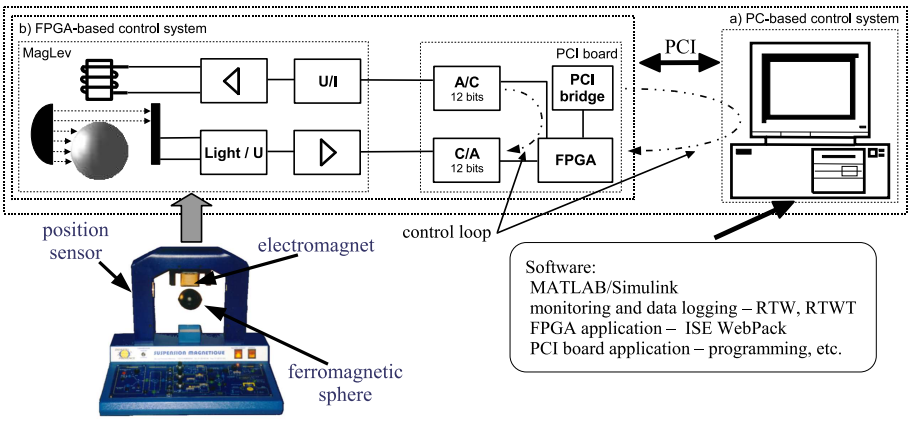


Fig. 4. Block diagram of control systems: a) PC-based, b) FPGA-based

The ferromagnetic ball has followed the changing reference value signal. The square wave with $2s$ period was applied as the reference value. The centre of the ball movement was in the distance of $0.0125m$ from the electromagnet and the amplitude of movement was equal to $0.002m$. Exactly the same parameters were used for both of the tested control systems.

PID controller algorithm was used for both performed experiments. The parameters of the algorithm were recalculated for the specific hardware architecture i.e. the type of arithmetic or the applied control period. The results of the experiments are presented in Fig. 5, Fig. 6 and Fig. 7.

A comparison of the PC controlled process response with the response of control system based on FPGA circuit is presented in Fig. 5. The control period of $700\mu s$ was used for both control systems. A smaller values of the overshoot and the shorter settling times are observed in the case of MagLev controlled by FPGA-based controller that guarantees shorter control loop execution time τ_s.

A comparison of the process response (FPGA-based control system) for different control periods—$700\mu s$ and $40\mu s$, is presented in Fig. 6. A smaller values of the overshoot and the shorter settling time are observed in the case of MagLev controlled by FPGA-based control system with the shorter control period T_0.

A comparison of the process response controlled by PC computer and the control system based on FPGA circuits, is presented in Fig. 7. A smaller values of the overshoot and the shorter settling time are observed in the case of MagLev controlled by FPGA-based control system with the boundary control period T_0^l and the boundary control loop execution time τ_s^l.

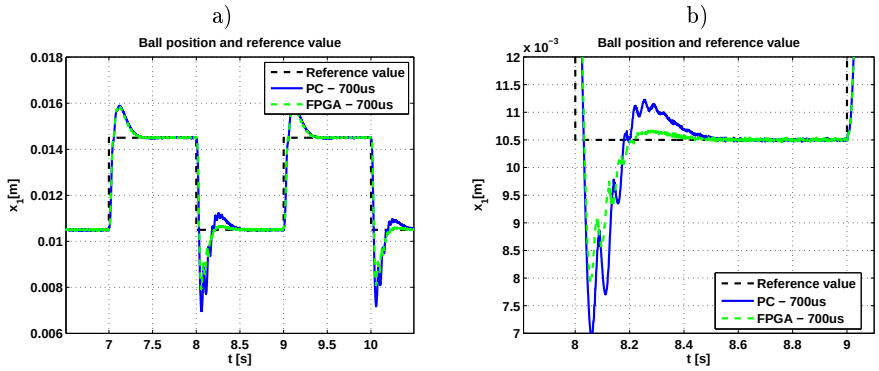


Fig. 5. A comparison of the MagLev process response controlled by PC system ($700\mu s$ control period) and the system based on FPGA circuits ($700\mu s$ control period): a) complete range of desired ball movement, b) magnification of up-motion part

5 Conclusions

Digital, computer based control systems are the applications that pose the very sharp timing requirements, because digital control theory usually assumes a highly deterministic sampling. Consequently, the application of the controller performing a number of tasks in the real-time into control loop makes the analysis and design of such applications more complex.

The performance of a digital control system depends on many variables, such as sampling period, control loop execution time, jitter, complexity of the control

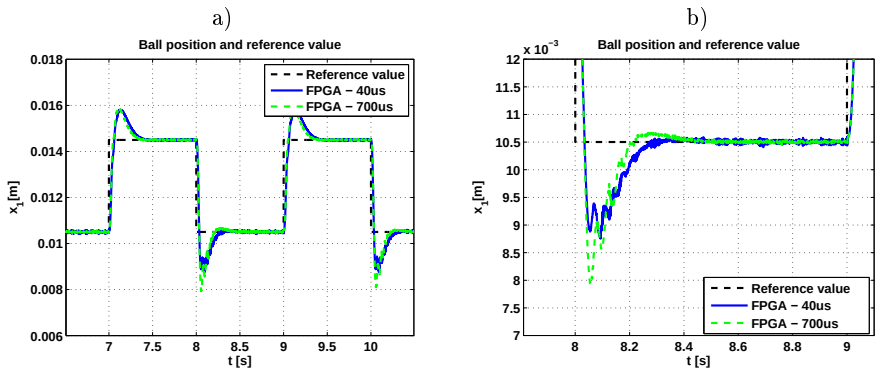


Fig. 6. A comparison of the MagLev process response controlled by the system based on FPGA ($700\mu s$ control period) and the system based on FPGA circuits ($40\mu s$ control period): a) complete range of desired ball movement, b) magnification of up-motion part

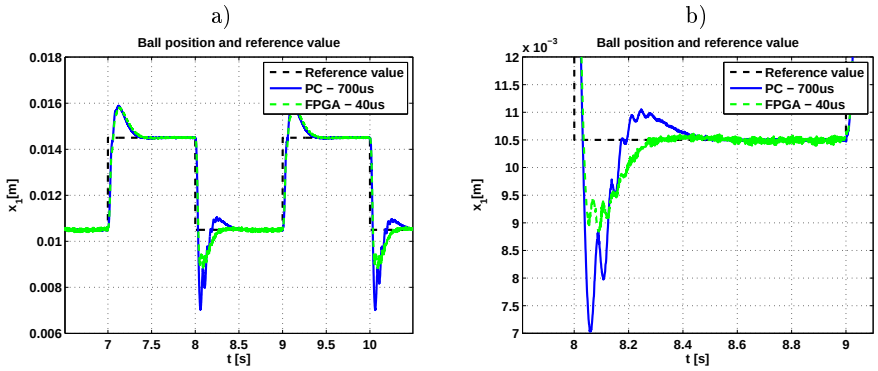


Fig. 7. A comparison of the MagLev process response controlled by the system based on PC ($700\mu s$ control period) and the system based on FPGA circuits ($40\mu s$ control period): a) complete range of desired ball movement, b) magnification of up-motion part

algorithm. The performance of a digital control system depends not only on the performance of its individual components but also on their interaction and cooperation.

The focus of this paper is the interaction of parameters of the scheduled tasks and the performance of control loops closed over digital controller. In particular, we have proposed a new design approach that is based on the relative speed system classification and applies the following paradigm: the application platform should be selected in such a way that closed-loop execution time and process dynamics are balanced. The relative speed of the system should be located just bellow the line separating "time critical" solution, giving optimal utilization of computing system resources.

The results of presented experiments have showed that reduction of the control period improves quality of control for magnetic levitation system. Both solutions have located the relative speed of the MagLev control system below the "time-critical" range. The obtained overshoot and settling time were better with FPGA-based controller (it has reduced control period below $40\mu s$). In the other case, better quality of control has been obtained by reduction a control loop execution time. The maximal improvement of control quality was observed for simultaneously reduced control period and control loop execution time. This effect was obtained for control loop closed directly via PID control application embedded into FPGA circuit.

Acknowledgments

This research has received support from the AGH University of Science and Technology.

References

1. Aström K. J., Wittenmark B.: *Computer Controlled Systems,* Prentice Hall, London (1997)
2. Arzen K-E., Cervin A., Eker J., Sha L.: *An Introduction to Control and Scheduling Co-design,* 39th IEEE Conference on Decision and Control, Australia, Sydney (December 12–15, 2000).
3. Franklin G. F., Powell J. D., Emami-Naeini A.: *Feedback Control of Dynamic Systems,* Addison-Wesley Publishing Company, London (1994).
4. Grega W.: *Stability of Distributed Control Systems with Uncertain Delays,* 8th IEEE International Conference on Methods and Models in Automation and Robotics, Miedzyzdroje (2002) 303–307.
5. Middleton R. H., Goodwin G. C.: *Digital Control and Estimation: A Unified Approach,* Prentice-Hall International, Englewood Cliffs, New Jersey (1990).
6. Nilsson J.: *Real-time Control Systems with Delays,* Ph.D. Dissertation, Sweden, Lund Institute of Technology (1998).
7. Piatek P.: *Application of Specialized Hardware Architectures for Realization of Time-critical Control Tasks,* PhD. Thesis (in Polish), Grega W. (supervisor). AGH University of Science and Technology, Department of Automatics, Poland, Krakow (2007).
8. Pilat A.: *Programmable Analog Hardware for Control Systems Exampled by Magnetic Suspension,* In Proc. Computer Methods and Systems, Poland, Krakow (November 14–16, 2005) 143–148.
9. Sågfors M.: *Optimal Sampled-Data and Multirate Control,* PhD. Dissertation, Process Control Laboratory, Faculty of Chemical Engineering, Åbo Akademi University (1998).
10. Schinkel G., Rantzer A.: *Sample Data with Varying Sampling Time,* In Procceding of European Control Conference, Portugal, Porto (September, 2001) 624–628.
11. Vaccaro R. J.: *Digital Control a State-Space Approach,* MCGraw-Hill, Inc., New York (1995).
12. Wittenmark B., Bastian B., Nilsson J.: *Analysis of Time Delays in Synchronous and Asynchronous Control Loops,* Proc. of 37th IEEE Conference on Decision and Control, USA, Tampa (December 12–15, 1998).

Proceedings of the International Multiconference on Computer Science and Information Technology pp. 903–912
ISSN 1896-7094
© 2007 PIPS

Dependability of Explicit DMC and GPC Algorithms

Piotr Gawkowski[1], Maciej Ławryńczuk[2], Piotr Marusak[2], Janusz Sosnowski[1], and Piotr Tatjewski[2]

[1] Institute of Computer Science
{P.Gawkowski, J.Sosnowski}@ii.pw.edu.pl
[2] Institute of Control and Computation Engineering
{M.Lawrynczuk, P.Marusak, P.Tatjewski}@ia.pw.edu.pl
Warsaw University of Technology, ul. Nowowiejska 15/19, 00-665 Warszawa, Poland

Abstract. This paper studies dependability of software implementation of DMC (Dynamic Matrix Control) and GPC (Generalised Predictive Control) Model Predictive Control (MPC) algorithms. Explicit formulation of algorithms is considered in which the control laws are calculated off-line. Dependability is evaluated usig software implemented fault injection approach. Tests are performed in the control system of a remotely controlled robot vehicle used in nuclear plants.

1 Introduction

Faults (transient, permanent or intermittent) appearing during system operation may result in logical errors, which can be critical for the realised applications [1, 8]. Transient faults are especially critical as they dominate in contemporary technologies. Hence, an important practical issue is to evaluate dependability of software applications in the presence of faults. It is particularly critical in many reactive systems (e.g. nuclear plants, satellites, aircrafts, chemical industry, medicine). One kind of such applications, erroneous behaviour of which might have some serious consequences, are control algorithms. This paper studies dependability of software implementations of explicit versions of DMC (Dynamic Matrix Control) and GPC (Generalised Predictive Control) Model Predictive Control (MPC) algorithms. The investigation is based on software implemented fault injection, which has been adapted to reactive applications [1, 5]. In particular, a new approach to test result qualification is proposed.

MPC is recognised as the only advanced control technique which has been very successful in practical applications [11–15, 19, 20]. As MPC algorithms use models of processes for calculation of the control policy they can be successfully applied to processes which are difficult to control. In particular, processes with significant time delay for which the PID controller does not give satisfactory control performance. Among different MPC techniques, DMC [3] and GPC algorithms [2] are the most popular. Both algorithms use linear models. The DMC algorithm uses a non-parametric model consisting of step-response coefficients

of the process. Such a model can be easily obtained in industry, but to precisely describe a process many coefficients are needed. The GPC algorithm uses a parametric model in the form of a discrete difference equation. Usually, such models are significantly less complicated in terms of the number of parameters than the step-response ones.

The paper is structured as follows. First, in Section 2, the investigated control algorithms and their explicit formulations are shortly characterised. Next, Section 3 presents the fault injection testbed, experiment set-up and some new aspects of adapting experiments to control algorithms specificity. Finally, Section 4 discusses experimental results and the paper is concluded in Section 5.

2 Model Predictive Control Algorithms

In the MPC algorithms [12–15, 19] at each consecutive sampling instant k a set of future control increments

$$\Delta \boldsymbol{u}(k) = [\Delta u(k|k)\ \Delta u(k+1|k) \dots \Delta u(k+N_u-1|k)]^T \tag{1}$$

is calculated assuming that $\Delta u(k+p|k) = 0$ for $p \geq N_u$, where N_u is the control horizon. It is usually done in such a way that the future control error values (i.e. differences between the reference trajectory and the predicted values of the output) are minimised over the prediction horizon N. The following quadratic cost function is typically used

$$J(k) = \left\| \boldsymbol{y}^{ref}(k) - \hat{\boldsymbol{y}}(k) \right\|^2 + \left\| \Delta \boldsymbol{u}(k) \right\|^2_{\boldsymbol{\Lambda}} \tag{2}$$

where $\boldsymbol{y}^{ref}(k) = \left[y^{ref}(k) \dots y^{ref}(k) \right]^T$, $\hat{\boldsymbol{y}}(k) = [\hat{y}(k+1|k) \dots \hat{y}(k+N|k)]^T$ are vectors of length N, $\boldsymbol{\Lambda} = diag(\lambda_0, \dots, \lambda_{N_u-1})$, $\lambda_p > 0$ are weighting factors, $y^{ref}(k)$ is the reference (i.e. the set-point) and $\hat{y}(k+p|k)$ are predicted values of the output.

Typically, $N_u < N$, which decreases the dimensionality of the optimisation problem and leads to smaller computational load. Because only the first element of the determined sequence (1) is applied to the process, the control law is

$$u(k) = u(k-1) + \Delta u(k|k) \tag{3}$$

At next sampling instant, $k+1$, the prediction is shifted one step forward and the whole procedure is repeated. Predicted output values $\hat{y}(k+p|k)$ are calculated using a dynamic model of the process.

When a linear dynamic model of the process is used, it is possible to express the output prediction as the sum of a forced trajectory (which depends only on the future input moves $\Delta \boldsymbol{u}(k)$) and a free trajectory $\boldsymbol{y}^0(k)$ (which depends only on the past)

$$\hat{\boldsymbol{y}}(k) = \boldsymbol{y}^0(k) + \boldsymbol{G} \Delta \boldsymbol{u}(k) \tag{4}$$

where $\boldsymbol{y}^0(k) = \left[y^0(k+1|k) \dots y^0(k+N|k) \right]^T$. The dynamic matrix $\boldsymbol{G}$ of dimensionality $N \times N_u$ is comprised of step-response coefficients of the model.

Thanks to using a linear model and the superposition principle (4), optimisation of the cost function (2) becomes a quadratic programming task. Hence, the vector of optimal control input increments is

$$\Delta \boldsymbol{u} = \boldsymbol{K}(\boldsymbol{y}^{ref}(k) - \boldsymbol{y}^0(k)) \tag{5}$$

where $\boldsymbol{K} = (\boldsymbol{G}^T\boldsymbol{G} + \Lambda)^{-1}\boldsymbol{G}$ is a matrix of dimensionality $N_u \times N$ which is calculated off-line.

2.1 Dynamic Matrix Control Algorithm

In the DMC algorithm the process dynamics is described, in a convenient way, by a discrete-time, finite step-response model. Thus, for any sampling instant k, output of the model is

$$y(k) = y(0) + \sum_{j=1}^{D} s_j \Delta u(k-j) \tag{6}$$

where s_i are step-response coefficients, D is the horizon of the process dynamics.

The DMC control law (5) can be expressed in the following form [12, 19]

$$\Delta u(k) = \Delta u(k|k) = k^e(y^{ref}(k) - y(k)) - \sum_{j=1}^{D-1} k_j^u \Delta u(k-j) \tag{7}$$

where k^e, k_j^u, $j = 1, \ldots, D-1$ are coefficients calculated off-line. The total number of parameters is D. The obtained explicit control law is a linear feedback from the difference between the set-point trajectory and values of the manipulated variable increments calculated at previous sampling instants. The structure of the explicit DMC algorithm is shown in Fig. 1.

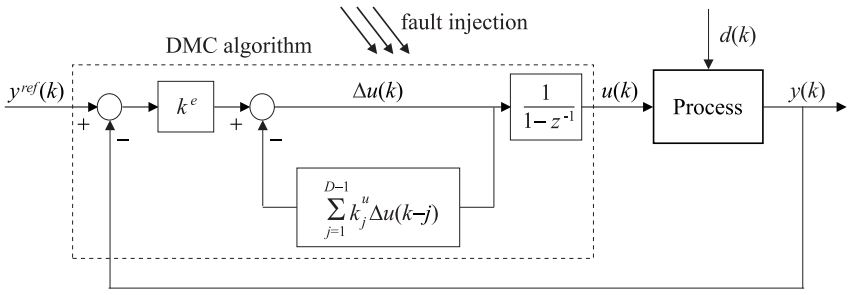


Fig. 1. Structure of the explicit DMC algorithm

2.2 Generalised Predictive Control Algorithm

The GPC algorithm uses a process model in the form of a discrete difference equation describing the process input-output relation

$$y(k) = \sum_{i=1}^{n_B} b_i u(k-i) - \sum_{i=1}^{n_A} a_i y(k-i) \tag{8}$$

where a_i, b_i are coefficients and n_A, n_B define order of the dynamics.

The GPC control law can be expressed in the following form [19]

$$\Delta u(k) = \Delta u(k|k) = k^e y^{ref} - \sum_{j=1}^{n_B} k_j^u u(k-j) - \sum_{j=0}^{n_A} k_j^y y(k-j) \tag{9}$$

where k^e, k_j^u, $j = 1, \ldots, n_B$ and k_j^y, $j = 0, \ldots, n_A$ are coefficients calculated off-line. The total number of parameters is $n_A + n_B + 2$. The obtained explicit GPC control law is a linear feedback from the reference trajectory, values of the manipulated variable calculated at previous sampling instants and values of the controlled variable measured at previous sampling instants. Structure of the explicit GPC algorithm is shown in Fig. 2. Comparing the structures of both studied algorithms, it is evident that in the DMC algorithm, there is only one feedback from the process output variable and $D-1$ feedbacks from values of past process input increments. In the GPC algorithm, there are feedbacks from the current process output value and last n_A past values of the output increments and from last n_B past values of the process input increments.

The step-response usually contains a large number of elements. At the same time, the process can be described precisely enough by a discrete difference equation of a relatively low order. Hence, a model used in the GPC algorithm has significantly less parameters than the model used in the DMC algorithm (i.e. $D \gg n_A$, $D \gg n_B$). It should be stressed, however, that a non-parametric step-response model is obtained on the basis of a simple experiment but it is necessary to conduct a full identification experiment to obtain a parametric model.

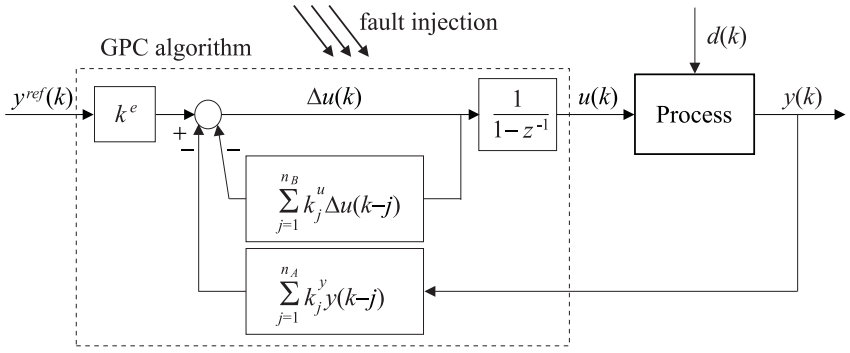


Fig. 2. Structure of the explicit GPC algorithm

3 Fault Injector and Experiment Set-up

The concept of the SoftWare Implemented Fault Injector (SWIFI) is based on the software emulation of a fault during the run-time of the application under test. In this research FITS fault injector is used [7, 17]. FITS is used for many years in research projects and proves to be the one of the most complex, flexible and easy to use SWIFI tool reported in the literature. It uses standard Win32 Debugging API to control the execution of the software application under test. In the whole process the following steps can be distinguished: optional source code instrumentation of the tested application, golden run execution, experiment configuration, fault injection and results analysis.

In order to assure good experiment controllability, the so called *testing areas* are introduced [5]. FITS disturbs directly the application only within those areas. They can be marked in several ways (e.g. with predefined magic sequence at the source code level by subset of code modules). Testing areas can limit the scope of disturbances only to the most interesting parts of the application, making further analysis easier. Additionally, some code within the application can be situated outside disturbance area during experiments. Here, the code of the controlled process model is added. The parts of the tested applications disturbed during the experiments (dashed box) as well as process models (not disturbed) are marked in Fig. 1 and 2. To simplify tracing fault effects some extra modifications of the tested application are introduced. The *user-messages* captured and collected by the FITS during experiments are inserted [17]. This mechanism provides supplementary communication between the tested application and the FITS. As a result, the tested application signals some measures related to internal variables values, output signal deviations etc.

During the so called *Golden Run* (GR – reference execution without faults) the execution trace and reference results are logged. Additionally, statistic information is collected on the tested application (e.g. resource usage, code size, instruction distribution). Those measures help to interpret experimental results and to profile experiments to be done (e.g. by elimination of injecting faults into unused resources). FITS simulates faults by disturbing the running application. In this process an important issue is policy of selection the type, location and time of fault injection (fault triggering). In this study single bit-flip faults within CPU and FPU registers, applications' data and machine instruction code are considered. Faults are injected pseudorandomly in time (program execution) and space (bit position within disturbed resource, distribution over applications' memory). There is a common consensus in the literature that such fault model well mimics Single Event Upset (SEU) effects. The experience also shows that such disturbances give the most interesting results for the fault susceptibility analysis [8]. Deeper discussion upon the fault injection policy can be found in [6, 8, 16].

An important issue (frequently neglected in the literature) is qualification of experimental results. In case of a typical calculation-oriented application correctness of its result is usually easy. It is much more complicated for other classes of applications, especially related to real-time systems [16]. Control algorithms

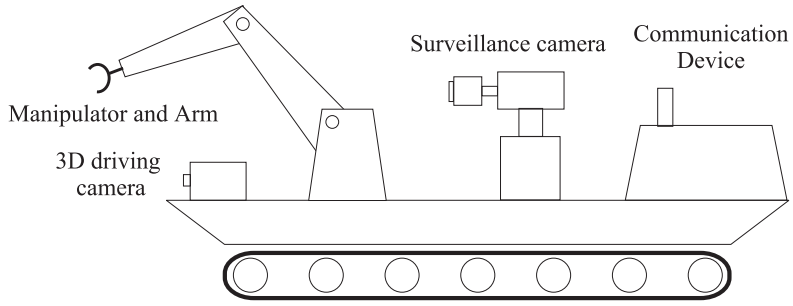


Fig. 3. Remotely controlled robot for nuclear plants

require complex analysis of the controlled process behaviour. In FITS the oracle procedure (in an external library) can be specified to examine the tested application outputs in accordance with an application's specificity [8, 16, 17].

3.1 Experiment set-up

The process under consideration is a remotely controlled robot vehicle for nuclear plants [4] shown in Fig. 3. The vehicle must act reliably in an hazardous and unsafe environment. Application of such vehicles can significantly reduce radiation exposure to personnel and improve maintenance programme performance. Discrete-time model of the process is (the sampling time is 0.5 min)

$$y(k) = b_9 u(k-9) - b_{10} u(k-10) - a_1 y(k-1) - a_2 y(k-2) \qquad (10)$$

where $b_9 = 0.022276$, $b_{10} = 0.019823$, $a_1 = -1.683638$, $a_2 = 0.704688$. Input u of the controlled process (output of the control algorithm in Fig. 1 and 2) is the voltage applied. The vehicle model (*Process* box in Fig. 1 and 2) limits the voltage range to $[-5; 5]$. Output y of the process is the velocity of the vehicle. Values of y at each sampling instant are considered to be the result of the whole application and are subject to correctness analysis. The input of the whole application (y^{ref}) is the required vehicle's velocity, a single step from 0 to 1 occurring at $k = 10$ is considered during the tests. The DMC algorithm uses the step-response model obtained from the model (10) (the horizon of the process dynamics $D = 100$) whereas the GPC algorithm uses directly the model (10). In both algorithms $N = 20$, $N_u = 5$ and $\lambda_p = 1$.

The whole experiment is conducted by FITS automatically. At the end of the experiment synthetic (aggregated) results for each fault location are given. In general, 4 classes of test results are distinguished:

- C: correct vehicle behaviour (ISE<20),
- INC: incorrect (unacceptable) vehicle behaviour (ISE≥20),
- S: test terminated by the system due to un-handled exception,
- T: timed-out test.

where the standard factor ISE (Integrated Sum of Squared Errors) is proposed as a measure of result (y) correctness. The reference ISE value (obtained during GR) is 12.79 (due to delayed vehicle response). System exceptions (S) are generated by hardware mechanisms embedded in contemporary COTS (commercial off-the-shelve) systems, e.g. memory access violation detection.

Analysis of fault effects requires detail information upon the faults injected and the application behaviour. FITS can provide details about every test (simulated fault injection) that allow manual replay the whole test execution. Moreover, all the events and user messages occurring during the test are recorded. The tested application is instrumented to save its outputs (here simulation results, i.e. a set of control signals in subsequent sampling instants) into separate files for each test (file names managed by FITS). This gives a possibility for deeper analysis (post-experiment) of fault effects in the correlation with the injected fault and observed behaviour for each single test.

4 Experimental Results

The cost of the DMC algorithm implementation is 173 bytes of binary code (50 machine instructions). At the same time, the GPC algorithm implementation takes 212 bytes (59 instructions). The main difference is the number of executed instructions, i.e. a single simulation execution of the DMC and GPC algorithms takes 121261 and 12192 machine instructions, respectively. Such a result is not surprising as the control law used in the DMC algorithm has 100 parameters whereas the GPC control law has only 14 parameters.

Fig. 4 depicts summarised results of experimental evaluation of DMC and GPC algorithms (results categories are described in Section 3). The main difference can be seen in the case of faults located in the data area used by the considered algorithms. As the DMC algorithm uses much more parameters (100) than the GPC one (14), it is more insensitive to single disturbances.

It is worth noting, that both algorithms can be further hardened by introducing exception handling. In the experiments carried out, no exception handling is present. Hence, approximately 50% of faults affecting the static image of the code, executed instruction stream and CPU registers resulted in unhandled exceptions (most of them relate to memory access violations). Previous experience shows [6, 8–10, 16, 17] that such behaviour can be efficiently improved.

A great number of tests have been performed (more then 25000 per algorithm) and their results compared. Because of limited space only two time-plots are presented here. Fig. 5 compares simulation results of the golden run (solid lines) and two example incorrect vehicle behaviour (dashed lines). In the first case (top figures), fault injected into static code of the DMC algorithm implementation results in slow response and big control errors (ISE=31.44). It leads to unacceptable output response (slow and with big control errors). Moreover, the manipulated variable (u) oscillations appear (the top left plot). In the second case (bottom figures) the responses from a different experiment with the DMC algorithm are shown (fault also injected into the static code). This time violation

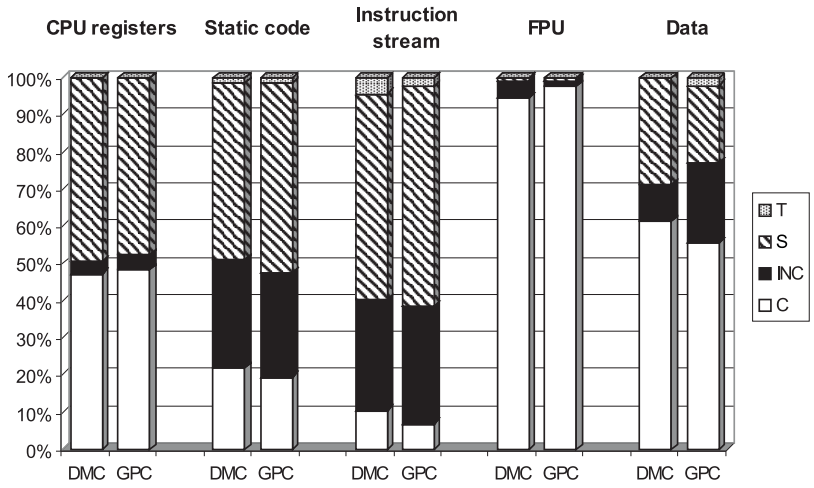


Fig. 4. Summary of the experimental results

of the manipulated variable constraint is observed (as shown in the bottom left graph). It would result in an oscillatory movement of the robot forward and back repeatedly.

Fig. 6 shows the distribution of ISE values observed (faults located in the static code) for tests considered as unacceptable (INC). It is worth noting that both the DMC and GPC algorithms have similar distribution for low ISE values (< 100). Moreover, more than 66% of the INC tests have very high ISE values (≥ 100), which means that the control system is unstable. Obviously, other correctness measures should be also considered and need further development and investigation.

5 Conclusions

The paper presents a novel approach to evaluation and comparison of dependability of software implementation of two most popular MPC algorithms, i.e. DMC and GPC. For this purpose software implemented fault injector is used. Explicit formulation of these algorithms is considered in which the control laws are calculated off-line. The experiments show that the performance of both algorithms is different in the case of some faults (in data) as well as their faults susceptibility. It is worth noting that robustness of software implementation can be improved by some basic software fault detection/tolerance techniques. The new measures of process behaviour are also considered to be developed. Those topics will be covered in the future research.

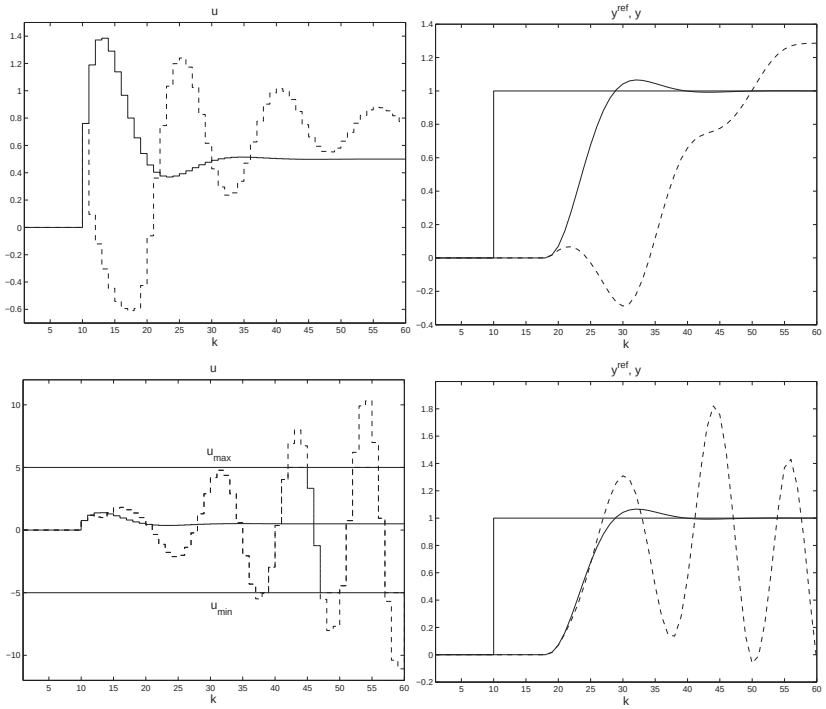


Fig. 5. DMC algorithm simulation results: **top:** slow response and big control errors (ISE=31.44); **bottom:** manipulated variable u constraint violation (ISE=23.94); solid line – golden run, dotted line – incorrect vehicle behaviour

Acknowledgement. This work was supported by a research grant from the dean of the Faculty of Electronics and Information Technology, Warsaw University of Technology.

References

1. Benso A., Prinetto P.: *Fault injection techniques and tools for embedded systems reliability evaluation,* Kluwer Academic Publishers. (2003).
2. Clarke D. W., Mohtadi C., Tuffs P. S.: *Generalized predictive control—I,* The basic algorithm. Automatica. **23** (1987) 137–148.
3. Cutler R., Ramaker B.: *Dynamic matrix control—a computer control algorithm,* AIChE National Meeting, Houston. (1979).
4. Dorf C. D.: *Modern control systems,* Addison-Wesley. Reading. (1995).
5. Gawkowski P., Sosnowski J.: *Experiences with software implemented fault injection,* Int. Conf. on Architecture of Computing Systems. VDE Verlag GMBH. (2007) 73–80.
6. Gawkowski P., Sosnowski J., Radko B.: *Analyzing the effectiveness of fault hardening procedures,* 11th IEEE Int. On-Line Testing Symposium. (2005) 14–19.

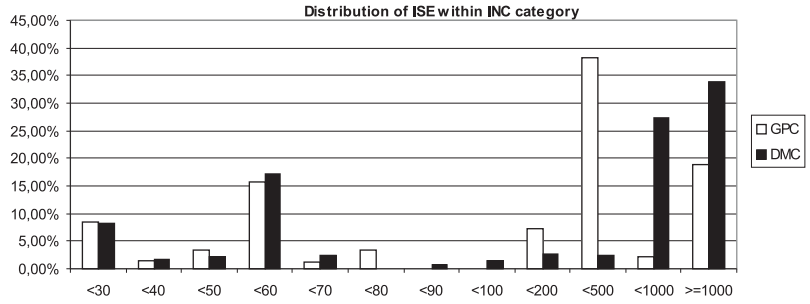


Fig. 6. Distribution of ISE in incorrect category

7. Gawkowski P., Sosnowski J.: *Analysing system susceptibility to faults with simulation tools,* Annales UMCS Informatica AI. **4** (2006) 123–134.
8. Gawkowski P., Sosnowski J.: *Dependability evaluation with fault injection experiments,* IEICE Transactions on Information & System. **E86-D** (2003) 2642–2649.
9. Gawkowski P., Sosnowski J.: *Experimental Validation Of Fault Detection And Fault Tolerance Mechanisms,* 7th IEEE Int. Workshop on High Level Design Validation And Test. Cannes (2002).
10. Gawkowski P., Sosnowski J.: *Analyzing fault effects in fault insertion experiments,* Proc. of On-line Testing Workshop. IEEE Computer Society Press (2001) 21–24.
11. Ławryńczuk M., Marusak M., Tatjewski P.: *Multilayer and integrated structures for predictive control and economic optimisation,* IFAC Symposium on Large Scale Systems, Gdańsk (2007), accepted for publication.
12. Maciejowski J. M.: *Predictive control with constraints,* Prentice Hall. Harlow. (2002).
13. Morari M., Lee J. H.: *Model predictive control: past, present and future,* Computers and Chemical Engineering. **23** (1999) 667–682.
14. Qin S. J., Badgwell T. A.: *A survey of industrial model predictive control technology,* Control Engineering Practice. **11** (2003) 733–764.
15. Rossiter J. A.: *Model-based predictive control,* CRC Press, Boca Raton. (2003).
16. Sosnowski J., Gawkowski P., Lesiak A.: *Fault injection stress strategies in dependability analysis,* Control and Cybernetics. **33** (2005) 679–699.
17. Sosnowski J., Lesiak A., Gawkowski P., Włodawiec P.: *Software implemented fault inserters,* IFAC Workshop on Programmable Devices and Systems. Ostrava (2003) 293–298.
18. Sosnowski J., Gawkowski P., Lesiak A.: *Fault injection stress strategies. 4th IEEE LATW 2003 Workshop,* (2003) 258–263.
19. Tatjewski P.: *Advanced control of industrial processes, Structures and algorithms,* Springer. London. (2007).
20. Tatjewski P., Ławryńczuk M., Marusak M.: *Linking nonlinear steady-state and target set-point optimisation for model predictive control,* IEE Int. Control Conference ICC 2006, Glasgow (2006), CD-ROM.

Proceedings of the International Multiconference on
Computer Science and Information Technology pp. 913–922
ISSN 1896-7094
© 2007 PIPS

Incorporating Fault Tolerance into Component-based Architectures for Embedded Systems

Shourong Lu and Wolfgang A. Halang

Fernuniversität in Hagen
Chair of Computer Engineering
58084 Hagen, Germany
{Shourong.Lu, Wolfgang.Halang}@FernUni-Hagen.de

Abstract. A component-based software architecture is presented to support the process of designing and developing fault-tolerant computerised control systems. To this end, we combine an idealised fault-tolerant component, the C2 architecture style and protective wrappers, and embed fault tolerance techniques into component definitions. The resulting architecture is described by normal- and abnormal-activity components aiming to support a wide range of fault tolerance features. Use of this architecture enables to reason about system dependability already from the earliest development stages on, and to customise fault tolerance strategies according to application characteristics.

1 Introduction

Fault prevention, fault tolerance, fault removal, and fault forecasting are the four main means to attain the various attributes of dependability [4]. Fault prevention and fault tolerance aim to prevent introducing faults, or to avoid service failures when faults occur. Fault removal and fault forecasting, instead, mean to reduce number and severity of faults, and to estimate present and future incidences and consequences of faults. Among them, fault tolerance is the most promising mechanism to meet the dependability requirements. Fault-tolerant systems work under the assumption that they contain faults (e.g., made by humans while developing or using systems, or caused by aging hardware), and aim to provide specified services in spite of faults being present. Fault tolerance depends on redundancy, fault detection, and recovery. Many fault-tolerant systems are complex because of redundancy, re-configurability, and various interactions between their components. This complexity has a strong impact on system architecture, as fault occurrences have to be taken into account from the earliest design stages on of systems required to be dependable. Therefore, architecture design is a crucial aspect for fault-tolerant systems, as indicated by well-known safety mechanisms such as masking, dynamic redundancy, and design diversity (e.g., N-version programming, recovery blocks) [8].

Component-Based Software Engineering (CBSE) focuses on producing software systems by composing prefabricated components, which can improve the

productivity and quality of target systems. In recent years, the emphasis of research on and practice in CBSE has changed from functional aspects to non-functional ones. Particularly, dependability of component-based systems is considered as one of the most crucial non-functional properties in CBSE. To cope with complexity and fault tolerance requirements, it may be beneficial to combine in their development process well-established fault tolerance techniques with component techniques, as well as to employ the Unified Modeling Language (UML) [10] to describe component-based software architecture models providing fault tolerance.

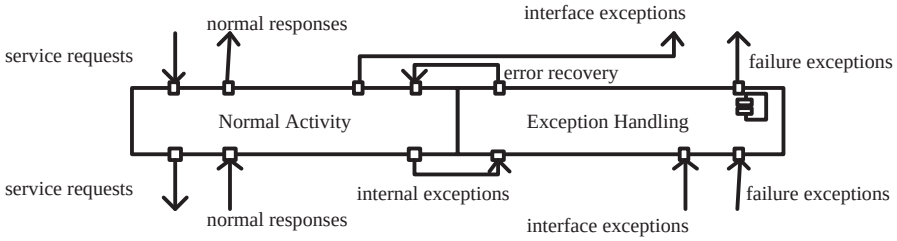
In a component-based system, the software architecture specification captures system structure, by identifying architectural components and connectors, and required system behaviour, by specifying how components and connectors are intended to interact. It is desirable that all components show only their normal behaviours. However, certain abnormal behaviours exist when some unexpected condition occur. In CBSE, the abnormal behaviours of components are usually represented by a set of exceptions defined by the component developers. It is impossible to eliminate all abnormal behaviours, but one can reduce them, and handle them properly after their occurrences. The main goal of this paper is to design a comprehensive architecture to develop fault-tolerant systems from components. By embedding fault tolerance mechanisms into an integration architecture, normal and exceptional behaviours of system components are specified.

The body of this paper is organised as follows. Section 2 briefly introduces an idealised fault-tolerant component meeting the architecture style C2 as defined in [7]. Section 3 describes the integration of idealised fault-tolerant components and the C2 architecture. Section 4 presents the main activities of designing fault-tolerant systems out of components.

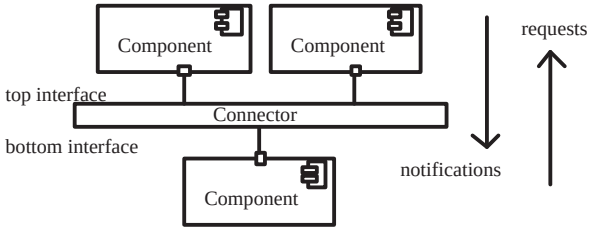
2 An Idealised Fault-tolerant Component and C2 Architecture Style

An idealised fault-tolerant component [5] is a structuring concept for coherent provision of fault tolerance in a system as shown in Fig. 1(a). It includes both normal and abnormal responses in the interface between interacting components. Idealised fault-tolerant components communicate through request or response messages, only. On receiving a service request, an idealised component will react with a normal response if the request is successfully processed, and with an external exception, otherwise. Such an external exception may be due to an invalid service request, in which case it is called interface exception, or due to a failure in processing a valid request, in which case it is called failure exception. Internal exceptions are associated with errors detected within a component that may be corrected, allowing the operation to be completed successfully; otherwise, they are propagated as external exceptions.

C2 is a software architecture style for systems with intensive user interfacing. The style is component-based, and supports large-grain re-use and flexible



a) Idealised fault-tolerance component



b) C2 architecture style

Fig. 1. Idealised fault tolerance component and C2 architecture style

system composition, emphasising weak bindings between components [7]. A C2 architecture as shown in Fig. 1(b) consists of software components and connectors, which transmit messages between components. Components maintain state, perform operations, and exchange messages with other components via two interfaces (named top and bottom). Each interface consists of a set of messages that may be sent or received. Intercomponent messages are classified into two types, viz., requests to a component to perform an operation, and notifications that a given component has performed an operation or changed state. In the C2 architectural style, both components and connectors have a top and a bottom interface. Systems are composed in a layered style, where a component's top interface may be connected to the bottom interface of a connector, and its bottom interface may be connected to the top interface of another connector. Each side of a connector may be connected to any number of components or connectors.

3 Integration of Idealised Components into C2

Guerra et al. [3] introduced the concept of the "idealised C2 Component" (iC2C), depicted in Fig. 2, as an equivalent, in structure and behaviour, to the idealised fault-tolerant component. The latter's purpose is to structure the architectures of component-based software systems compliant with the C2 architectural style. Service requests and normal responses of an idealised fault-tolerant component

are mapped as requests and notifications in the C2 architectural style. Interfaces and failure exceptions of an idealised fault-tolerant component are considered subtypes of notifications. An iC2C is composed of five elements: NormalActivity and AbnormalActivity components, and iC2C top, iC2C internal, and iC2C bottom connectors.

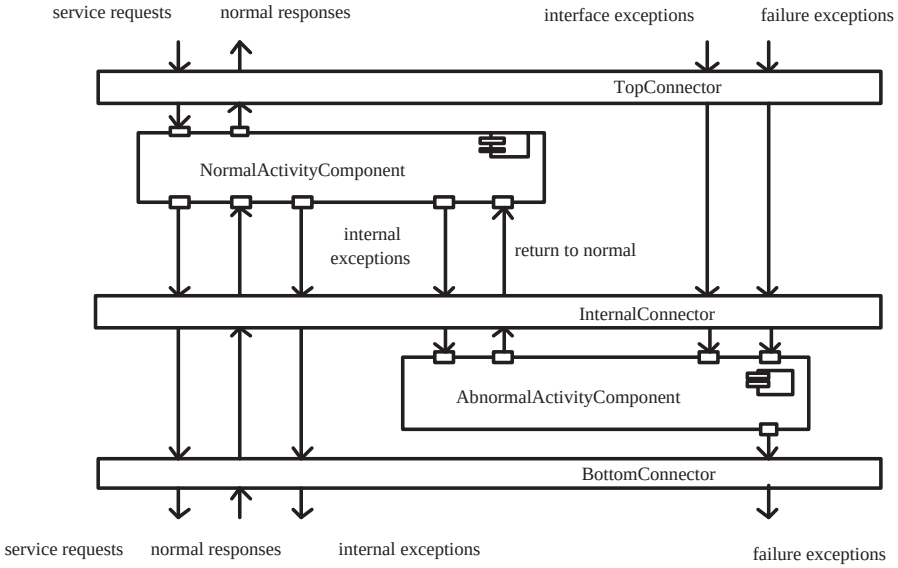


Fig. 2. An idealised C2 component

The NormalActivity component processes service requests and answers them through notifications. It implements the normal behaviour, responsible for error detection during normal operation, and the signaling of interface and internal exceptions. The AbnormalActivity component is responsible for the exception handlers (error recovery) of the iC2C, and the signaling of failure exceptions. While an iC2C is in its normal state, the AbnormalActivity component remains inactive. When an exceptional condition is detected, it is activated to handle the exception. In case the exception is successfully handled, the iC2C returns to its normal state and the NormalActivity component resumes processing. Otherwise, a failure exception is signaled to components in lower layers of the architecture, which become responsible for handling it.

The iC2C top connector encapsulates the interaction between the iC2C and components located in upper levels of an architecture. It is responsible to guarantee that service requests sent by the NormalActivity and AbnormalActivity components to other components located in upper levels of the architecture are processed synchronously, and that response notifications reach the intended des-

tinations. The iC2C top connector also performs domain translation, converting incoming notifications to a format the iC2C understands, and outgoing requests to a format the application understands. The iC2C internal connector is responsible for message routing inside the iC2C. The destination of a message sent by the internal elements of the iC2C depends on its type, and whether the iC2C is in a normal or abnormal state. The iC2C's bottom connector connects it with the lower components of a C2 configuration, and serialises the requests received. Once a request is accepted, this connector queues new requests received until completion of the first one. When a request is completed, a notification is returned, which may be a normal response, an interface exception, or a failure exception.

4 Extending the iC2C Architecture by Fault Tolerance Mechanisms

The proposed architecture is constructed by adding fault tolerance mechanisms to normal- and abnormal-activity components as shown in Fig. 3. The fault tolerance mechanisms are responsible for detecting errors or suspicious activities, and for executing appropriate recoveries whenever possible. The incorporation of fault tolerance mechanisms into the iC2C architecture requires to re-define normal and abnormal activities.

4.1 Definition of the NormalActivity Component

In the iC2C architecture, the NormalActivity component encapsulates a desired functionality (normal activity). It may represent both a single component and a configuration established through connectors. To embed error detection mechanisms into the NormalActivity component, the concept of protective wrappers, known to be the most general approach to develop fault-tolerant systems based on COTS components, is employed. Component wrapping is a well-known structuring technique, and a cost-effective solution to many problems in component-based software development [2].

In our extension, a set of detectors is wrapped as iC2C connectors connecting the normal activity component, which is viewed as redundant software, and responsible for (1) detecting exceptional conditions anticipated by the specification and signalling them by raising exceptions in the provided interface of the exception component, and (2) signalling other exceptional conditions specific to the implementation of the component, by raising exceptions. This set consists of well-defined error detectors concerned with special purposes such as "Fail-Stop Processor", "Acknowledgment, "I Am Alive" — "Are You Alive" [6]. When a constraint violation is detected, the detector sends an exception notification to the AbnormalActivity component. In addition to error detectors embedded in the NormalActivity component, an exception component is required to specify the raising of local and co-operating exceptions, i.e., signalling of an appropriate

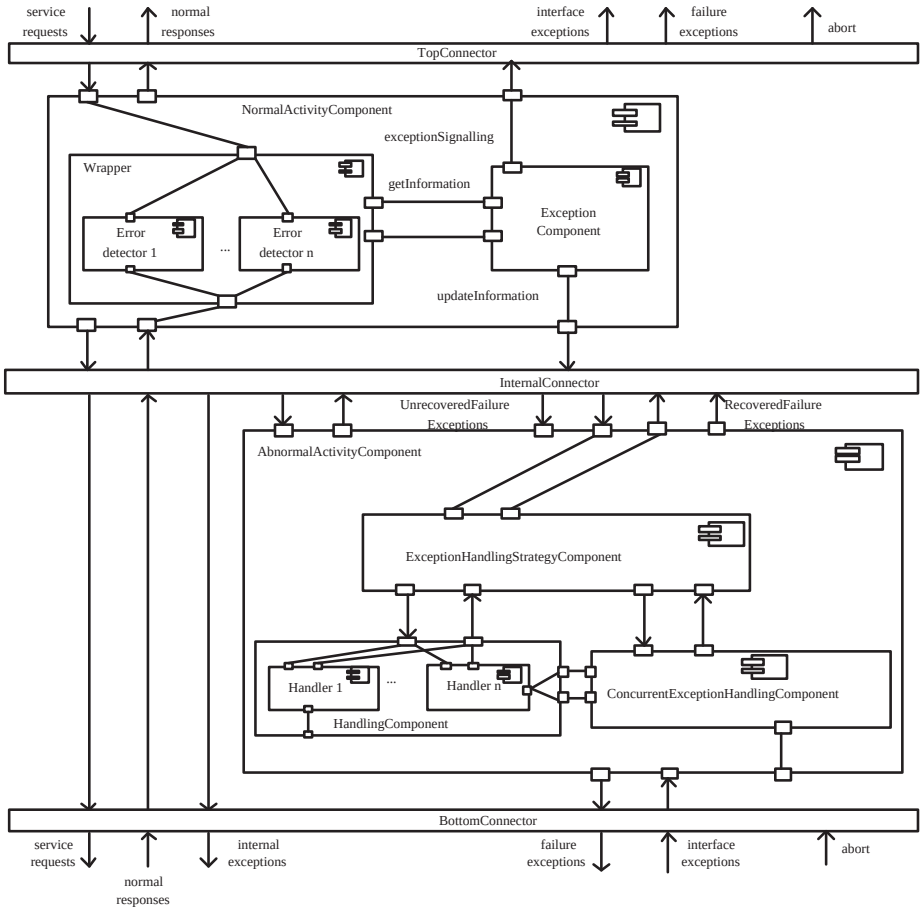


Fig. 3. Extending the iC2C architecture by fault tolerance mechanisms

exception. Hence, an exception component is designed working as an extra information holder, and keeping information about application exceptions which are used by the other components. In other words, the exception component interacts with the handler, concurrency and strategy component located in the AbnormalActivity component in order to obtain and update information about exception occurrences and handling. The exception component should have the required interface for getting information and the one for updating information. The former allows the application and other components to obtain information about exception occurrences, whereas the latter allows its clients to update information about exceptions.

4.2 Definition of the AbnormalActivity Component

Raising an exception results in an interruption of a component's normal activity, followed by a search for an appropriate exception handler to deal with the exception signalled. Therefore, the AbnormalActivity component can be defined as an exception handling mechanism as shown in Fig. 3 aiming for error recovery. When an exception is raised, the exception handling mechanism begins to work. It should be able to find and activate an exception handler to recover from errors, and to put the system back into a coherent state.

Abnormal behaviour is not restricted to failures of a single component, but also associated with invalid interactions between two or more components, and with combinations of component failures. An intra-component exception is raised by an individual component. The strategy to deal with it is based on local exception handlers, which are associated to a component's implementation. Their main responsibility is to cope with anticipated exceptions, which are more related to the component's application domain, and are declared in its required interfaces, i.e., they are responsible to handle the external exceptions of the component's required interfaces and the internal exceptions raised by the component's implementation. When possible, the handlers should implement forward error recovery to mask the exceptions.

Inter-component exceptions are raised by component configurations. The strategy to deal with them has to consider the integration of pre-existing components into a new configuration, and is based on co-operating exception handlers. These are associated with hierarchical structure, and deal with exceptions that could not be handled within the single components. The handlers are responsible for providing error recovery and masking by means of redundant components in the configuration, and resolving failure semantics mismatches between server components and their clients. They should be capable of dealing with all exceptions signalled by a server component. Upon receiving an exception from a server component, the handlers should try to mask it invoking the same operation on a redundant (replicated or diversely designed [1]) server component, in case it is available. The handlers are also responsible for translating unmasked exceptions from the server component's domain to the client component's domain, before propagating them to client components. Exceptions requiring no further adaptation are automatically propagated. When direct propagation is not possible, a new exception is created wrapping the unmasked exceptions raised by the server component.

As there are intra- and inter-component exceptions, the architecture is designed to consist (see Fig. 3) of

- HandlerComponent (HC),
- ExceptionHandlingStrategyComponent (EHSC), and
- ConcurrentExceptionHandlingComponent (CEHC).

The *ExceptionHandlingStrategyComponent* is responsible to locate the different handlers required to resolve an exception. It implements services related to the strategy for exception handling. Hence, its responsibilities are deviation of

control flow and search for handlers. It plays a central rôle in the architecture, and interacts with all other components. From ExceptionComponent it receives information about an exception occurrence while searching for its corresponding handler. Its provided interface provides the service for handler search.

The handling and propagation of an exception depends on whether it is an intra-component exception or an inter-component one. For the former, the handling is limited within the component's AbnormalActivity. If the exception cannot be handled, then it is propagated. A handler may also explicitly re-signal the exception to a component in a layer of the architecture. An inter-component exception raised on receiving a service request by a component is not handled there, since it does not indicate that the component is faulty. Therefore, the exception is propagated to the client component which issued the request. The client component handles the exception in the same manner as an intra-component one, because it possibly indicates a fault within the client component. The strategy to search for handlers of intra- and inter-component exceptions can be formulated in pseudo-code as follows:

```
if exception is propagated from intra-component then
  if local handler exists then
     call local handler;
     if not handled then
        go to a higher (action) level handling;
     end if;
  else
     go to a higher (action) level handling;
  end if;
  default handler;
else
    error detection in client component;
    if exception is raised (error has been found) then
        if local handler exists then
          call local handler;
          if not handled then
            go to a higher (action) level handling;
          end if;
        else go to a higher (action) level handling;
        end if;
        default handler
     end if;
end if;
```

The *HandlerComponent* is constructed with a set of handlers to cope with abnormal activities, i.e., it is responsible for fault masking and recovering. After a handler is found, EHSC asks the HandlerComponent to invoke the exception handler. According to the hierarchical structure, handlers may be associated with a component, a connector, or a configuration as well as with exceptions themselves (default handlers). A default handler is executed when there is not a more specific handler in an application. The HandlerComponent has a required interface which allows the EHSC to invoke a handler when the appropriate handler has been found. If an exception reaches the lowest level of an architecture, the handler for the entire system should be executed.

The *ConcurrentExceptionHandlingComponent* is designed for concurrent cooperative actions which use the services provided by the exception handling strategy in order to carry out the strategy for concurrent exception handling.

When co-operating exceptions are raised during an action, the exception resolution is accomplished by this component. It has a provided interface which is accessible by applications to create concurrent co-operative actions.

To design co-operative component activities using nested atomic actions, it is described how each individual component is involved in such activities, and co-operative exception handling for all participants in each action is developed. We employ the Coordinated Atomic (CA) action scheme [9] in which components take part, by defining a group as an action, and participants are components. Then, only co-ordinated recovery needs to be activated within the participant tasks of a group. This obviously restricts system design, but enables to regard each group as a recovery region, and to attach fault tolerance activities to each group participant.

Each group has participants which may be activated by some external activities, e.g., tasks, and which co-operate within the group's scope. Participants execute object methods that should have been designed to work co-operatively by means of shared objects. Participants may enter asynchronously into a group activity, but should exit in a synchronised way. Each group participant has a set of exception handlers that are designed to recover the group co-operatively from eventual errors. If any suitable handler has not been defined at least in one of the group participants, an "abort exception" is raised and, then, the group activity must be undone (backward error recovery), and such an exception must be signaled to the enclosing group. If the backward error recovery is not executed successfully within the group, a "failure exception" is signalled to the enclosing group.

Exceptions can be raised by participants during an action. Some of them can be handled internally by a local handler attached to the participant that raised an exception. If an exception occurrence is not handled internally by a participant, then it should be handled co-operatively by all action participants. A set of co-operating exceptions is associated with each action. Each participant has a set of handlers for (all or part of) these exceptions. Participants are synchronised, and probably different handlers for the same exception have to be invoked in the different participants. These handlers are executed concurrently, and co-operate in handling the exception in a co-ordinated way. Therefore, we employ local handling at the level of individual components. If local handling does not succeed, an exception is propagated to the level of an action in which this component is involved to be handled co-operatively. When action-level handling is not possible, an exception is propagated to the containing action.

The newly defined AbnormalActivity component encapsulates the exception handling mechanisms (intra-component and inter-component exception handling). After implementation, it requires an interface which should have the function handleException(). If an exception is successfully handled, handleException() returns a message which is sent to the NormalActivity component. Then, processing is resumed. Otherwise, an exception is raised in the body of handleException().

5 Conclusion

With the concept of the extended iC2C we aim to add fault tolerance mechanisms to the NormalActivity and AbnormalActivity components of an iC2C. In our approach, error detector and exception components are encapsulated into the NormalActivity component, with error detectors contained by a wrapper. These detectors are responsible to verify that messages do not violate the acceptable behaviour constraints specified for a system. The exception handling mechanism is embedded into the AbnormalActivity component.

The advantage of integrating fault tolerance techniques into the process of designing and developing computer control systems required to be dependable is that appropriate fault tolerance techniques can be selected from a set of mechanisms provided and customised according to application characteristics. This approach will enhance the safety of control systems. Employing its built-in extension mechanisms, UML can be extended to suit dependable applications with respect to aspects such as error detection, error recovery, or configuration of redundancy measures. Thus, for each aspect to be modeled, the most expressive techniques can be selected by the user. Furthermore, UML and the here specified extensions constitute an effective environment to design dependable computer control systems in a comprehensive way taking fault tolerance into account throughout the entire development process.

References

1. Anderson T. and Lee P. A.: *Fault Tolerance: Principles and Practice,* Dependable Computing and Fault Tolerant Systems, Vol 3. Springer-Verlag, 1990.
2. Anderson T., Randell B. and Romanovsky A.: *Wrapping the Future,* Proc. 18th IFIP World Computer Congress, pp. 165–173, 2004.
3. Guerra P. A. de C., Rubira C. M. F. and de Lemos R.: *An Idealized Fault-Tolerant Architectural Component,* in *Architecting Dependable Systems*, pp. 21–41. LNCS, Springer-Verlag, 2003.
4. Laprie J.-C.: *Dependability: Basic Concepts and Terminology,* Proc. 25th IEEE Intl. Symp. on Fault-Tolerant Computing, pp. 42–54. IEEE Computer Society Press, 1995.
5. Lee A. and Anderson T.: *Fault Tolerance: Principles and Practice,* Springer-Verlag, 1990.
6. Saridakis T.: *A System of Patterns for Fault Tolerance,* Proc. EuroPLoP.
7. Taylor R. N., Medvidovic N., Anderson K. M., Whitehead E. J., Robbins J. E., Nies K. A., Oreizy P. D. and Dubrow L. A.: *Component- and message-based architectural style for GUI software,* IEEE Trans. on Software Engineering, 22(6): 390–406, 1996.
8. Torres-Pomales W.: *Software Fault Tolerance: A tutorial,* NASA/TM-2000-210616, 2000.
9. Xu J., Randell B., Romanovsky A. B., Rubira C. M. F., Stroud R. J. and Wu Z.: *Fault tolerance in concurrent object-oriented software through coordinated error recovery,* Proc. Symp. on Fault-Tolerant Computing, pp. 499–508, 1995.
10. Unified Modeling Language: Superstructure. *OMG document ptc/2003-08-02*, 2003.

Proceedings of the International Multiconference on Computer Science and Information Technology pp. 923–932
ISSN 1896-7094
© 2007 PIPS

Using Preemption ForDependable Urban Vehicle Traffic

Tiberiu Letia, Sergiu Barbu and Florin Dinga

Technical University of Cluj-Napoca
Dept. of Automation
15 C. Daicoviciu St.,
400020 Cluj-Napoca, Romania

Abstract. A new approach for design and implementation of urban vehicle control system is proposed. The vehicle streams on lanes are considered similar with the streams of instructions in multitask programs. Real-time scheduling algorithms are used to allocate the green lights to phases. An adaptive component is used to calculate new vehicle flow parameters when a failure appears as a consequence of an accident. The real-time schedulers use the parameters to obtain new feasible resource allocations.

1 Introduction

Urban vehicle traffic control is one of the most complex problems due to the large number of variables or parameters involved and their unexpected variations. There are controlled inputs (by the traffic lights) and uncontrolled inputs. The flow of the vehicle stream can be measured by appropriate detectors (or sensors). Vehicle presence can be detected and occupancy can be measured. The unmeasured inputs and outputs introduce non deterministic factors. The flow splits can be measured or estimated but cannot be controlled in an acceptable manner [8].

Better traffic control leads to improved safety for all road users, shorter travelling times through the controlled part of the traffic system and it also reduces negative environmental impact [11]. Therefore, vehicle traffic control systems have to face: variable inputs (part of them controllable) represented by the vehicle flows entering the systems, variable transfer splits of the flows on the intersections and accidents that significantly modify the parameters. These require the achieving of adaptive control systems that take into account the variable ratio of the rates with the aim to maintain a high throughput and to reduce travelling time. The control system gets information about traffic from the detectors and controls it using the traffic lights.

2 Approaches of Urban Vehicle Traffic Control

Urban vehicle traffic control can be actuated or non-actuated. Control can be non-coordinated (implemented for isolated intersections) or coordinated. The

actuated urban vehicle traffic control can be performed on the following approaches: controlling the flows (i.e. the volume control), density control, queue length control, singular event reaction, phase time extension until a minimum gap out, and phase time extension until timeout.

The solution for vehicle traffic control problem contains: the intersection cycle durations (if cycles are chosen), the durations of phases (the split of the cycle) and their periods if there are no cycles, the order of phases (of each intersection) or their priorities, and the offsets (between intersections or phases).

Papageorgiou, et al. present some methods for local, centralized or distributed control of vehicle urban traffic [14]. Bazan presents a coordination method based on multiagent system that uses game theory to get the crossing times and the synchronization of intersection signalling [3]. The negotiation method can also be used [8]. Intelligent vehicle traffic management can be used, too [6]. Non artificial intelligent coordination methods are often based on optimization [15].

Kutil et al. use a model of a simple traffic intersection where the state variables represent the queue length and the average waiting times in the queues to obtain a fair traffic control [7]. They use the balancing waiting times to obtain a linear quadratic regulator and a nonlinear model predictive controller.

Liu and Tate propose an intelligent adaptation speed system that uses in-vehicle electronic devices to enable the speed of vehicle to be regulated automatically [12]. This offers a flexible method for speed management and control in urban area. Avella et al. present a method to solve the shortest path problem under resource constraints for vehicle fleet on a road net [1].

The majority of approaches (using deterministic or heuristic methods) consider the vehicle traffic system as working under probable (often previously measured and statistically expressed) conditions. But as a matter of fact, the demand (required green light) times and the granted durations (of neighbour intersections green lights) can vary considerably from instance to instance and are not known completely at the moment of the decision making. Even when these are known, the proposed methods are based on the estimated time demands of the traffic and on the probable values (splits) of the rates.

3 Preemption Based Control of Urban Vehicle Traffic

Unlike the previous ones, another approach is based on the worst case behaviour of the system. That means the usage in design of the highest accepted demands and the highest requirements relative to the timelines instead of the corresponding probable values.

The usage of real-time scheduling methods to signal control of the traffic lights in urban vehicle traffic is introduced in [9]. Vehicle flows on the lanes associated to an intersection phase are considered similar with the stream of instructions (of a task) that has to be processed by a processor. Each vehicle stream has its own period (that can be different from the periods of other phases of the same intersection) and duration to cross the intersection.

Figure 1 represents the vehicle streams of the traffic system.

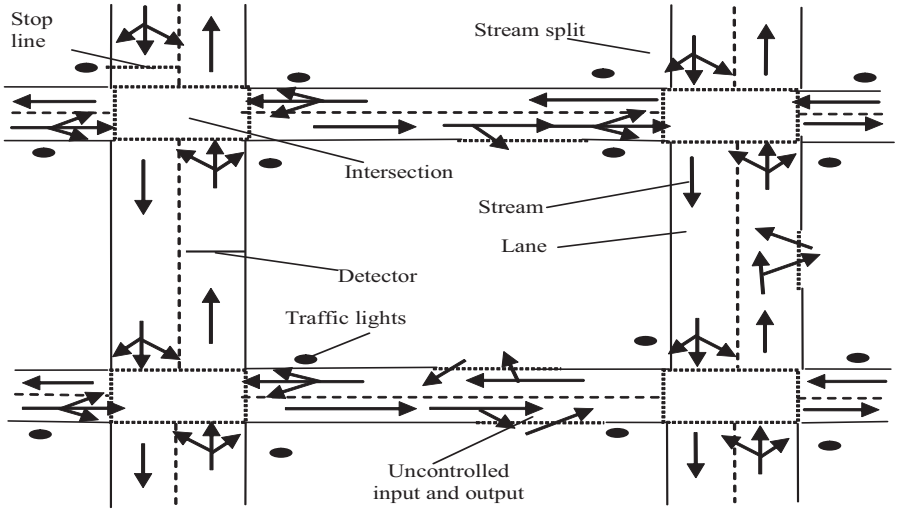


Fig. 1. Example of urban vehicle traffic net.

Lu et al. present a feedback control real-time scheduling framework for adaptative real-time systems [12]. They use feedback control theory to design algorithms that satisfy the transient and steady state performance of real-time systems.

One of the main problems of urban vehicle traffic control has to solve the scheduling of resources representing the places used by vehicles on lanes or intersections (most critical resources). The scheduling algorithms can be categorized as static or dynamic. In static scheduling algorithms have complete knowledge about stream set and its constraints (such as deadlines, crossing times, precedence constraints and future demand times). In dynamic scheduling the algorithms have no complete knowledge about stream set or its timing constraints. The dynamic scheduling algorithms can be divided into algorithms that work in sufficient resource environment and those that work in insufficient resource environment. The traffic congestion is an example when the system reaches a state in which, at least for the moment, the available resources are insufficient.

The scheduling methods can be applied to the design and control of the urban vehicle traffic system such that it can fulfil some specified requirements. Figure 2 shows the temporal relations between the demanded moment of time (dt) and green light durations of two consecutive intersections i-1 and i. Each of them should be opened within a specified period (T). The offset (O) between them also has to be implemented. The jitter (J) appears due to variation of vehicle speed or opening of the (linked) phase from the previous intersection. The opening of a phase before the demand moments of time (dt) is useless. If the end of the

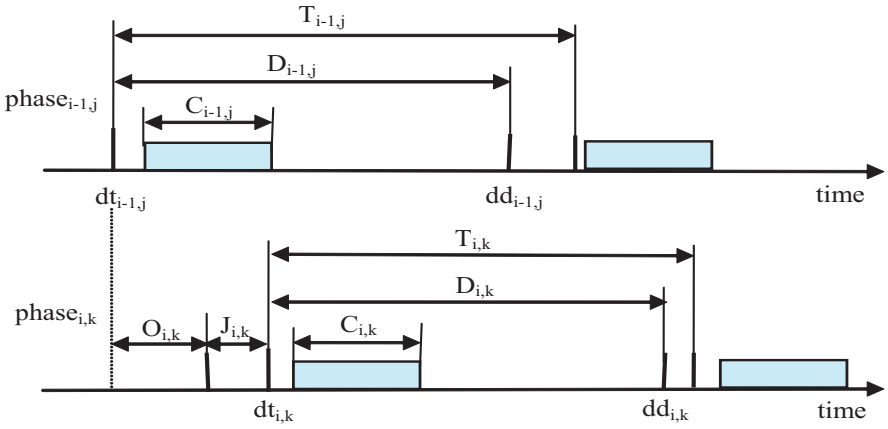


Fig. 2. Temporal relations between phases of two consecutive intersections.

crossing time (C) is after the deadline (dd) this can lead to congestion. When a phase is opened at the dt (just when the vehicle stream arrives) because the vehicles have the desired speed (they are moving) the throughput is higher. This leads to a higher usage of the intersection (representing a critical resource). If the demanded durations are higher than the granted durations for some periods of time, this leads to congestions, also because lane capacities are exceeded.

Figure 3 describes the UML state machine of a phase from an intersection. The switching of the phases is performed by the implementation of the real-time scheduler by updater (Figure 4) and dispatcher (Figure 5). They can implement different types of real-time schedulers used to control the urban vehicle traffic systems. The updater calculates the waiting times (w) and adds the phases (using add(phase) method) to the ready queue when they expire. It also calculates the number of requests (req) of each phase that are not honored.

The real-time schedulers use the following parameters:

- the crossing time (C) of a vehicle stream during a cycle (period) through an intersection
- the period of a phase (T)
- the deadline (D) of the openness of a phase at a new demand
- the offset of the phase (O)
- the phase local priority and
- the global priority.

In figure 5 are denoted by cp the current phase, fp the first phase of the ready queue (readyQ), update() the recalculation of the ready queue according to the scheduling algorithm, remove(x) the removing of the phase x from the ready queue, red(x), yellow(x) and green(x) the application of the corresponding color to the phase x. For jitter implementation the diagram from the figure 5 has to be extended with a new state and transitions to catch the demand events.

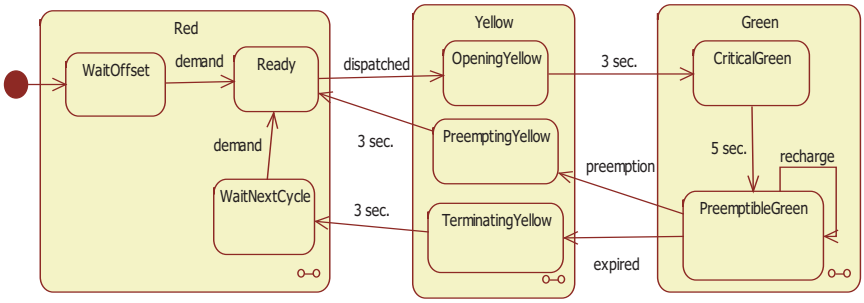


Fig. 3. UML state machine of an intersection phase.

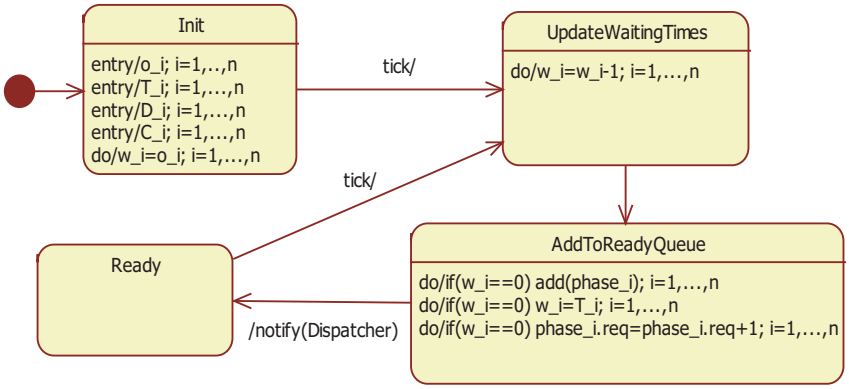


Fig. 4. UML state machine of Updater.

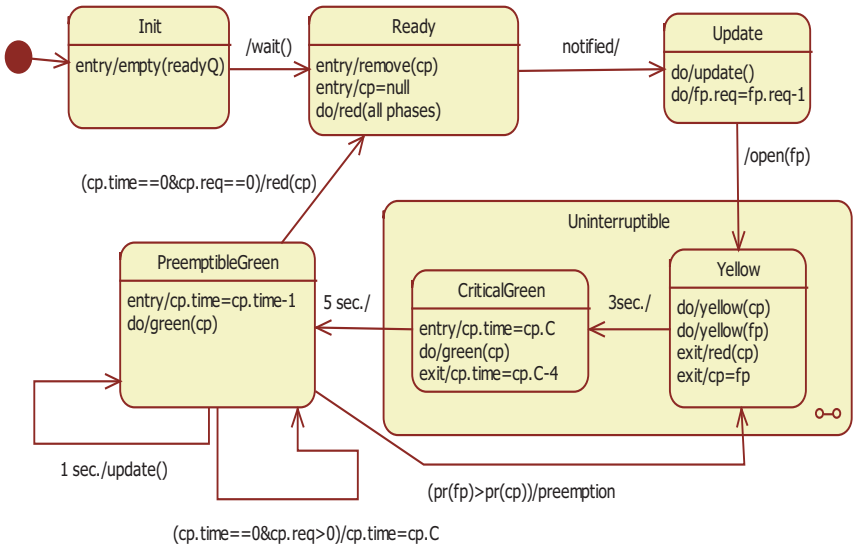


Fig. 5. Time driven dispatcher UML state machine.

Fidge describes the most largely used real-time scheduling methods and their tests [4]. Between them are: RMS (rate monotonic scheduling), EDF (earliest deadline first) and SPS (static priority scheduling).

Using such scheduling methods for the network of intersections composing urban vehicle traffic system, a system that fulfils some specified timing constrains is obtained. The crossing times of neighbour intersections should be correlated.

4 Dependable Urban Vehicle Traffic System

Taking into account the article of Avizienis et al. ([2]), the dependability of an urban vehicle traffic system is given by the following attributes: availability (readiness for usage—i. e. the acceptance of vehicle entrance in the traffic), reliability (continuity of service—the moving of vehicles without unnecessary unbounded stops), safety (absence of catastrophic consequence on the user), confidentiality (absence of unauthorized disclosure of information, usually less important for vehicle traffic system in regular usage), integrity (absence of improper system alteration) and maintainability (ability to undergo repairs and evolutions). The failures of urban vehicle traffic system are:

- failures of the control system caused by bad control decisions, message transmissions or signalling of events or states
- failures of traffic systems (distributed controlled process) caused by exceeding of specified demands or driver's faults that lead to accidents

The vehicles that remain inside an intersection after their phase looses the right of crossing (green lights), because they cannot enter the output lanes, involve congestions [10]. Their number multiplied by a coefficient dependent on the structure of the intersection can be used as a measure of the current degree of congestion. The average values of the congestion degrees of all intersections of the traffic system provide information about current congestion degree of the entire system. The evolution of the average congestion degree can be used to evaluate the traffic system behaviour under the implemented control algorithms.

A fault of a driver can lead to a traffic failure (an accident) that blocks the vehicle streams of a street. Following this the remained vehicle streams are modified (some of them split) such that the traffic flows avoid the blocked lanes. As a consequence, new parameters should be provided to traffic lights controller (based on scheduling algorithms) to adapt to the modified environment and having the goal to avoid the congestion of the system. The congestion of an intersection can lead to the congestion of the entire traffic system if it is highly loaded with vehicles.

The on-line control uses the estimation of flows based on the measure of flow inputs and the flow splits. Using the real-time scheduling algorithms, the control system estimates if the necessary crossing times lead to a feasible solution (that fulfills the timing constraints). The local controllers receive the parameters (offset, period, green duration and priority) of each phase and implement the on-line schedulers that send the signals to the traffic lights.

The centralized adaptive algorithm activated at the detection of a failure is:

for *all intersections*

Estimate or receive from local controllers the splits of the rates.

for *all intersections*

for *all phases*
Calculate the necessary crossing time.
while (*feasibility of scheduling is not fulfilled*)
Reduce the crossing time of each phase proportionally with its global priority, but not lower than a specified limit.

Communicate to local controllers the new scheduling parameters.
end

5 Testing and Evaluation

The performance evaluation of urban vehicle traffic system can be developed from the concepts of real-time systems given by [5]. The evaluation can be based on: qualitative binary criteria, qualitative gradual criteria and quantitative criteria.

1. Qualitative binary criteria are: timelines (the ability to meet some specified deadlines), no unbounded delay nor arbitrarily long crossing (or travelling) times, functional correctness, permanent readiness, all applicable physical constraints and congestion prevented

2. Qualitative gradual criteria are: safety, reliability, availability, congestion, simplicity, robustness, fault tolerance, graceful degradation on the occurring of undesired events, portability (of the control system), flexibility (to change of the vehicle traffic structure) and extensibility (of the control system with the development of the traffic net)
3. Quantitative criteria are: worst case travelling times on a specified path, worst case crossing times of an intersection, worst case duration to detect the failures and to correct the consequences (speed of adaptation), capacity (throughput) reserves and transfer capacities (throughput) of road net system or intersections

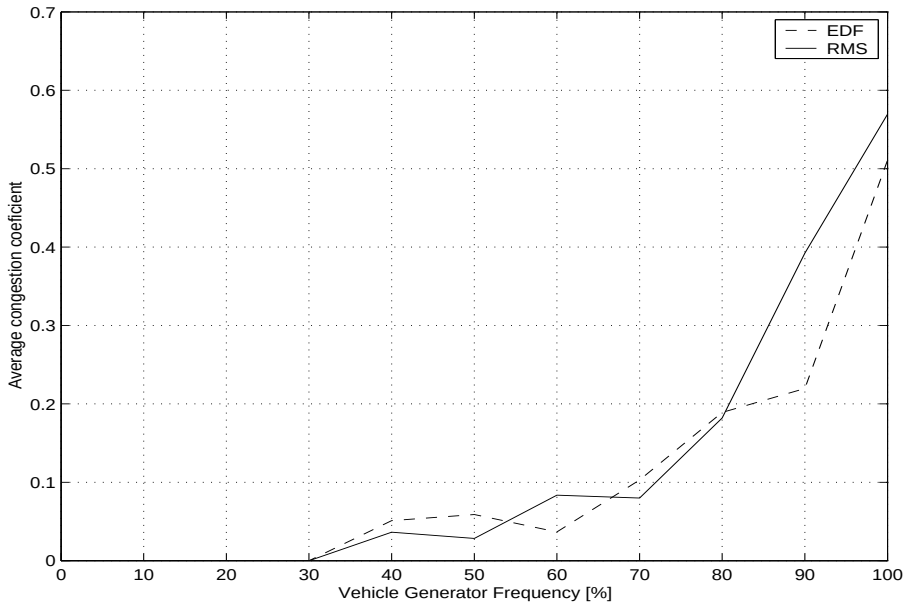


Fig. 6. System congestion degree without incidents.

To evaluate the proposed control method a special urban vehicle traffic simulator was built. It uses the microscopic method for simulation of vehicles on lanes and a macroscopic method to evaluate the number of vehicles inside intersections. The simulator is able the calculate the congestion degree of each intersection taking into account the number of vehicles that remain inside the intersection after the traffic lights of the lanes they leave change to the red light, as well as the intersection structure. The average intersection congestion degrees are used by the simulator to evaluate the entire vehicle traffic system congestion degree. Their values for different vehicle frequency inputs are shown in figures 6 and 7. The first mentioned figure shows the system congestion degree, under normal

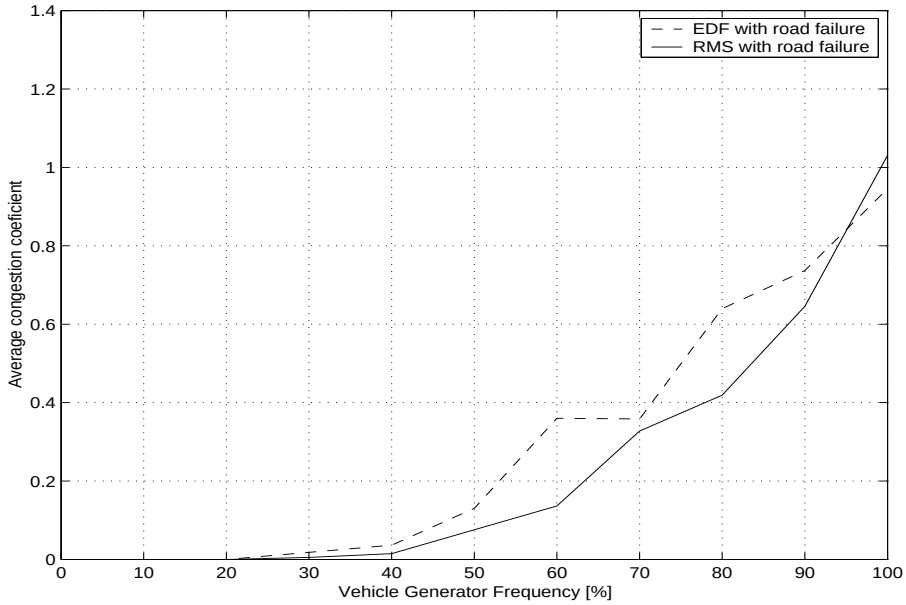


Fig. 7. System congestion degree with a traffic incident.

circumstances, using two different scheduling algorithms. Figure 7 represents the same system congestion degree when an incident appears that blocks a street.

6 Conclusions

The urban vehicle traffic dependability can be improved by specification, design of the control system and synthesis of the control algorithms. Observations of the urban vehicle traffic behaviour when failures appear are useful to obtain better specification of description and requirements.

Models that describe more closely the behaviour of drivers when failures appear, should be used to improve the requirements of the control system. The proposed method uses different periods for the phases of the same intersection. Usually, the periods of phases linked in a path should be correlated. Further studies should be developed to find the need (the effect) of modification of phase periods at the failure appearance.

The local priorities are used to schedule or analyse the feasibility of critical resources scheduling. The global priorities are used to determine the global behaviour of the traffic system. They serve to maintain the main features of the system when failures appear. The relations between global priorities and quantitative criteria (for traffic performance evaluation) are to be studied with the aim to improve dependability.

The performance evaluation results (from simulations) demonstrate that the proposed algorithms provide robust transient and steady state performance when

traffic volumes are at low values. At high volumes, without a throughput reserve the control system is not able to correct the effect of failures. The necessary reserve is dependent of the place of the failure and the current volumes of neighbour flows.

References

1. Avella P., Boccia M., Sforza A.: *Resource constrained shortest path problems in path planning for fleet management,* Journal of Mathematical Modelling Algorithms. **3** (2004) 1–17.
2. Avizienis A., Laprie J. C., Randel B., Landwehr C.: *Basic concepts and taxonomy of dependable and secure computing,* IEEE Trans. on Dependable and Secure Computing, **1** (2004) 11–33.
3. Bazzan A. L. C.: *A distributed approach for coordination of traffic signal agents,* Autonomous and Multi-Agent Systems. **10** (2005) 131–164.
4. Fidge C. J.: *Real-time schedulability tests for preemptive multitasking,* Real-Time Systems **14** (1998) 61–93.
5. Halang W., Gumzej R., Colnaric M., Druzovec M.: *Measuring the performance of real-time systems,* The International Journal of Time-Critical Systems **18** (2000) 59–68.
6. Kirschfink H., Hernandez J., Boero M.: *Intelligent traffic management models,* Proc. of ESIT (2000) 36–44.
7. Kutil M., Hanzalek Z. and Cervin A.: (2006) *Balancing the waiting times in a simple traffic intersection model,* Proc. of 11th IFAC Symposium on Control in Transportation Systems (2006) (to appear).
8. Letia T., Astilean A., Hulea M., Valean H.: *Flow price based vehicle traffic distributed control.* Proc. of the 2nd International Workshop on Discrete-Event System Design (2004) 91–96.
9. Letia T.: *Real-time approaches of urban vehicle traffic,* IEEE-TTTC International Conference on Automation, Quality and Testing, Robotics, **I** (2006) 346–350.
10. Letia T.: *Avoidance of urban vehicle traffic congestions,* Proc. of 11th IFAC Symposium on Control in Transportation Systems (2006) (to appear).
11. Lindley J. A.: *Urban freeway congestion: Quantification of the problem and effectiveness of potential solutions,* Institute of Transportation Engineers Journal **57(1)** (1989) 27–32.
12. Liu R., Tate J.: *Network effects of intelligent speed adaptation systems,* Transportation **31** (2004) 297–325.
13. Lu C., Stankovic J. A., Son S. H.: *Feedback control real-time scheduling: framework, modelling and algorithms,* Real-Time Systems **23** (2002) 85–126.
14. Papageorgiou M., Diakaki C., Dinopolou V., Kotsialos A., Wang Y.: *Review of road traffic control strategies,* Proceedings of the IEEE **91** No. 12 (2003) 2043–2067.
15. Porche I., Lafortune S.: *Dynamic traffic controll: decentralized and coordinated methods,* IEEE Conference on Intelligent Transportation Systems **9-12** (1997) 930–935.

Proceedings of the International Multiconference on
Computer Science and Information Technology pp. 933–937
ISSN 1896-7094
© 2007 PIPS

The modular approach of a real-time course

Shmuel Tyszberowicz

Tel-Aviv University and The Academic College Tel Aviv Yaffo

Abstract. This article describes some thoughts concerning the needed syllabus for a "Real-Time Systems" course for computer science students. The ideas described are based on several years of experience in teaching such a course both for graduate and advanced undergraduate (in the last year of studies) students at several academic institutes: Tel-Aviv University, the Open University of Israel, and the Academic College of Tel-Aviv Yaffo.

1 The problem

For over a decade I have been teaching undergraduate and graduate students at several academic institutes various "Reactive and Real-Time Systems" courses. *Reactive systems* are computer systems that continuously react to their physical environment, at a speed determined by the environment. *Real-Time Systems* (RTSs) have reactive behavior. An RTS involves control of one or more physical devices with essential timing requirements. The correctness of an RTS depends both on the time in which computations are performed as well as the logical correctness of the results. Adding the term reactive system to the name is due to the fact that we mainly deal with the reactivity of the system.

The syllabus of the course, its rationale, and the kind of projects the students should do have already been described in [4, 6, 5]. Basically, both courses focus on the development of *provably correct* reactive real-time systems. They cover the full life-cycle of a system development: specification, design, implementation, and verification. The courses combine theoretical issues with practical implementation experience, trying to make things as tangible as possible. The courses have been planned such that no special purpose hardware is needed and all software tools used are freely available from various Internet sites and can be installed quite easily. This makes our course attractive to institutions and instructors for which purchasing and maintaining a special lab is not feasible due to budget, space, or time limitations (as in our case).

The approach we chose was to provide a set of predefined teaching topics, where the instructor selects which topics to teach, and how much time to devote to each of them. As a reference point for the set of topics one can use those suggested by Zalewski [9]. We have carefully studied the lecture topics, and put less emphasis on the knowledge units. Thus, for example, whereas we described analog and digital input/output, we removed topics such as buses, DSP, LAN. This was mainly done due to the fact that computer science students (at least in the above mentioned institutes) do not have the appropriate background, Thus, we actually eliminated the real-time hardware architecture knowledge unit. On the other hand, we added other units such as the synchronous

approach [1]. We also updated the content of others, e.g. the real-time development methodologies knowledge unit. Note that whereas the syllabus offered by Zalewski refers to undergraduate students, the topics can be adjusted to graduate students as well. Actually, we can use identical or at least very similar topics for both the undergraduate and graduate class, but with different content.

During the years I have been teaching those courses, I felt that, much more than in other courses I teach, the success of the real-time course depends on the background of the students. Background in this context refers both to courses already studied and to industry experience, i.e., developing projects that are larger than academic ones. This may have several reasons. Both the content of some courses and the list of mandatory courses is changing from time to time. This fact has directly influenced the content of the real-time course and sometimes also its quality. The academic background of the students may also have a great influence on the content of the course. For example, suppose that the students have already studied threads in Java (or that they have studied both Java and operating systems). If the real-time course requires programming, it is reasonable to emphasize embedded systems, since Java provides good interfaces. Thus it easier to create interfaces to the embedded hardware, and we can avoid low-level programming. On the other hand, Java is not the right language to deal with hard real-time systems, as it does not have the notion of interrupts and provides no means to specify timing constraints. Moreover, features such as garbage collection or polymorphism, make it difficult to predict the timing, a crucial demand for hard real-time systems. If, on the other hand they know C, hard real time systems can be dealt with.

The mandatory courses can then be used to decide what would be emphasized in the course. Of course, if a minimal number of prerequisites is needed in order to teach other topics, this should be no problem.

Another issue where the real-time systems course differs from other courses is its preparation. For most other running courses, when preparing the materials for a new semester, it seems that the basic topics have not changed, and what might be added are new topics, for example the additions to Java 1.5 as compared to Java 1.4. When looking for new assignments, I almost always end up with a problem that describes a controller of some embedded system, with virtual sensors and actuators. Virtual, since there is no budget to buy even Lego Mindstorm. I always envy teachers who could ask their students to drive satellites, etc (cf. the various proceedings of the Real-Time Educational Workshops), or at least drive a Lego train. And I always asked myself what examples will I use if there would be enough budget.

The tools we have used for our course have also caused a lot of extra work. For example, I am teaching the synchronous language Esterel. The free version from Esterel's website [10] is from 2000. In order to adjust it to changing environments, I have to prepare a file with the needed instructions and the files that are missing in newer environments. A tool that translates Esterel to SMV does work only on the previous Unix version of Esterel. The free academic version of Esterel [11] needs a new license every year.

2 The solution

All the reasons mentioned in Section 1 caused me to take a break from teaching this course. Nevertheless, since I knew that some day I will be asked to teach them again, I tried to see what can be changed, hoping that there is a better, at least easier, way to handle these issues.

It became clear to me that in order for the courses to be successful, and in order to make the preparation process smoother, each of the courses should be built in a way that will easily enable it to be adjusted to the class. It is not enough to have two separate courses, one for undergraduate students and the other for graduate ones; several other aspects need to be considered.

Creating a syllabus for a given class should be based on pre-defined, existing, teaching modules. This can be seen as a multi-dimensional matrix, where each cell contains the relevant teaching module. The first dimension, which also is the most important one, contains the various topics that should be taught.

Other possible dimensions could be the following (in no particular order):

- Who are the students: computer science or engineering students?
  There is definitely a difference in the background of the students, and it might be natural to have a more theoretical course for CS students, and a more practical one for engineering students.
- Graduate or undergraduate students?
  We can choose the very same topics for both classes, but both the level of formality and thus the difficulty will vary. For example, if the instructor decides that he would like to teach analysis of real-time systems, the following modules will be available:
  - The non formal approach, e.g. Hatley and Pirbahi [3] or Ward and Mellor [8] for undergraduate students.
  - The semi formal approach, e.g., Statecharts [2] as a drawing tool, using only an informal semantics, for undergraduate students and with coverage of the semantics for the graduate students.)
  - The formal approach, e.g., temporal logic [7], for graduate students.
- Theoretical or practical course?
  On the one extreme there are real time courses which only teach scheduling algorithms or formal methods for specification and verification of such systems. In a practical course the theory has to be implemented as well.
  Subtopics here could for example be which kind of scheduling algorithms to teach: EDF, rate monotonic, both?
- What dominating aspects of a system would we like to emphasize?
  Each system is dominated by a specific aspect, which influence the choice of the appropriate language.
  - Data flow dominated systems, such as a heat-control system or active noise reduction system. The preferred language would be Matlab/Simulink.
  - Hard-real-time systems (i.e., systems where the most important issue is timing), such as emergency braking system (in spite of the fact, that there is also substantial data-flow in such a system). The preferred languages are synchronous one, e.g., Esterel, Lustre, Signal [1] or synERJY.

  - Embedded systems, for example smart phones. Here the preferred language would be Java.
- What languages do the students already know?
  Whereas in the previous item the kind of the system determines the chosen language, here the idea is to chose the kind of systems based on the existing knowledge, as discussed above.
- Budget.
  This definitely may influence the course nature. With no budget, the course can be a theoretical one, or use free software. However, almost no large real life project can be implemented.
- Class size.
  In a larger class we can have large groups of students working on each project, for example.

Each cell in this multi dimensional matrix contains the appropriate study module. This matrix can also serve as a kind of road map, where the required knowledge for each module is provided. Each cell should also contain information regarding the time needed to teach the module. Hence, while preparing the course, the instructor will know if he or she can teach in the planned order and how much time it should take. This approach also enables cases where the instructors decide to choose only one topic. Thus, one of my colleagues is teaching a course which deals only with scheduling algorithms for hard real-time systems, whereas another is mainly teaching how to formally specify system properties and how to verify these properties using model checkers.

This modular approach makes it easy to add new consideration and hence new study items. I envision instructors adding more dimensions and the relevant modules for each dimension. First, the list of topics should be updated, then the factors that influence the content of each topic should be treated. Last, and maybe most important, modules should be built. The best would be something like a wiki based site, with all the relevant materials. This will include theory, papers, and presentations.

References

1. Nicolas Halbwachs. *Synchronous Programming of Reactive Systems*. Kluwer Academic, 1993.
2. David Harel and Michal Politi. *Modeling Reactive Systems with Statecharts: The Statemate Approach*. McGraw-Hill, 1998.
3. Derek Hatley and Imtiaz Pirbhai. *Strategies for Real-Time System Specification*. Dorset House Publishing, 1987.
4. Gilad Koren and Shmuel Tyszberowicz. Graduate course: Reactive and real-time systems. In *RTEW '97: Proceedings of the Second IEEE Real-Time Systems Education Workshop, Canada*, pages 96–103, USA, 1997. IEEE Computer Society.
5. Tal Lev-Ami and Shmuel S. Tyszberowicz. Reactive and real-time systems course: How to get the most out of it. *Real-Time Systems*, 25(2-3):231–253, 2003.
6. Ran Lotenberg and Shmuel Tyszberowicz. Student projects in reactive and real-time systems course. In *RTEW '98: Proceedings of the Third IEEE Real-Time Systems Education Workshop, Poland*, pages 57–62, USA, 1998. IEEE Computer Society.

7. Zohar Manna and Amir Pnueli. *The Temporal Logic of Reactive and Concurrent Systems*. Springer-Verlag, New York, 1992.
8. Paul Ward and Stephen Mellor. *Strategies for Real-Time System Specification*. Prentice Hall, 1985.
9. Janusz Zalewski. Real-time systems. undergraduated course syllabus, 1993.
10. Esterel home page. `http://www-sop.inria.fr/esterel.org/`.
11. Esterel Technologies. the Esterel academic program. `http://www.esterel-technologies.com/technology/academic-program/`.

Proceedings of the International Multiconference on
Computer Science and Information Technology, pp. 939 – 942
ISSN 1896-7094
© 2007 PIPS

A Model for Educating Real-Time Software Engineers On-Demand

Janusz Zalewski[1]

[1] Department of Computer Science
Florida Gulf Coast University
Ft. Myers, FL 33965-6565, USA
zalewski@fgcu.edu
http://www.fgcu.edu/zalewski/

Abstract. This paper discusses educating software engineers at the undergraduate level to prepare them for taking immediately available jobs in industry, in response to the industry demand in the area of real-time systems. It is focused on the innovativeness of organizational aspects of the educational process, involving its stakeholders, rather than on the curricular aspects, traditionally considered first.

1 Introduction and Background

In the era of prospective significant mismatch between the demand for engineers and the number of graduates who could fill all available positions and be well prepared for entering the job market, some unconventional steps have to be taken to reverse the negative trends. In other words, radically new models of engineering education have to be proposed, built and verified, which would address the concerns and issues currently considered detrimental to the engineering education. Since the author's expertise is from the domain of Real-Time Software Engineering, this discipline is used as a vehicle to convey new ideas, but it is believed that the essential concept is not limited to any single engineering discipline.

Briefly speaking, the author proposes "Education On-Demand", which would address several most significant aspects of the educational process and its interaction with stakeholders, and lead to producing graduates and educators, who would act as a "rapid response force" in meeting the immediate needs of employers. The three crucial aspects in this process are considered to be:

(1) *local awareness*, that is, enforcement of interactions among the local communities, such as the university, being the "producer" of graduates, high schools, being the "feed" of talent to universities, and industry, being the recipient of graduates;
(2) *global awareness*, that is, taking advantage of the fact that we all live in McLuhan's global village and both students and graduates, faculty and administration, should be prepared to interact with other stakeholders worldwide, for example, faculty teaching a course in the US from Europe, project being developed in a lab in Florida by a team of students located in four different countries around the globe, flexible curricula combining for-credit courses from best programs in the discipline around the world, etc., and

(3) *cross-disciplinary interaction*, which means not only blending traditionally different and disjoint engineering disciplines (a good example of which is mechatronics), but also reaching across the boundaries of engineering to such disciplines as psychology, biochemistry, economics, social and decision sciences, etc.

The author is actively pursuing all three aspects in building the proposed model in a variety of ways, for example:

(1) engaging local high school students and teachers by periodically visiting high schools with college projects' demos (such as remote wireless data acquisition, robots, modern web design); engaging industry leaders in presentations called "Distinguished Lecture Series";
(2) engaging faculty and industry worldwide in his web-based courses to address student questions; developing (under a government grant) a web-based real-time software engineering lab;
(3) initiating collaborations with faculty in traditionally non-engineering disciplines, by submitting joint papers and grant proposals to involve those disciplines in revitalizing computing education.

Since the space in this paper is very limited, we will discuss here sample student projects, which were brought to the classroom by the leading software companies from the area.

2 Sample Projects

To illustrate the scope of the educational activities, we discuss below a sample of multiple student projects that were conducted for the last couple of semesters in EMbedded Systems Programming and Senior Software Design courses.

2.1 Robotics

The two robots used, a small wheel bot and a robotic arm are shown in Figure 1. The experimental task was to apply alternative programming methods to improve performance.



Figure 1. Ridgesoft Intellibrain bot and UMI RTX Robotic Arm.

The smaller robot's interface was changed from serial cable to Bluetooth, which allowed remote control of behavior and full operation with sensors. The robotic arm's control algorithm was improve to allow time optimal motion control. More specific information is included in a report available upon request.

2.2 Sensor Networks

This project relied on configuring a sensor network from Crossbow, to read light sensor and other sensor data and make them available through a server to the Internet. A sample network configuration is shown in Figure 2.

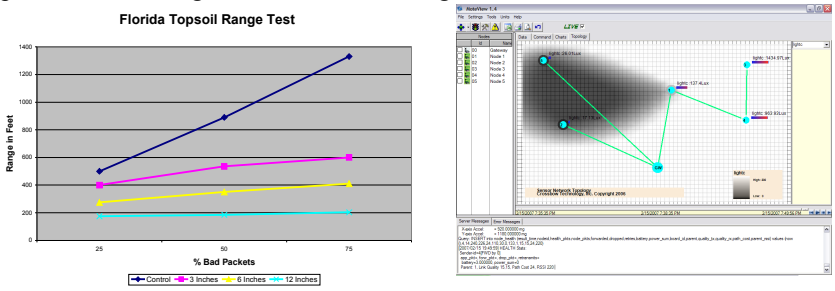


Figure 2. Sensor Network Topology Map Showing Operation of Light Sensing (left) and Sample results from the ZigBee Network Range Test (right).

Another project of that sort was conducted with the use of a ZigBee-based sensor network from MaxStream, and aimed at testing the range of the network for taking some soil measurement. Result sof the test for various types of Florida soil are shown in Figure 2. The distances vary from 0 to 1400 feet and the increase in packet loss is reflected on the diagram.

2.3 RFID Network

The final project involves a small RFID network to track paths of a certain fish species in the local river. A diagram of the server part is shown in Figure 3. The fish are implanted with RFID chips, which generate signals detected by the antenna.

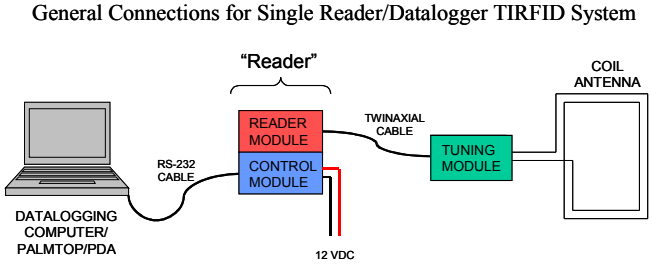


Figure 3. Illustration of the RIFD Fish Tracking System (Server Part)

3 Conclusion

The metaphor used in the title obviously comes from the "video on-demand." The model is based on presumption that there is a market for Software Engineers, just like

there is a market for videos, and there are customers who want to "retrieve" them on demand, making sure that the product meets customer's expectations regarding the functions and quality. Consequently, there should be an aggressive industry (understood as a societal mechanism) formed, which would "produce" such graduates, just like an industry has been formed to produce and deliver electronics goods, in particular, video on-demand. That way, the "On-Demand" aspect of engineering education will be met. Following this analogy further, the author believes that such mechanism, if sustained, would ultimately lead to making software engineers a commodity. If successful, this model can be extended to educate engineers in the majority of professions.

Acknowledgement

Work on this project was funded in part by the National Science Foundation grant No. DUE-0632729.

References

1. N. Skoro, J. Eren, *A Study on Robotics,* COP 4908 Course Project, Florida Gulf Coast University, April 2006.
2. J. Maxwell, *A Study and Implementation of A Crossbow Sensor Network,* CEN 4935 Course Project, Florida Gulf Coast University, April 2007.
3. T. Rodriguez, *PDA User Interface and Experiments with an RFID Fish Tracking System,* CEN 3213 Course Project, Florida Gulf Coast University, April 2007.

Proceedings of the International Multiconference on Computer Science and Information Technology, pp. 943 – 948
ISSN 1896-7094
© 2007 PIPS

ILERT - International Learning Environment for Real-Time Software-Intensive Control Systems

Andrew J. Kornecki [1] , Thomas B. Hilburn [1] , Wojciech Grega [2] , Miroslav Sveda [3] , Jean-Marc Thiriet [4]

[1]Embry Riddle Aeronautical University, Daytona Beach FL, USA
kornecka@erau.edu, hilburn@erau.edu
[2]AGH University of Science and Technology, Krakow, Poland
wgr@IA.AGH.edu.PL
[3]Brno University of Technology, Brno, Czech Republic
sveda@fit.vutbr.cz
[4]Laboratoire d'Automatique de Grenoble, Grenoble, France
jean-marc.thiriet@lag.ensieg.inpg.fr

Abstract. Due to heavily software-centric nature of modern reactive and time-critical systems, there is an increasing demand for efficient development of high quality Real-Time Software-Intensive Control systems (RSIC). The study discussed in this paper is focused on the creation of international curriculum framework centered on RSIC – this important aspect of the computer-system-control-software engineering education. The study explores the mechanism for involving students from multilingual geographically separated institutions in a coordinated educational experience. It exposes them to the problems, methods, solution techniques, infrastructure, technologies, regulatory issues, and tools in the domain of dependable real-time safety-critical software-intensive control systems. The ultimate objective is the creation of a model RSIC curriculum, which can be used by engineering schools both in the USA and the EU.

1 Introduction

There is an increasing importance and demand for efficient development of high quality Real-Time Software-Intensive Control systems (RSIC). Such systems need to meet stringent safety and reliability requirements and often are developed by companies operating across national boundaries. To educate modern engineers it is critical to establish a methodology for creation of multinational engineering programs, which will produce graduates capable of working efficiently in multidisciplinary teams engaged in international collaboration on industrial RSIC projects. Such projects typically require conformance to specific national and international standards mandated by regulatory authorities.

Modern systems are heavily software-centric, implementing reactive and time-critical software, where safety is the issue and the margin for error is narrow. Examples include aircraft avionics, air traffic control, space shuttle control, medical equipment, and nuclear power stations. It is vital for future software developers to understand the basic real-time applications concepts. The issues of timing, concurrency, inter-process communication, resource sharing, interrupts and handling external devices are of primary importance. The area of real-time safety-critical control systems is one of the most advanced and challenging fields of computer science and engineering, depending on designs developed according to the latest advances in science and modern principles of engineering practice.

The study discussed in this paper is focused on the creation of international curriculum framework centered on RSIC – this important aspect of the computer-system-control-software engineering education. The study explores the mechanism for involving students from multilingual geographically separated institutions in a coordinated educational experience. It exposes them to the problems, methods, solution techniques, infrastructure, technologies, regulatory issues, and tools in the domain of dependable real-time safety-critical software-intensive control systems. The ultimate objective is the creation of a model RSIC curriculum, which can be used by engineering schools both in the USA and the EU. This objective addresses the nations' needs for researchers and developers of real-time safety-critical systems who are capable of engaging in projects spanning the nations' boundaries and promoting a student-centered, transatlantic dimension to higher education and training.

2 Global Aspects of the Software Industry

The modern world, which relies increasing on international collaboration and globalization, is a key element for the nations of the world to come together in a peaceful way. Creation of common educational experience allows young engineers to find commonalities; it promotes teamwork and collaboration in joint projects crossing the nations' boundaries. Specifically, in the area of RSIC systems used in a regulated industry, it is critical to be familiar with the variety of regulations, standards, and guidelines required for designing, implementing and approving software-intensive systems. The proposed study creates a viable vehicle for such solutions.

There is a clear need to prepare professionals for international collaboration. Understanding critical issues related to RSIC, the tools and techniques used, and documentation required for approval of systems in a regulated industry is critical for the future global projects. One such example is the RTCA – Radio Telecommunication Committee for Aeronautics (`www.rtca.org`) charged with providing guidance to the Federal Aviation Administration. RTCA works jointly with EUROCAE (`www.eurocae.org`) representing the European aviation community. Special committees and working groups are formed to address the aviation community guidelines. This international forum including representatives from various countries needs to reach consensus to produce guidance documents. The study is meant to support such international collaboration and consensus building.

3 Curricular Issues

Despite well established computing curricula [1,2], the variety of excellent engineering offering in colleges and universities on both sides of the Atlantic, at this time, there is no international, interdisciplinary curriculum that directly focuses on real-time control systems, dependable software development, safety, reliability and the certification issues in highly regulated industries like aerospace, medicine, transportation, and nuclear energy – a curriculum that would also include globalization aspects of the modern engineering profession. The current study will lead to the design of such curriculum framework, identification of implementation and assessment mechanisms, collection of data necessary to evaluate the process, and guidelines for expansion of the proposed approach to other engineering programs. These objectives are consistent with the following complementary goals:

- Identify a methodology for design and implementation of a transatlantic, multidisciplinary engineering program.
- Stimulate students to follow careers by encouraging them to consider the area of real-time safety-critical control systems and expose them to opportunities of international collaboration.
- Encourage the exchange of staff and students between collaborating institutions.
- Offer multidisciplinary and multicultural experiences to students who would not otherwise have such opportunities.
- Provide a forum for the faculty exchange of ideas on the issues of curricula building, laboratory experiments, and assessment activities.
- Create a foundation for Internet-based laboratory educational experiences, which will expose students from different countries to tools, methods, and techniques used in the creation of highly dependable safety critical systems for regulated industry.
- Stimulate the teaching staff of the European partners to develop and introduce English versions of lectures and teaching materials.
- Foster a strong technological and education research base.

4 Activities

The two-year study, supported by the US Department of Education Fund for Improvement of Post-Secondary Education (FIPSE) and the EU European Commission, is dedicated to creation of a unique RSIC curriculum focusing on real-time software-intensive control and safety-critical dependable systems. The study involves international collaboration of four universities, which allows for exploration of global implication of offering transferable engineering curricula. The partners include one American and three European universities located in three different EU countries, where English is not the language of instruction (Poland, France, Czech Republic). These partners have adequate educational/research potential; and, through their industry and international outreach, they also recognize the needs of the current and prospective labor market for real-time control education, both in Europe and in the USA. They have published on the issues related to engineering curricula improvement [3-8].

A two-pronged approach is used. The first includes active partners' collaboration on identification of the learning objectives and outcomes, description of the curriculum core and supporting units, development of guidelines on the implementation and assessment, identification of the technology infrastructure, and the description of faculty and staff requirements, pedagogy and delivery concepts, accreditation issues and constraints, etc. On-site research by the project faculty and selected students will be enhanced by frequent communications and dedicated working sessions at the partners' sites. The second part of the approach is a practical case study on how the proposed framework can be implemented by the partner institutions. This will include identification of the existing or easily modifiable courses, which can be used as units in the RSIC curriculum, The case study will also include a description of the laboratory infrastructure, necessary administrative procedures (admission, scheduling, and credit transfer), an assessment methodology, and experimental development and delivery of a selected RSIC unit to partner institutions. This experimental concurrent delivery will engage on-site students only. The observations collected in the case study will be an integral component of the project. The knowledge gained from the experience and relevant observations will constitute a base for establishing a dedicated international transatlantic program in Real-Time Software-Intensive Control and as a framework for development of other global engineering curricula.

To facilitate the participating partners' collaboration during the project, a dedicated website (via Blackboard) has been established to exchange files, provide a discussion forum, and post progress reports. The site is a test for use in a classroom setting for distance courses offerings, which allows the students from geographically distant sites to engage in a common educational experience.

The following deliverables are expected to be achieved at the project completion:

- An analysis of industry requirements for graduates of the RTSC domain
- An international, interdisciplinary Real-Time Software Intensive Control Systems curriculum framework
- A design for a selected unit supporting the proposed RSIC curriculum with the draft of lecture materials and laboratory handouts
- A plan and a pilot implementation of a laboratory infrastructure allowing students to actively participate in class activities and experiments on a remote basis
- Experimental concurrent delivery of the designed unit at the four partner sites
- Identification of activities and data for program assessment and evaluation, and those issues and elements required to consider program accreditation
- Reflection on a process and methodology for creation of multidisciplinary, transatlantic engineering programs, including guidelines for extension of the approach to other engineering programs,

5 Assessment

Quantitative and qualitative indicators need to be used to assess the degree of fulfillment of the planned objectives in the proposed time-frame, based on the detailed plan of work with specific timeline and detailed deliverables. Three categories of

indicators, related to the output, results, and impact will be identified and tracked. The output indicators will provide information elaborating the immediate and short-term effects resulting from the planned execution of the project. These will include curricula design, educational events, external institutions that have benefited from the project outputs, created artifacts, and presentations shared with external audiences. The results indicators will provide information on the project output and the immediate results: new cooperative initiatives, published papers, position documents generated, etc. The impact indicators will measure the long-term effects: institutions implementing the proposed framework, learning tools and services created by the project, citations and references to the project documents and papers, third-party references to project outcomes, etc.

Real-time software-intensive safety-critical control systems represent a rather narrow and specific education and research area. Not many courses are offered at the undergraduate and graduate level, since appropriate coverage requires a significant amount of diverse domain knowledge difficult to include in typically overloaded computer science and engineering programs.

6 Conclusions

Nearly all engineering disciplines are ultimately engaged in creation of RSIC systems. The systems are implemented worldwide, which requires a well prepared workforce of scientists and engineers, who can cooperatively address issues in a multi-disciplinary and international fashion. This project is intended to strengthen international co-operation and the global links in engineering education. An interdisciplinary specialization in RSIC was selected to produce not only a number of educational artifacts in a domain in high demand by the industry, but also (what is more important) a process and a methodology for creation of engineering programs with a compatible quality assurance and assessment process. The graduates of such program will be better prepared to work on projects requiring interdisciplinary and multicultural viewpoints. This enhances mobility of the future workforce and facilitates their advancement and career changes.

The objective of the proposed activity is not only to serve the critical population of safety-critical real-time control system developers, but also to disseminate results and provide guidelines to a broader audience of engineering education faculty. The project will capture the process and methodology used by this multidisciplinary and geographically diverse activity for potential re-use by others. Through internet-discussions and face-on meeting activities, this project will develop the platform and materials to create RSIC curriculum framework acceptable on both sides of Atlantic. The collected observation and data will provide the base and guidelines for future implementation of complete coordinated multinational engineering programs.

References

1. *Computer Engineering 2004, Curriculum Guidelines for Undergraduate Degree Programs in Computer Engineering,* The Joint Task Force on Computing Curricula, ACM IEEE Comp. Society, December 2004 (`http://www.acm.org/education/CE-FinalReport.pdf`)
2. *Software Engineering 2004, Curriculum Guidelines for Undergraduate Degree Programs in Software Engineering,* The Joint Task Force on Computing Curricula, ACM IEEE Comp. Society, August 2004 (`http://www.acm.org/education/curricula.html`)
3. W. Grega, Towards the Bologna-BMD Model: Poland, in: *Towards the Harmonization of Electrical and Information Education in Europe* (J.M. Thiriet editor), Nancy University 2003, ISBN 972-97738-3-1, 152 – 158
4. T. B. Hilburn, W. S. Humphrey, *The Impending Changes in Software Education,* IEEE Software, Vol. 19, No. 5, September / October, 2002.
5. A. J. Kornecki, *Real-Time Computing in Software Engineering Education,* Proceedings of 13th SEE&T Conference, Austin, TX, March 2000, pp. 197-198
6. A. J. Kornecki, *Real-Time Systems Course in Undergraduate CS/CE Program,* IEEE Transactions on Education, CD-ROM Supplement, Vol. 40, No. 4, November 1997, pp. 295-296.
7. J. M. Thiriet, M.J. Martins, Editors, Monograph: *Towards the Harmonization of Electrical and Information Engineering Education in Europe* - Ed. EAEEIE, August 2003, 350 pages, ISBN, book version: ISBN 972-97738-2-3, ISBN: CD-ROM version: ISBN 972-97738-3-1.
8. J. M. Thiriet, M. Robert, P. Lappalainen, M.Hoffmann, M. J.Martins, A.Seoane, *Toward a Pan-European Virtual University in Electrical and Information Engineering,* IEEE Trans. on Education, May 2002, Vol.45, n. 2, pp.152-160.
9. F. L. Whetten, A. J. Kornecki , *Distance Teaming: A Bi-located Undergraduate Program in Computer Engineering,* Proceedings of 13th Annual Conference on Distance Teaching and Learning, Madison, WI, 1997, pp. 361-364.

Proceedings of the International Multiconference on
Computer Science and Information Technology pp. 949–952
ISSN 1896-7094
© 2007 PIPS

Educational Objectives for Embedded Systems

Wolfgang A. Halang

Fernuniversität in Hagen
Chair of Computer Engineering
58084 Hagen, Germany
Wolfgang.Halang@FernUni-Hagen.de

Abstract. Adequate concepts, professionalism and ethics are identified as the goals in properly educating systems engineers for professional life. Criteria and concepts appropriate to design feasible solutions must be based on fully understanding the peculiarities of the environments where embedded systems are employed. Bad practices of the computing profession need to be overcome to foster professionalism, and the consequences of the shift from hardware to software must be realised. This gives rise to some rules of ethics and proper conduct.

Which issues should be addressed by curricula on real-time and embedded systems that are longer lived than technical artefacts, and that constitute elements of proper university education? My answer is that engineering professionals should be educated on the following fundamental essentials:

adequate concepts, professionalism, ethics.

Computer control totally differs from other areas of computing as processing takes place not only in real time, but also in real environments into which real-time systems are embedded and to whose requests they react. Essential is that they always deal with reality, and that there are always physical environments with their own laws, which cannot be forced under the rules of computers. Although performing control functions, embedded systems thus always take the position of slaves, whereas the environments constitute the masters. This very nature of real-time computing must particularly be taken into account when developing abstractions in order not to lose coherence with reality.

Owing to the embedding into physical environments, systems engineers must have a deep understanding of computing, of physics, and of the application-specific requirements. The methods to be employed are not general-purpose ones, but must always be selected in environment- and process-specific ways. Dealing with the real world also implies that physical constraints must be observed, holding first and foremost for the time dimension and time-related requirements. When designing systems, this calls for applying static design and analysis methods as well as the concept of resource adequacy, i.e., computers must physically provide all resources required in the worst cases. In other words, one may not resort to the popular method of dynamically creating "virtual resources".

By their very nature, embedded real-time systems are always complex. To cope with them, engineers must employ a holistic view as design baseline, and methods apt to master complexity. Moreover, real-time systems are almost always safety-related, as they are embedded in environments which they control. For this safety issue the requirement of dependability results to be fundamental for real-time systems. It can only be met by a priori safety-licensing.

The performance criteria applicable to embedded systems are only very seldom quantitative, but mostly of a qualitative (binary, i.e., requirement fulfilled or not) nature. This is best exemplified by the most important criterion, viz., whether (hard) deadlines are met or not. Corresponding to the holistic view to be taken at design time, cost evaluations must be oriented at overall figures made up of costs for hardware, software, the embedding environment or processes, safety-licensing, potential damage in case of malfunction etc.

In contrast, the common focus on meaningless processor costs is typical for a number of mainstream misconceptions, which are rather unnatural from the common sense point of view. A further misconception is to conclude from randomly distributed data arrivals that real-time systems should not behave deterministically. On the contrary, as only deterministically behaving systems are safety-licensable, all computer reactions must be precisely planned and fully predictable, particularly for all cases of conflict and error. The need to observe worst-case requirements and physical constraints renders harmful all so-called dynamic and virtual features, which are so popular in mainstream computing.

Scheduling is still the topic of most active academic research in real-time systems. Its rationale is to optimise the utilisation of resources, which makes sense as long as resources are scarce and expensive. It turns into nonsense, however, when just the utilisation of processors is considered, which are now very cheap and available in abundance. For embedded and safety-critical real-time systems, maximum processor utilisation is irrelevant and, thus, cannot be a credible research topic anymore. On the system level, the costs of overdimensioned and, hence, underutilised processors are negligible, but may, on the other hand, be very well invested if system dependability can be enhanced by simplifying software. Since less or less complex software is also cheaper, higher spending in processors may even result in overall cost savings. In other words, the only meaningful optimisation criterion is to minimise the bottom line.

When it comes to the notion of time, the influence of mainstream computer science is most detrimental. For real-time control systems, however, time is the central concept, because processes proceed in time, and there is no functional correctness without correct timing. Time is an absolute measure and a practical tool allowing to plan (technical) processes and future events with all their mutual interactions requiring no further synchronisation. That is why the concept of time, which is an abstraction from natural phenomena and a model of thought, is so useful. The model employed in everyday life, viz., as the fourth dimension of our Euclidian space of experience, is also appropriate to design real-time systems. It is important to note that it is nonsense to model time, because time is already a model, that the time relevant to technical systems is defined by law,

that incorrect timing may have legal consequences such as liability suits, and that the model is closely approximated by technical means in form of atomic clocks, which provide the legal Universal Time Co-ordinated. All these notions are strange for computer scientists, but essential to real-time systems engineers.

As a scientific and technical discipline, computing is still highly immature, which is manifested by fashions, buzzwords, and hypes for supposedly new developments and artefacts. As a consequence, in the embedded systems field often adequate and proven tools, such as proper real-time programming languages, are abandoned in favour of worse tools, such as the non-real-time language C, for irrational reasons like gossip among managers, but not on the basis of sound evaluations of technical merits. In such an environment it comes as no surprise that good designs and engineering solutions are not recognised as such, are not well documented, and are not disseminated as design patterns. Instead, one resists to learn, and re-invents the wheel about every five years. Observing the field for over 30 years now, I have already seen many of these re-inventions, which very often are clearly inferior to the original solutions. This is highly unprofessional. A scholarly discipline ought to place one layer of knowledge on the other, and ought to prefer good solutions over just new ones.

Hence, there was negative progress in the field, which was more advanced around 1980 than it is now. This is not only evidenced by the situation in languages, but also, for instance, by the disappearance of good real-time computers or the fact that no contemporary real-time operating system can match the desired functionality as described and implemented in 1975. As a result, we witness a loss of quality, reliability, and safety. The reason for this decay is mainly the adoption of general-purpose hardware, software, methods and tools. Their adoption was not only caused by wrongly understood cost considerations, but to a large extent by the already mentioned fashions and other marketing gimmicks.

Systems engineers need to develop a professional "sense of proportion", attributing correct weights to design requirements and issues. An outflow of this principle into the direction of real-time systems research is to address only real problems, but not "toy problems". For designers it is very important to exercise self-discipline. There is a common habit among software developers to program any function anew, because they cherish their own programming skills. Professionals need to refrain from this. Although they may emotionally favour more elegant solutions, they observe design guidelines and programming rules in the interest of dependability. The main objective of self-discipline is to prevent the creation of artificial and unnecessary complexity, and to ease verification.

As result of a remarkable paradigm shift now over 60% of the revenue generated by the electrical and electronic industries can be attributed to software, whereas their tangible products and devices moved to a secondary rôle. Real-time systems engineers must be fully aware of this fundamentally changed situation. The sense of proportion demanded above will lead them to address in their designs the most pressing problems, i.e., the ones related to software and its intrinsic dependability deficit.

In contrast to instruction of knowledge and training of capabilities, the objective of education is to form personalities. From mature and valuable persons we expect a behaviour guided by ethic principles. Both the design of systems and the design of tools ought to be oriented at humans and their thinking. Systems are to serve people. Thus, system design should not be driven by interesting new technology or market forces, but by the real needs of people. There is a plentitude of so-called methodologies and formal methods around, which claim to support design processes and to allow for rigorous system verification. None of them found widespread acceptance. One of the reasons for this failure is that many tools do not appeal to the way of thinking of their envisaged users, in this case engineers.

Among the real needs of people is that technical systems should be dependable. This is of increasing importance as daily life in all its aspects more and more depends on technical systems of any kind. Embedded control systems are among the most complex and challenging ones as they are composed of elements from various engineering disciplines, and of hardware and software with their totally different natures. Actually as system integrators, on control engineers rests most of the responsibility for the proper function of technical systems. As many of them are safety-critical, real-time systems engineers must always be aware that life and health of persons depend on them. Therefore, their work must be guided by the fundamental principle to fight (artificial) complication, and to strive for the most simple solutions, because the latter allow for most easy verification.

The bad influence of the computing community can be overcome if systems engineers observe the following rules of conduct already well established in other scientific and engineering disciplines:

- take care that programs and systems remain intellectually tractable,
- prefer simple structures and solutions for their clarity,
- fight complexity and strive for simplicity,
- base your work on the achievements of earlier generations,
- do not abandon the good for the worse,
- resist market forces and fashions.

To summarise, the fundamental concepts of real-time computing and embedded systems differ considerably from mainstream thinking in computer science, causing the field to be much more closely related to electrical and control engineering. The mainstream computer science thinking needs to be overcome in order to meet the demands of real-time and real-world applications, and not to fall behind real-life practices. The pressing problems of embedded systems are their integrity and safety imposing responsibility for human life and well-being on real-time systems engineers. Given the present state-of-the-art, to cope with these problems first and foremost the dependability of software must be improved. To this end, complexity has to be fought in all aspects. In order to foster this, their education needs to convince future real-time systems engineers that simple solutions are the most difficult and challenging ones, because they demand high innovation and full intellectual penetration of subject matters.

Workshop on Ad-Hoc Wireless Networks: Urban Legends and Reality (WAHOC'07)

As a concept, wireless ad-hoc networking has been around for at least 20 years, but, despite the extensive coverage in academic literature, its practical success so far has been less than spectacular. Most contemporary attempts at exploiting the ad-hoc paradigm in real-life applications fall into the realm of sensor networks, where it promises low-cost, ease of deployment, flexibility, reliability, resilience and ubiquity. However, few practical deployments of sensor networks are actually ad-hoc based: there seems to be a considerable reluctance in the industrial world to embrace "mesh networking," and most actual solutions are based on asymmetric schemes involving various flavors of base stations. The purpose of this workshop is to get to the roots of that reluctance. We solicit papers addressing the practical aspects of ad-hoc wireless networking, with stress on case studies and empirical verification of ideas. We shall not shy away from negative conclusions, as long as they are well supported.

The list of topics includes (but is not limited to):

- Routing protocols for ad-hoc wireless networks
- Medium access control schemes
- Modeling methodologies for wireless channels
- Simulation systems and techniques
- Performance studies and comparisons
- Case studies (ad-hoc networks at work)
- Security issues in ad-hoc networking
- Asymmetric wireless networks with ad-hoc components
- Cooperation issues at the MAC and network layers
- Pricing and incentive arbitration in ad hoc mesh topologies

International Programme Committee

- Ian F. Akyildiz, Georgia Tech, USA
- Franco Davoli, University of Genoa, Italy
- Janelle Harms, University of Alberta, Canada
- Jacek Ilow, Dalhousie University, Canada
- Ahmed E. Kamal, Iowa State University, USA
- Bozena Kaminska, Simon Fraser University, Canada
- Jerzy Konorski, Gdansk University of Technology, Poland
- Pawel Matusz, Intel Technology Poland, Poland
- Marek Natkaniec, AGH - University of Science and Technology, Poland
- Ioanis Nikolaidis, University of Alberta, Canada
- Wladek Olesinski, Sun Microsystems Laboratories, USA
- Andrzej Pach, AGH - University of Science and Technology, Poland
- Piotr Pacyna, AGH - University of Science and Technology, Poland
- Lu Yan, University College London, UK

Organizing Committee

Paweł Gburzyński, University of Alberta, Canada
Jozef Wozniak, Gdansk University of Technology, Poland

Proceedings of the International Multiconference on
Computer Science and Information Technology pp. 955–964
ISSN 1896-7094
© 2007 PIPS

Acoustic Target Classification In Wireless Audio–Sensor Networks

Baljeet Malhotra, Ioanis Nikolaidis, and Janelle Harms

Department of Computing Science,
University of Alberta,
Edmonton, Alberta, Canada T6G 2E8
Emails: {baljeet,yannis,harms}@cs.ualberta.ca

Abstract. Target tracking is one of the important problems in wireless sensor networks. We consider an aspect of tracking: the classification of targets based on the acoustic signals produced by vehicles. We present a naïve classifier and simple distributed schemes based on features extracted from acoustic signals. We demonstrate a novel way of using Aura matrices to create a new feature derived from the Power Spectral Density (PSD) of a signal, which performs at par with other existing features. An experimental study has been conducted using real acoustic signals of different vehicles in an urban setting. Our proposed schemes using a naïve classifier achieved highly accurate results in classifying different vehicles into two classes. Communication overheads were also computed to capture the tradeoff of energy cost vs. classification quality.

1 Introduction

Networked sensors can be equipped with various sensing devices, as well as memory, processor, radio, and a power supply. However, they are still constrained by limited memory, processing power, channel capacity, and, most importantly, energy reserves. When tracking is considered as an application, data–intensive sources (e.g., high frame rate / high resolution video) is usually avoided as being more energy expensive than low data rate sources. For this reason tracking using audio signals is usually preferable. Vehicle tracking on acoustic data is based on the fact that different vehicles produce distinctly different acoustic signals because their engine and propulsion mechanisms are unique [9]. The problem of vehicle detection using the acoustic signature has existed for years and many solutions have been proposed in the literature [1, 9, 10]. Recently, target classification based on acoustic signals in wireless sensor networks has been addressed in [3, 6]. The advantage of sensor networks is that they provide redundancy in terms of sensing and processing units. Hence, they can operate together in a distributed coordinated fashion to detect and report the presence of a target vehicle, possibly refining the tracking and classification quality as the target is moving.

We should add that vehicle tracking includes various objectives that must be supported by a number of steps. These steps include vehicle detection, identification and/or classification, and localization. Depending on the specific tracking

objectives, all or combination of these steps may be required. In this paper we restrict our attention to classification. Classification is necessary because sensors can report on a specific moving vehicle only after they recognize vehicles that are of some interest. Classification is naturally more challenging if there are multiple targets of various types (e.g., tanks, jeeps, other types of military vehicles, civilian vehicles, etc.). Furthermore, there may be a number of vehicles of the same type *e.g.* tanks of a particular make. We define as *classification* the problem of identifying which class a vehicle belongs to. Identifying a *particular* vehicle goes one step further and is not within the scope of the current paper.

Various techniques have been proposed to address the classification problem [1, 3, 9], relying on *feature extraction* that differ in the way features are extracted. For example, [1], proposed a wavelet based method for feature extraction, which works as following: three different types of acoustic signature are extracted: (i) squeak sound, (ii) sound under motion, and (iii) exhaust sound. The data points contained by each of these signatures are decomposed to a 12 element feature vector using the multi-resolution analysis [7]. These 12 element feature vectors represent the energy concentration of the signature signal at 12 different resolution levels. The continuous wavelet transform (CWT) and the STFT (short-time Fourier transform) plots were used for two other feature vectors. Finally, these feature vectors are used to compute the distance between the reference and unknown signatures. Wu *et. al.* in [10] proposed a principle component analysis based method for recognition of acoustic signatures. The basic idea of their proposed method is to use together the mean adjusted sound spectrum, and key eigenvalues of the covariance matrix to characterize an acoustic signature.

An adaptive threshold based algorithm is proposed in [2] for vehicle detection, based on the average energy of an acoustic signal crossing a threshold value before a decision on the detection of the vehicle can be made. The threshold is updated adaptively. Also, [3] details experiments carried out during the 3rd SensIT situational experiment (SITEX02) organized by DARPA/IXO. Various military vehicles were used in these experiments, real world data was generated and archived for future studies. The objective was to detect and accurately locate vehicles using energy-based localization algorithm. Frequency spectrum based features are extracted from the acoustic and seismic signals captured by the sensors. These features were extracted by using 512–point FFT. Then, for classification purposes, three different approaches are used *i.e.* (i) k-nearest-neighbor classifier (ii) Maximum Likelihood classifier, and (iii) support-vector-machine classifier. In [6], a framework for collaborative signal processing in sensor networks is designed for the purpose of multiple targets detection, classification, and tracking.

In this paper we study the impact of increasing the complexity and memory footprint of a classification algorithm to achieve better classification accuracy. We consider this to be a reasonable tradeoff since what is usually assumed to be expensive in terms of energy is communication, and not computation/storage (within reason of course). While we are also increasing the computation cost, we argue that as long as the computation is allowed to be completed within a

reasonable amount of time, computation can be spread over a longer period of time by proper reduction of the CPU clock [13].

One distinct contributions of our work is that in order to maximally benefit from the information collected by a sensor, we consider multiple representations of features. Some, are well known (FFT, PSD, etc.) but we also introduce *Aura matrices*. Aura matrices [4] have been used in the past for analyzing and predicting texture patterns [11]. In our study we create "artificial" 2-dimensional "textures" by arranging PSD data into matrices, and then using Aura matrices to summarize the information of the 2-dimensional matrix. In other words, Aura matrices attempt to visually approximate the arrangement of PSD values. For details about the construction of Aura matrices the reader is referred to [4].

Section 2 presents the features considered in this study. Section 3 describes a naïve classification scheme and elementary forms for a distributed implementation over a sensor network. Section 4 presents performance evaluation results in terms of classification accuracy and energy expenditure tradeoffs for the different distributed implementations. Finally, Section 5 summarizes the findings of the paper and outlines future research objectives.

2 Features Extraction

A vehicle sound is a stochastic signal. In practice, sound of a moving vehicle observed over a period of time will not be a stationary signal. However, a signal of fairly short duration can be treated as a stationary signal [10]. In our case we chose the signal's duration to be 23.27 ms *i.e.* 512 data points sampled at a frequency of 22 kHz. In our study we considered five features that are generated using FFT and PSD of time series data of a given signal. In the rest of the paper the terms signal and signature are used interchangeably.

1. **Linear FFT feature (LFFT):** This feature is generated using FFT of 512 data points that gave us a linear vector (of size 512) representing frequencies with a resolution of 21.48 Hz.
2. **Linear PSD feature (LPSD):** This feature is generated by taking Power Spectral Density (PSD) estimates of 512 data points using the Thomson multi-taper method. With a resolution of 21.48 Hz that method gave a linear vector (of size 512) to form a linear PSD feature.
3. **Multidimensional FFT feature (MFFT):** In this case 50 blocks of 512 FFT data points are used to form a multidimensional FFT feature. This feature can be seen as a matrix of size 512×50. The size 50 was determined by trial and error method using our acoustic data set.
4. **Multidimensional PSD feature (MPSD):** In this case 50 blocks of 512 PSD data points are used to form a multidimensional PSD based feature. This feature can be seen as a matrix of size 512×50.
5. **Aura of a Multidimensional PSD feature (AMPSD):** It has been shown in [5] that power spectral density (PSD) is not an optimal feature for signal recognition. We explored to improve the PSD based feature using

some established statistical techniques, namely Aura matrices. In order to construct AMPSD features we simply compute Aura of a MPSD matrix. For computing the Aura of a matrix the reader is referred to [4].

The first four features are popular in the signal classification literature. The fifth feature (AMPSD), which is part of our contributions has been added to improve the performance of MPSD feature.

3 A Naïve Classifier

Existing techniques such as k-Nearest Neighbor (k-NN) can be used by sensor nodes to perform the classification. It is based on the idea that similar objects are closer to each other in a multidimensional feature space. k-NN is one of the simplest, yet accurate, classification methods and recently it has been used in sensor networks for target classification [3, 6]. Unfortunately, finding k for the optimum solution is non-trivial. In contrast to k-NN algorithm we adopt a naïve approach where first a learning set $|U|$ is defined for each class of vehicles. Equal number of samples are assumed to be in the learning set of these classes. Note that class labels of the learning samples are known in advance. Now in order to classify an unknown sample using a particular feature, the naïve classifier does the following: For each class, and for each sample of the $|U|$ samples in the learning set of each class, we calculate the distance of the unknown sample from the learning set samples. We determine the average distance of the unknown sample from the learning set samples of each class. The unknown sample is determined to be in the class with the smallest average distance.

As a vehicle crosses through an area monitored by a sensor network, the nodes self-organize in neighborhood "clusters" using a technique similar to [12] but where tie breaking criteria for selection of the "master" node is the signal quality of the monitored vehicle (the master is the node with higher average power of received signal, hence possibly closest to the vehicle). Each of the neighborhoods can perform a classification method independently of other neighborhoods. However, multiple neighborhoods may collaborate with each other for two main reasons: (i) better accuracy in classifying a vehicle, and (ii) sharing the costs associated with classification. In our study we examined various scenarios of single and multiple neighborhoods based classification. We propose four basic schemes:

- **Single Neighborhood Based Classification using Local Signatures (SNBC-LS):** In this scheme each sensor in a neighborhood uses its own local copy of vehicle's signature. Sensors compute the distance of unknown sample from known samples in each of the class using their own (local) copy of unknown sample. The master node collects results from all participating nodes in the neighborhood and classifies the unknown sample (vehicle).
- **Single Neighborhood Based Classification using Global Signatures (SNBC-GS):** In this scheme each sensor in a neighborhood uses a global copy of vehicle's signature. A global copy of vehicle's signature is transmitted

to them by the master node of a neighborhood. Sensors compute the distance of unknown sample from the known sample in each of the class using the global copy of the unknown sample. The master node collects results from all participating nodes in the neighborhood and classifies the unknown sample (vehicle).

- **Multiple Neighborhood Based Classification using Local Signatures (MNBC-LS):** In this scheme a master node not only collects results from sensors in its own neighborhood, but it also invokes its adjoining neighborhoods to seek the classification results. All sensors in participating neighborhoods use their local copy of vehicle's signature.
- **Multiple Neighborhood Based Classification using Global Signatures (MNBC-GS):** The basic difference between this scheme and the previous scheme (MNBC-LS) is that sensors in a particular neighborhood use a global copy of vehicle's signature provided to them by their respective master nodes.

4 Experimental Study

We used acoustic signals of various urban ground vehicles. These acoustics signals were recorded using Panasonic US395 microphone. Approximately 50 samples of various vehicles were recorded at two main locations of Bonnie-Doon mall and University of Alberta bus stop in the city of Edmonton. This real database of acoustic samples consisted of acoustic signatures of ETS buses, which are part of the public transportation system at the city of Edmonton, different types of cars, small trucks, SUVs, and mini vans. All acoustic signals were first recorded to the local memory of the microphone, and then transferred to MATLAB for the simulation of classification algorithms.

We assume that every sensor has a copy of learning set, U for each class. A sensor's copy of unknown (testing) sample is different from others because of their geographical positioning. In order to create a local copy of unknown (testing) sample for a sensor we attenuate the signal based on its distance from the moving vehicle and add noise to it. A vehicle's sound can also be degraded by reverberations, however, we considered an outdoor open environment, so we have neglected the effect of reverberations. In our simulation we considered various scenarios for sensor setup. In these experiments sensors are assumed to be placed along two straight parallel lines, i.e., as they would be deployed along the sides of a street. Sensors are placed 5m apart and their sensing and radio range is 15m. Vehicle is considered to be moving with a speed of 53± Km/H.

4.1 Performance Metrics

We consider two performance metrics: (i) classification accuracy, and (ii) energy expenditure. Classification accuracy is computed based on the leave-one-out policy. Under this policy one sample is removed from the acoustic dataset consisting of all samples in a class. This sample is called the testing sample. The rest of the

samples in the dataset constitute the learning set U for that class. Class label of the testing sample is assumed to be unknown. Then, the distance of the testing sample is computed from all the samples in the learning set of each class. This process is repeated for all samples in our acoustic dataset. If two samples are represented by matrices $X_{m\times n}$ and ${X'}_{m\times n}$, then the distance between them is computed as following:

$$d = \sum_{q=1}^{|n|} \sum_{p=1}^{|m|} \sqrt{(x_{pq} - {x'}_{pq})^2} \tag{1}$$

Classification accuracy is calculated as a percentage of testing samples that are correctly classified from the total number of testing samples. In the experiments we used a simplistic case of only two classes. The objective was to classify the previously mentioned vehicles into two classes *i.e.* ETS buses and other vehicles that are not ETS buses.

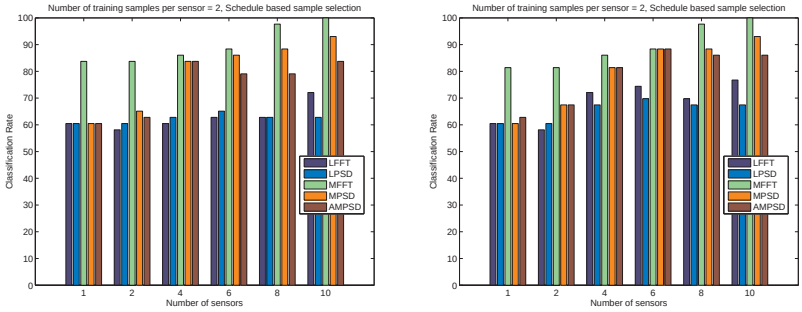
4.2 SNBC-LS & SNBC-GS

In the SNBC-LS scheme all sensors use their own copy of vehicle's signature. Note that every sensor's copy might be different, because of its position. In these experiments we attenuated a sensor's signal based on its distance from the moving vehicle. In the case of SNBC-GS scheme, all sensors use a common copy of an unknown vehicle's signature that is provided to them by the master node.

The results for classification accuracy in SNBC-LS and SNBC-GS schemes are presented in Figures 1(a) and 1(b), respectively. With a better copy of vehicle's signal available to all participating sensors, SNBC-GS scheme performs better than SNBC-LS. Classification using LFFT, LPSD, MPSD and AMPSD based features improved in SNBC-GS scheme. As noticed in these figures classification accuracy improves as the number of sensors per neighborhood increases. The reason for improved accuracy is that collecting distance measurements from all sensors in a neighborhood, and taking their average improves the approximation of the distance measurements. An accuracy of 99% is achived using MFFT feature that is better than the accuracies reported in [8], which uses k-NN classifier.

Communication costs are much higher in the SNBC-GS scheme. The results for SNBC-LS and SNBC-GS schemes are presented in the Figures 2(a) and 2(b), respectively. The reason for larger costs in SNBC-GS are obvious. The master node needs to communicate the signature in its neighborhood. More specifically it transmits 512 data point in the case of LFFT and LMSD features and 512 $\times$ 50 data points for MFFT, MPSD and AMPSD features.

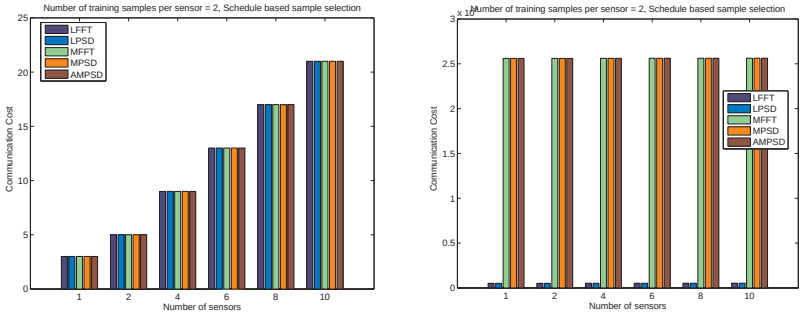
These results suggest that there is a trade-off between communication costs, computational costs, and achieving a higher classification accuracy. A higher classification accuracy comes at a higher energy cost for sensors. Note that when number of training samples per sensor are fixed, then varying the number of sensors per neighborhood affects the classification accuracy. Having more sensors in a neighborhood increases the classification accuracy but at a higher cost



(a) SNBC-LS (b) SNBC-GS

Fig. 1. Classification accuracy.

of communication. On the other hand selection of a feature is also an important decision. Some features are more expensive to use than others, but their classification results are better.



(a) SNBC-LS (b) SNBC-GS

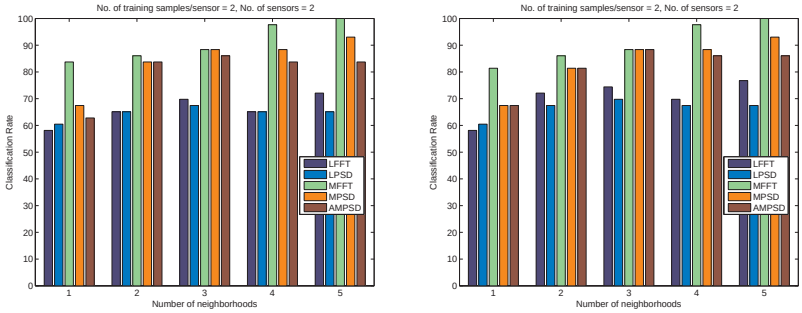
Fig. 2. Communication costs.

4.3 MNBC-LS & MNBC-GS

Classification based on multiple neighborhoods may arise in many situations. Consider the case in which number of nodes (and samples) in a particular neighborhood are not enough to achieve a higher classification accuracy. This may happen due to a number of reasons like malfunctioning of sensors because of insufficient battery power or physical damages. In these experiments we assume that only two sensors are available per neighborhood. We also assume that learning set U is distributed over multiple neighborhoods though the sensors in a single neighborhood have a common learning set. In the MNBC-LS scheme all

sensors in the corresponding neighborhood use their own copy of vehicle's signature. In the case of MNBC-GS scheme all sensors use a common copy provided to them by the master node of their neighborhood.

The results for classification accuracy in MNBC-LS and MNBC-GS schemes are presented in the Figures 3(a) and 3(b), respectively. There is not much difference between classification results for single neighborhood scheme (*e.g.* SNBC-LS) and multiple neighborhood scheme (*e.g.* MNBC-LS). The reason for this behavior is that in multiple neighborhood schemes, learning samples are distributed over multiple neighborhoods that result in a smaller learning set for each of the neighborhoods. Therefore distance calculation of unknown sample is approximated from readings of multiple neighborhoods instead of sensors from a single neighborhood. In the case of global signatures, a similar trend can be seen in multiple neighborhood schemes. With a better copy of vehicle's signal available to all participating sensors in a neighborhood, MNBC-GS scheme performs better than MNBC-LS. Again, classification accuracy using LFFT, LPSD, MPSD and AMPSD based features improved in MNBC-GS scheme.



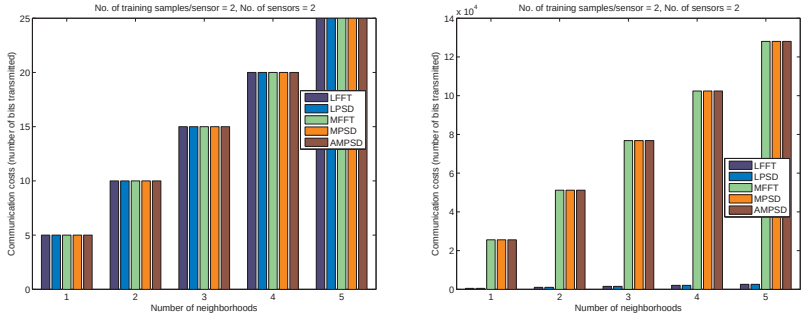
(a) MNBC-LS (b) MNBC-GS

Fig. 3. Classification accuracy.

As expected communication costs are much higher in MNBC-GS scheme. These results are presented in the Figures 4(a) and 4(b) for MNBC-LS and MNBC-GS schemes respectively. As mentioned previously MNBC-GS is more expensive because the master node transmits 512 data point in the case of LFFT and LMSD features and 512×50 data points for MFFT, MPSD and AMPSD features.

5 Conclusions and Future Directions

Classifying ground vehicles is an important problem in wireless sensor networks. Features extracted from the acoustic signatures of these vehicles form the basis for classification. Whether sensor networks provide for efficient implementation of tracking, depends on whether necessary operations, such as classification, can



(a) MNBC-LS (b) MNBC-GS

Fig. 4. Communication costs.

be performed efficiently in a distributed fashion, achieving high classification accuracy at a reasonable energy cost. In this paper we proposed several distributed schemes for vehicle classification. These schemes are based on the idea of collaborations in a single and multiple sensor neighborhoods. We conducted a simulation study using real acoustic signals of urban ground vehicles. Simulation results have revealed the performance of our proposed schemes. Our proposed schemes achieved up to 99% accuracy for a binary classification using a naïve classifier. These results are even better than some existing results on k-NN classification. In the future we would like to improve the efficiency of our proposed schemes. We also plan to conduct an experimental study where we consider more than two classes of ground vehicles.

References

1. R. Braunlin, R.M. Jensen, and M.A. Gallo, Acoustic target detection, tracking, classification, and location in a multiple-target environment, Proc. of SPIE Conf. on Peace and Wartime Applications and Technical Issues for Unattended Ground Sensors, vol. 3081, pp. 57-66, Jul. 1997.
2. J. Ding, S.-Y. Cheung, C.-W. Tan, P. Varaiya, *Signal processing of sensor node data for vehicle detection,* Seventh International IEEE Conference on Intelligent Transportation Systems, 2004.
3. M. Duarte and Y.-H. Hu, *Vehicle classification in distributed sensor networks, Journal of Parallel and Distributed Computing,* vol. 64 no. 7, pp. 826–838, 2004.
4. I. M. Elfadel and R. W. Pickard, *Gibbs Random Fields, Cooccurences, and Texture Modeling,* IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 16, No. 1, Jan. 1994.
5. L. German, *Signal recognition: Both components of the short time Fourier transform vs. power spectral density,* Pattern Analysis and Applications, vol. 6, no. 2, pp. 91–96, 2003.
6. D. Li, K. D. Wong, Y. H. Hu, and A. M. Sayeed, *Detection, Classification, and Tracking of Targets in Distributed Sensor Networks,* IEEE Signal Processing Magazine, 19(2):17–30, March 2002.

7. S. G. Mallat, *A Theory for Multiresolution Signal Decomposition: A Wavelet Representation,* IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 11, no. 7, July 1989.
8. B. Malhotra, I. Nikolaidis, J. Harms, *Distributed Classification of Acoustic Targets in Wireless Audio-Sensor Networks,* Submitted to the Computer Networks (special issue on Wireless Multimedia Sensor Networks), 2007.
9. G. Succi and T. K. Pedersen, *Acoustic target tracking and target identification—recent results,* Proc. of SPIE Conf. on Unattended Ground Sensor Technologies and Applications, vol. 3713, pp. 10–21, Jul. 1999.
10. H. Wu and M. Siegel and P. Khosla, *Vehicle Sound Signature Recognition by Frequency Vector Principal Component Analysis,* IEEE Transactions on Instrumentation and Measurement, vol. 48, no. 5, Oct. 1999.
11. X. Qin and Y.-H. Yang, *Similarity Measure and Learning with Gray Level Aura Matrices (GLAM) for Texture Image Retrieval,* Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), 2004.
12. S. Basagni, *Distributed Clustering for Ad Hoc Networks,* Proceedings of the 1999 Intl. Symposium on Parallel Architectures, Algorithms and Networks (SPIN), 1999.
13. M. Weiser, B. Welch, A. Demers, and S. Shenker, *Scheduling for Reduced CPU Energy,* Proceedings of the 1st USENIX Symposium on Operating Systems Design and Implementation., pp. 13–23, November 1994.

Proceedings of the International Multiconference on
Computer Science and Information Technology, pp. 965 – 974
ISSN 1896-7094
© 2007 PIPS

Analysis of IEEE 802.11b/g Card Behavior in Multirate Ad-hoc Networks

Katarzyna Kosek, Szymon Szott, Marek Natkaniec, Andrzej R. Pach

Department of Telecommunications
AGH University of Science and Technology
Kraków, Poland
{kosek, szott, natkanie , pach}@kt.agh.edu.pl
http://www.kt.agh.edu.pl/

Abstract. In multirate ad-hoc networks, mobile stations usually adapt their transmission rates to the channel conditions. This paper investigates the behavior of IEEE 802.11b/g cards in a multirate ad-hoc environment. WLAN network cards from different vendors are analyzed. The theoretical upper bound estimation of the throughput in multirate ad-hoc networks is derived. The measurement scenarios and obtained results are presented. For result validation the theoretical and experimental values are compared.

1 Introduction

Wireless networks based on the IEEE 802.11 family of standards have become widespread in recent years. Even though Access Points are being deployed both at home and public places, it is the ad-hoc mode of 802.11 which is expected to become increasingly popular in the near future. One of the features of 802.11 devices, which can significantly increase their performance, is the use of adaptive multirate transmission schemes.

All four currently used IEEE standards support multirate, i.e., 802.11 [5], 802.11a [6], 802.11b [7], and 802.11g [9]. Each of them allows different speeds in the uplink and downlink directions depending on current physical conditions of the radio channel.

The theoretical performance of multirate capable devices has been measured extensively but practical results vary. This is not only due to different testbeds and radio conditions, but also because of vendor implementations.

A good example of this problem can be found in [17], where several IEEE 802.11 cards from different vendors were analyzed. The stress was laid on MAC implementations and hardware delays. Two meaningful conclusions appeared. First of all, it was shown that a notable unfairness in rate selection appears among different commercial cards and, furthermore, that the unfairness is a result of different hardware/firmware implementations. It is expected that a similar situation will be observed in multirate IEEE 802.11b/g ad-hoc environments.

The aim of this work is to show the differences in performance and interoperability of multirate IEEE 802.11b/g cards of the following vendors: Linksys, Lucent and Proxim. All cards were operating in ad-hoc mode. The concept of the presented scenario was to capture and analyze FTP traffic sent by a server to two clients.

The rest of the paper is organized as follows. The state of the art is presented in Section II. A mathematical model for calculating transmission rates in 802.11 is described in Section III. The measurement scenarios and results are shown in Sections IV and V, accordingly. Section VI gives a validation of the achieved results. Section VII closes the paper summarizing the main conclusions.

2 State of the Art

The IEEE 802.11 family of standards does not provide any method of automatic rate selection in the presence of multirate capable devices. Because of this, there are many methods of choosing the appropriate rate and it is up to the card vendors to decide which one to use.

The co-operation of cards of different standards is possible because the preamble and header of each frame is sent with the basic rate – understandable by all cards. Only the payload can be sent at higher rates (cf. Table 1). This is especially important for 802.11b and 802.11g co-operation.

Table 1. Comparison of Preamble, Header, and Payload Rates

Mode (lp/sp: long/short preamble)	**Physical Layer Convergence Procedure (PLCP)**		**Payload [Mbps]**
	Preamble [Mbps]	*Header [Mbps]*	
802.11	1	1	1 or 2
802.11b lp	1	1	1,2, 5.5 or 11
802.11b sp	1	2	2, 5.5 or 11
802.11g lp	1	1	1, 2, 5.5, 6, 9, 11, 12, 18, 22 (optional) 24, 33 (optional), 36, 48 or 54
802.11g sp	1	2	2, 5.5, 6, 9, 11, 12, 18, 22 (optional) 24, 33 (optional), 36, 48 or 54

It must also be noted that transmission rates are not linear. Therefore, e.g., an 11 Mbps link with a delivery ratio of just above 50% is always better than a 5 Mbps link.

Multirate algorithms can be based on statistics. The ARF (Auto Rate Fallback) [1] protocol is perhaps the most commonly used (and possibly the first such algorithm). To determine the channel quality, ARF utilizes link layer ACK frames (i.e., the Frame Error Rate, FER). After a given number of consecutive ACKs have been received, the transmission rate is increased. The loss of a similar number of ACKs causes the node to decrease the transmission rate. The main advantage of ARF is that it is simple to implement and does not interfere with the 802.11 standards. However, it is slow to adapt to channel conditions – it tries to change the rate even for stable links, and can mistake collisions for channel losses.

Most popular WLAN cards currently use the Atheros chipset which (under Linux) can be configured with the innovative MadWiFi driver. This driver implements three different rate adaptation algorithms: Onoe [12], AMRR (Adaptive Multi Rate Retry) [10], and SampleRate [2]. Onoe, the default algorithm, is based on ARF and looks for the highest bitrate that has a loss rate less than 50%.

A binary exponential backoff scheme enables AMRR to work well for high latency systems. SampleRate uses aggressive probe packets to estimate the optimum transmission rate.

A different approach to multirate selection is presented by SNR-based algorithms such as RBAR (Receiver Based Auto Rate) [4]. In this solution, the receiver measures the SNR value of the received RTS and uses the CTS frame to inform the sender of the desired rate. This allows for very fast adaptability, but requires changes in the IEEE 802.11 standard and the constant use of the RTS/CTS mechanism.

A very efficient approach seems to be the OAR (Opportunistic Auto Rate) protocol [15]. It utilizes the coherence times of good channel conditions to send high-rate multi-frame bursts. This is similar to the TXOP (transmission opportunity) feature of 802.11e [8]. OAR has low overhead and can increase fairness in the network. However, it also requires changes to the IEEE 802.11 standard.

Despite many theoretical analyses of 802.11 performance, not much study has been done to measure interoperability performance between cards belonging to different vendors. A recent analysis can be found in [17] (closely related to the work done in [16]). The authors measure the performance of six 802.11b cards (in infrastructure mode) to determine whether they adhere to standards. Their main conclusion is that most of the unfairness between commercial cards is due to the hardware/firmware implementations, rather than channel properties. Furthermore, they state that cards belonging to the same vendor exhibit better fairness.

Garoppo et al. have presented an interesting comparison between analytical, simulation and experimental results for two 802.11b cards from different vendors [3]. Their results show high correlation between the modeled, simulated and measured values. However, they also notice a meaningful difference in the performance of the two cards in an infrastructure network.

Performance measurements of the saturation throughput [1] of five different IEEE 802.11b access points (APs) can be found in [14]. The upper bound of the AP throughput was considered. The three major observations are as follows. Firstly, an increase in the load offered to the AP's Ethernet interface does not always result in throughput increase. Secondly, for several APs, if the offered load exceeded their bridging capabilities they reduced their downlink throughput. Finally, better performance in certain directions was observed. The overall conclusion was that meaningful differences in the maximum saturation throughput exist for APs from different vendors.

3 Mathematical Model

The mathematical model derived in this section is based on work presented in [11] and [13]. The aim of this model is to obtain the theoretical upper bound estimation of the throughput in a multirate ad-hoc environment.

We consider a situation in which station i starts its transmission of a DATA frame of length l to station j at time t . The basic assumptions are that data frames are of equal length, no hidden stations appear and all data frame transmissions are

1 We define throughput as the ratio of the data transmitted in the link layer (including frame headers) to the time needed to deliver the data from one node to another.

independent. Furthermore, the MAC performance is only evaluated, pure DATA/ACK mode is assumed and all currently transmitting/receiving stations remain stationary.

Let us assume the following notation: A is the set of all stations in a BSS, N is the total number of stations in A, l is the length of the data frame, l_{ACK} is the length of the ACK frame (all measured in normalized time units). Other parameters are as follows: β is propagation delay, S is overall system throughput, T_S $\left(T_C\right)$ is expected time interval between periods when the channel is idle for a DIFS (Distributed Inter-Frame Space) period, within which at least one successful (collided) transmission took place.

A successful transmission must fulfill the following conditions: (a) the sender and the receiver stations are not hidden from each other, (b) no other station being within the range of the receiver starts its transmission within the time period $[t-\beta, t+\beta]$, (c) no other station being within the range of the sender receives any successful frame within the time period $[t-\beta, t+\beta]$.

Once a channel is sensed idle for a DIFS interval, time needed for the data frame destined to station j to be generated at station i is assumed to be exponentially distributed with a rate λ or $G(i, j)$ (equivalent terms). As a consequence, the total rate for a common channel in a single BSS is $N(N-1)\lambda$ or $G=\sum_{i,j} G(i, j)$.

Simple observation shows that:

$$T_C \geq 1/G + l + DIFS + \beta \tag{1}$$

$$T_S = 1/G + l + SIFS + l_{ACK} + DIFS + \beta \tag{2}$$

where $1/G$ is the expected time until the beginning of a transmission of the first frame after the channel was sensed idle for DIFS.

Let us denote by $p_s(i, j|m, n)$, where $m, n \in A$, the probability of a successful data frame transmission from station i to station j only under the condition that, after a DIFS interval, a data frame transmission between stations m and n occurs. As a result, the effective lower bound estimation of the expected number of successful transmissions for a Poisson process can be given as follows:

$$p_s(i, j) = e^{-N(N-1)\lambda\beta} \tag{3}$$

The probability that station i starts its transmission to station j before the end of an idle period is $\frac{G(i, j)}{G}$. The probability that station i starts its transmission to

station *j* before β (after the idle period was interrupted by a transmission between stations *m* and *n*) is given by $\frac{G(m,n)}{G}\left(1-e^{-\beta G(i,j)}\right)$.

Let us denote by $S(i,j)$ the throughput between stations *i* and *j* and, because $l_{ACK}+SIFS \ll l$, let us assume that $T_C \approx T_S$. As a consequence we get:

$$S(i,j) \approx \frac{p_s(i,j)\frac{G(i,j)}{G}+p_s(i,j)\left(1-e^{-\beta G(i,j)}\right)\sum_{(m,n)}\frac{G(m,n)}{G}}{T_S} = \tag{4}$$

$$= \frac{p_s(i,j)\frac{1}{N(N-1)}+p_s(i,j)\left(1-e^{-\beta\lambda}\right)\frac{N(N-1)-1}{N(N-1)}}{1/G+l+SIFS+l_{ACK}+DIFS+\beta}$$

Denoting the overall upper bound of the system throughput as $S=\sum_{(i,j)\in A} S(i,j)$ we get:

$$S = N(N-1)\frac{p_s(i,j)\frac{1}{N(N-1)}+p_s(i,j)\left(1-e^{-\beta\lambda}\right)\frac{N(N-1)-1}{N(N-1)}}{1/G+l+SIFS+l_{ACK}+DIFS+\beta} = \tag{5}$$

$$= \frac{p_s(i,j)+p_s(i,j)\left(1-e^{-\beta\lambda}\right)N(N-1)-1}{1/G+l+SIFS+l_{ACK}+DIFS+\beta}$$

4 Measurement Scenarios

The measurements of the performance and interoperability of 802.11b/g wireless cards from different vendors were carried out in usual office conditions. The tested cards were: Linksys WPC-11, Lucent Silver PC24E, and Proxim 8480-WD. All cards worked in ad-hoc mode. Their output power was set to 30mW. The card vendors do not provide information on the type of multirate algorithms used.

In the considered scenario, the testbed consisted of three homogenous stations (see Fig. 1): one FTP server (Station C) and two clients (Stations A and B). Both clients, when connected to the server, began downloading a 1GB file what allowed to capture more than 50 thousand FTP frames transmitted from the server to the clients.

Station B was mobile. It increased its distance from the server. Station A was stationary. All measurements were performed in three different points marked in Fig. 1 by triangles. The aim of this experiment was to determine, whether the increasing distance of Station B would impact the multirate capabilities of Station C, i.e., whether the transmission from the server to Station A would be influenced.

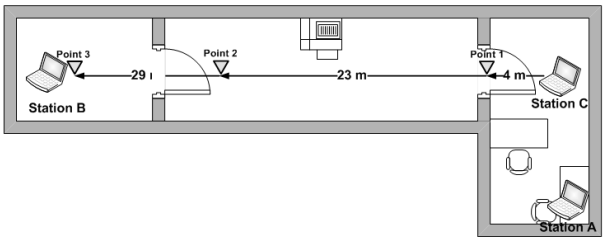


Fig. 1. Testbed

All possible sources of interference in the 2.4 GHz and 5 GHz bands (e.g., access points or Bluetooth devices) were eliminated for all experiments.

5 Measurement Results

From all the acquired results, we have decided to present two case scenarios, which serve as an illustration for certain important findings. It is important to keep in mind that since the clients were downloading data from the FTP server, the vast majority of the analyzed data are the DATA frames sent by the server and the ACK frames it received in return. Therefore, the results show how the server behaved (in terms of rate selection) when simultaneously communicating with two clients.

The first scenario consisted of a Lucent server (i.e., Station C was using a Lucent card, c.f. Fig. 1) and two Linksys clients (Stations A and B) communicating in the 802.11b standard. The results are presented in the form of graphs in Fig. 2 and Fig. 3. The first graph shows the percentage of all frames at a given measurement point, the second graph – the percentage of all data.

In Fig. 2 it can be clearly seen that while Station A (the stationary one) received a constant rate of 11 Mbps for its DATA frames, Station B's transmission rate dropped successively the further it was from the server.

Fig. 3 is a demonstration that, in terms of actual bytes sent, the overall share of the ACK frames is extremely small when compared to the DATA frames. Therefore, it would be impossible to observe the rate changes of ACK frames in this figure.

The second scenario proved to be more complex in terms of the data rates used. The server used a Proxim card, the stationary client – a Linksys card, and the moving client – a Proxim card as well. The results are shown in Fig. 4 and Fig. 5 (for the percentage of all frames and all data, respectively). The cards were operating in the 802.11g standard, which allows for a wide range of transmission rates (up to 54 Mbps). The first observation from the presented figures is that the Proxim card present at the server was using the basic rate (1 Mbps) to send its DATA frames to the Linksys client. This occurred despite the fact that the Linksys card was returning ACK frames in multiple rates (up to 11 Mbps). The reason for this is most likely vendor incompatibility. On the other hand, the Proxim client established a high speed link with the Proxim server. Both the DATA and ACK frames were able to utilize the potential of multiple transmission rates. At the first measurement point, the majority of DATA frames were sent with the highest available speed (54 Mbps), whereas all the ACK frames were sent at a speed of 24 Mbps. In the third measurement point up to 8 different rates were used (depending on radio conditions).

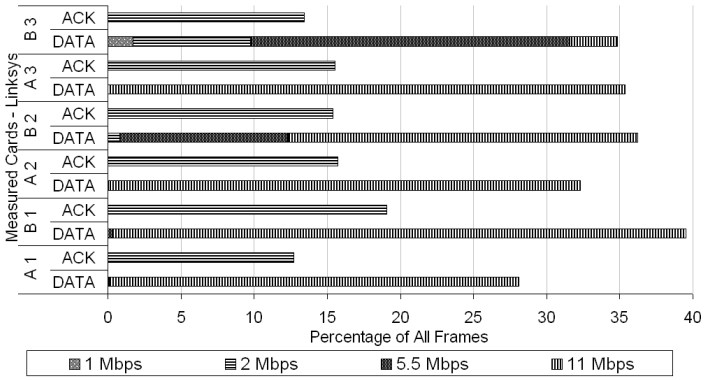


Fig. 2. Multirate performance of two Linksys cards (A and B) at three measurement points (1, 2, and 3): transmission speed vs. percentage of frames sent (at a given measurement point). The server was using a Lucent card

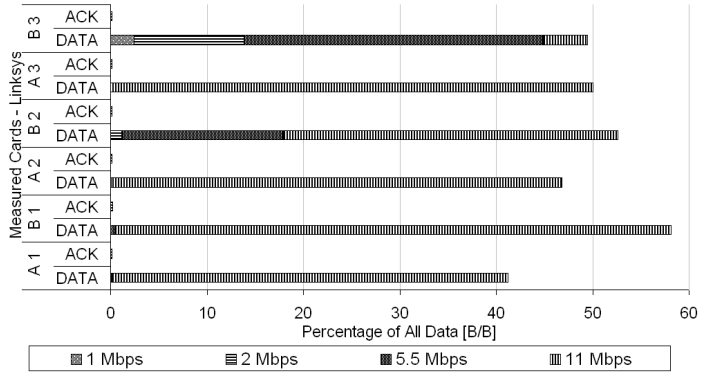


Fig. 3. Multirate performance of two Linksys cards (A and B) at three measurement points (1, 2, and 3): transmission speed vs. percentage of data sent (in bytes/bytes, at a given measurement point). The server was using a Lucent card

The fact that the Linksys card was transmitting at a rate of 1 Mbps means that it was underusing the channel and, therefore, degrading overall network performance. This is an example of how vendor incompatibility can lead to unfairness in the shared radio channel.

Comparing the two scenarios, we can see that in the first one (with 802.11b cards), the ACK frames were sent at a constant, basic rate of 2 Mbps. However, in the second scenario (with 802.11g cards) multiple rates were used. Based on these measurements it seems that in the 802.11b standard ACK frames are transmitted at a rate no large than 2 Mbps, whereas in 802.11g much higher rates can be used (up to 24 Mbps).

Furthermore, we can see that the rate of the mobile station does not impact the established rate of the stationary one. This means that near and far stations can coexist with multiple rates.

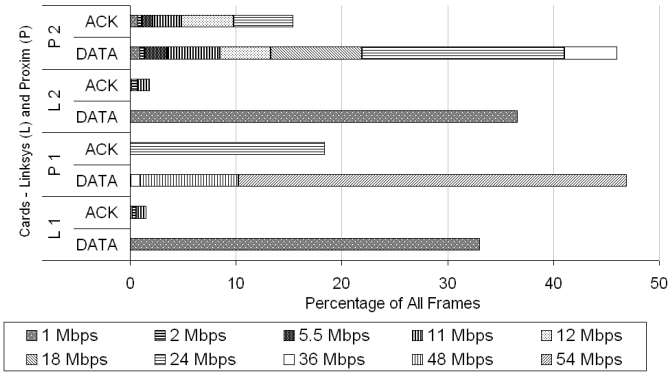


Fig. 4. Multirate performance of a Linksys (L) and Proxim (P) card at two measurement points (1 and 2): transmission speed vs. percentage of frames sent (at a given measurement point). The server was using a Proxim card

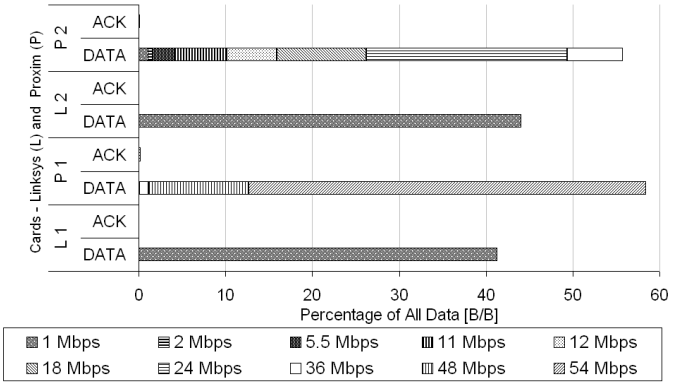


Fig. 5. Multirate performance of a Linksys (L) and Proxim (P) card at two measurement points (1 and 2): transmission speed vs. percentage of data sent (in bytes/bytes, at a given measurement point). The server was using a Proxim card.

6 Result Validation

In order to evaluate the obtained link layer throughput, we have compared them with theoretical values derived from the analytical model presented in Section 3. This comparison is presented in Table 2.

In order to take into account the use of multiple rates by the station, the theoretical value of the system throughput was calculated for each available rate and then summed up using a weighted average (based on bytes transmitted with a given rate). The DATA frame length l was taken as a weighted average of all transmitted DATA frames.

For the first scenario (Lucent server, Linksys clients), the measured results quite closely resemble the theoretical calculations. In this scenario, not many transmission

rates were used and we believe this is the reason why the results are similar. This is also further validation of our mathematical model.

In the second case, however, the number of rates used was much larger and the difference between theoretical and measured values is quite significant. This is because our model did not take into account the procedures needed to change the rate and the impact of lost frames. This is why the measured values were much lower than the theoretical ones.

Table 2. Comparison of Theoretical and Achieved Throughput

Point	FTP Server	Receiving Station	Throughput [Mbps]		Difference [%]
			Theoretical	*Measured*	
1	Lucent	Linksys A	4.99	5.16	3.4
	Lucent	Linksys B	4.96	5.03	1.4
2	Lucent	Linksys A	4.97	4.72	0.75
	Lucent	Linksys B	4.15	3.80	8.4
3	Lucent	Linksys A	5.01	4.11	17.96
	Lucent	Linksys B	2.35	2.35	0
1	Proxim	Linksys	0.51	0.004	99.2
	Proxim	Proxim	21.64	3.66	83.1
2	Proxim	Linksys	0.51	0.005	99.0
	Proxim	Proxim	9.13	1.95	78.6

7 Conclusions

The behavior of IEEE 802.11b/g cards in multirate ad-hoc environments has been presented in this paper. Certain popular and widely available WLAN cards from different vendors were tested in terms of throughput and interoperability. Both the measurements and analytical results were compared.

The following general conclusions can be formulated. First of all, the obtained results show the inefficiency of multirate algorithms used in commercial cards. Secondly, it can be observed that cards of the same model from one vendor co-operate much better. If the number of used rates grows significantly (which is possible for 802.11g), the achieved throughput drastically decreases. This is because cards spend time adjusting to the channel conditions by trying to find the appropriate rate.

The differences between the ideal, theoretical and measured results (as exemplified in the second scenario) can be 1000-fold. Therefore, there is a strong need to develop new, efficient multirate algorithms. Most importantly, adequate agreements between different vendors are required to improve the co-operation of WLAN devices, especially since multirate IEEE 802.11b/g combo cards dominate the market.

Disclaimer. This work has been carried out under the EU-funded NoE CONTENT project no. 038423.

References

1. B. Awerbuch, D. Holmer and H. Rubens, *High throughput route selection in multi-rate ad hoc wireless networks,* Technical report, Johns Hopkins University, Computer Science Department, 2003.
2. J. Bicket, *Bit-rate Selection in Wireless Networks,* Master's thesis, MIT, 2005.
3. R. G. Garroppo, S. Giordano and S. Lucetti, *IEEE 802.11 b performance evaluation: convergence of theoretical, simulation and experimental results,* Telecommunications Network Strategy and Planning Symposium. NETWORKS 2004, 11th International, 13-16 June 2004.
4. G. Holland, N. H. Vaidya and P. Bahl, *A rate-adaptive MAC protocol for multi-hop wireless networks,* Mobile Computing and Networking, 2001.
5. IEEE 802.11 *Standard for Wireless LAN: Medium Access Control (MAC) and Physical Layer (PHY) Specification,* New York, IEEE Inc., 1999.
6. IEEE 802.11a: *High-Speed Physical Layer in the 5 GHz Band,* New York, IEEE Inc., 1999.
7. IEEE 802.11b: *High-Speed Physical Layer Ext. in the 2.4 GHz Band, New York,* IEEE Inc., 1999.
8. IEEE 802.11e: *Medium Access Method (MAC) Quality of Service Enhancements,* New York, IEEE Inc., 2005.
9. IEEE 802.11g: *Further Higher-Speed Physical Layer Extension in the 2.4 GHz Band,* New York, IEEE Inc., 2003.
10. M. Lacage, M. H. Manshaei and T. Turletti, *IEEE 802.11 Rate Adaptation: A Practical Approach,* INRIA Research Report, No 5208, 2004.
11. J. B. Lee and A. Elftheriadis, *A Performance Analysis for the IEEE 802.11 Wireless LAN MAC Protocol,* IEEE Trans. on Networking, March 1998.
12. Madwifi: *Multiband Atheros Driver for WiFi,* `http://madwifi.sourceforge.net/` September 2005.
13. M. Natkaniec, *A Performance Analysis of the IEEE 802.11 DCF Protocol,* PGTS 2002, 23-24 September 2002.
14. E. Pelletta and H. Velayos, *Performance Measurements of the Saturation Throughput in IEEE 802.11 Access Points,* Proceedings of the Third international Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks, 04-06 April 2005.
15. B. Sadeghi, V. Kanodia, A. Sabharwal, and E. Knightly, *Opportunistic media access for multirate ad hoc networks,* September 2002.
16. Di Stefano, A. Scaglione, G. Terrazzino, I. Tinnirello, V. Ammirata, L. Scalia, G. Bianchi and C. Giaconia, *On the Fidelity of IEEE 802.11 Commercial Cards,* Proceedings of the First international Conference on Wireless internet, 10-15 July 2005.
17. Di Stefano, G. Terrazzino, L. Scalia, I. Tinnirello, G. Bianchi and C. Giaconia, *An Experimental Testbed and Methodology for Characterizing IEEE 802.11 Network Cards,* in Proceedings of the 2006 international Symposium on World of Wireless, Mobile and Multimedia Networks, 2006.

Proceedings of the International Multiconference on
Computer Science and Information Technology, pp. 975 – 984
ISSN 1896-7094
© 2007 PIPS

Self-organizing Ad-hoc Architecture as a Solution for Coverage Issues in WiMAX Metropolitan Area Networks

Krzysztof Gierłowski, Józef Woźniak, Krzysztof Nowicki

{gierk, jowoz, know}@eti.pg.gda.pl

Abstract. New WiMAX technology offers several advantages over currently available (GSM or UMTS-based) solutions. It is a cost effective, evolving, and robust technology providing quality of service guarantees, high reliability, wide coverage and non-line-of-sight transmission capabilities. All these features make it especially suitable for densely populated - urban environments. In the paper we discuss design and implementation difficulties concerning network coverage, discovered in the test-bed implementation during measurements and tests. We point out unexpected "coverage white spots" - not characteristic to WiMAX technology. As one of possible solutions of this significant drawback of the very promising technology we consider reconfigurable mesh organization of WiMAX base stations. We also suggest directions for further development of this kind of network operation, partly based on our practical experience.

Introduction

The new WiMAX technology, based on IEEE 802.16 standard [1], offers reliable, fast and Quality of Service (QoS) aware transmissions over significant distances. WiMAX technology provides both line-of-sight (LOS) and no-line-of-sight (NLOS) solutions. The LOS solution allows transmissions with rates over 70 Mbps over distances up to 50 kilometers (or even more), as long as antennas of both devices have straight (not shaded) view of each other. The second one supplies a connectivity using reflected signals when a path between antennas is shaded by various obstacles. In such case the range is limited to about 5 kilometers only. The technology supports different modulation and coding schemes coupled with their adaptive adjustment of transmission parameters in order to maximize stable coverage area.
Other strong advantages of WiMAX systems include high security, reliability and integrated QoS support, which jointly allow operators to guarantee their users a required level of network services.
The most popular WiMAX system architecture follows a point-to-multipoint (PtMP) data communications model with a coordinating base station (BS) and participating client terminals (subscriber stations - SSs). The standard also specifies a foundation of a mesh-mode in which peer stations participate in self-organizing network structure and/or its connectivity.

All these characteristics make WiMAX an economically appealing solution. Specially, the NLOS capability described above, makes WiMAX a key technology for urban environments, allowing for a relatively small number of BSs working in PtMP mode to cover large areas. However at the same time its complexity significantly

complicates system design, particularly in terms of system coverage prediction and client station capabilities in a given location, with respect to available throughput and specific QoS parameters.
Due to these difficulties there is a need for specialized software tools able to help system designers to assess effective coverage area and exact transmission parameters inside of it.

In the following part of the paper we discuss basic theoretical wireless network design models, pointing out approaches proposed by them in case of NLOS capable WiMAX technology. Next, we describe measurements conducted in our WiMAX test-bed installation which uncovered the "coverage white spots" effect – a phenomen very disadvantageous to nomadic and mobile users. This previously undocumented effect strongly undermines the popular beliefs concerning excellent performance of WiMAX NLOS mechanisms and can significantly raise cost of network deployment. As such state of affairs is difficult to accept, steps must be taken to solve the problem. So, in the final part of the article we describe our novel solution of this difficulty - a self-organizing ad-hoc WiMAX-based mesh architecture as a solution for coverage issues. Finally, we also propose mechanisms necessary for the mesh mode to effectively counter the "coverage white spots" in case of nomadic and mobile networks.

Theoretical coverage models

In all types of wireless systems, including WiMAX, prediction of their coverage area is a very challenging task, especially when we want to mark out the coverage with a required accuracy. Because of this there is a need for methods to verify provisional results, obtained via theoretical calculations. There are two basic methods used for current design practices. The first one requires a test-bed installation and depends entirely on empirical measurements. The second one includes software tools able to estimate system coverage with use of one of available propagation models. As the first method is rather time consuming and costly, the software tools are widely used to support coverage calculation for wireless systems, such as short range local area systems (WLANs) or more complex wide area networks, consisting of multiple BSs (WMANs, WWANs).

There are two basic types of propagation models employed in wireless systems design [2,3]:

- empirical (or statistical) models, which are based on a stochastic analysis of series of measurements conducted in the area of interest. They are relatively easy to implement but not very sensitive to environment's geometry,
- site-specific (or deterministic) models, which are far more accurate and do not need signal measurements. However, they require huge amounts of data concerning environment geometry, terrain profile, etc. and high computational efforts.

WiMAX systems should usually provide effective coverage in highly urbanized environments. Following this requirement we are mostly interested in deterministic models, as they can give us results sufficiently accurate for such areas.

Of course, there is always a theoretical possibility to calculate exact propagation characteristics solving sets of Maxwell's equations. However, this method would require very complex data processing and very high computational power, causing such solution to be very inefficient. Due to this fact, current software tools, based on a deterministic propagation model, usually employ simplified simulations: mainly ray-tracing or ray-lunching techniques, based on Uniform Geometrical Theory of Diffraction (UTD). [2]. Such approach enables significant simplification in calculations, making the model an efficient design tool, but with a loss of accuracy.

WiMAX technology coverage characteristics differ significantly from other wireless network technologies that are employed in similar environments (mainly due to its NLOS capability – see fig. 1), thus a dedicated software model is required to give exact results [4].

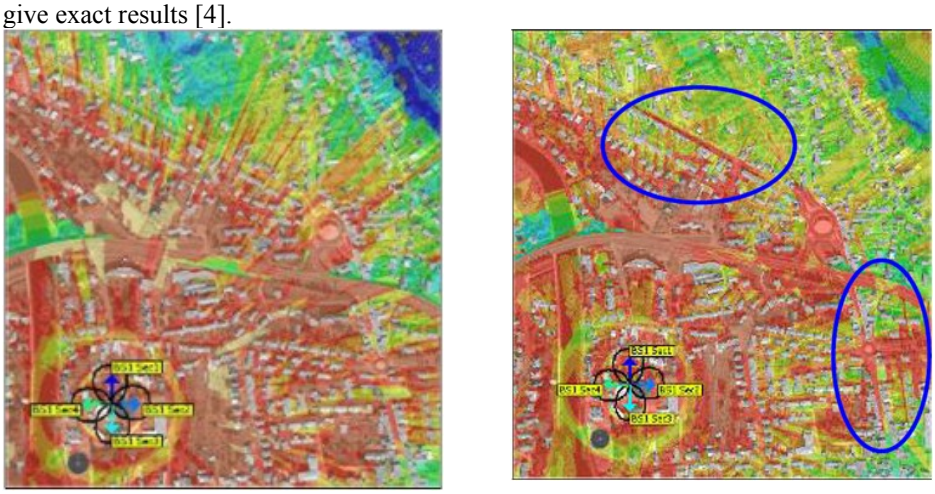


Fig. 1. Results of WiMAX BS coverage simulation: general model (left) and specialized WiMAX coverage model (right).

Regardless of employed theoretical models and their accuracy, experience in wireless systems design and implementation suggests a necessity of conducting empirical measurements in order to confirm that the system design and theoretically obtained parameters are correct [5]. In accordance with a good design practice we implemented a test-bed installation of WiMAX with one base station, and conducted extensive measurements and tests of its coverage and transmission parameters [6].

Test-bed installation and example measurements

From both modeling procedures and hardware manufacturers' tests it follows that WiMAX technology is indeed very well suited for metropolitan environment and generally offers good coverage, even in highly urbanized areas [7]. To verify these statements and prove accuracy of available software design tools as well as to gather practical design experience, we prepared a study test-bed installation consisting of a single WiMAX BS located at Gdansk University of Technology.

We employed a BreezeMAX Micro Base Station [8] provided by Alvarion company and using 3.5 GHz licensed frequency band.
We also developed a dedicated software package consisting of a number of control and monitoring tools. They communicate with BS, client terminals, GPS receivers and are able to automate the experiments to a significant degree. They also handle a real-time, initial data analysis to help measurement teams to optimize their work.

One of our main points of interest was the coverage of WiMAX services in a densely populated metropolitan environment. We performed a variety of tests including:

- measurements of BS signal strength in physical layer,
- modulation and coding profile usage as function of signal quality,
- efficiency of transmission in media access control layer (BER, PER),
- quality of service contract adherence for transport layer services.

The tests were performed with use of hardware spectrum analyzer, equipped with an omnidirectional antenna, BreezeMAX PRO BS, SS subscriber stations (PRO and Si models) [8] and transmission performance counters of the base station.

Overall test results confirmed that, in case of LOS, using the equipment mentioned above, we could expect a reliable communication up to 30 km, and 5 km in majority of cases related to NLOS scenarios [6].

Such general statements sound promising. However, we also made quite unexpected observations. It turned out that in case of NLOS communication the network did not cover entirely the tested area. We were able to find some points not covered by our BS ("coverage white spots"). In some cases, using WiMAX specific propagation models, it would be possible to predict such areas - taking into account terrain profiles and buildings layout.

At the same time, we also detected that there are many locations at which the measured coverage (signal parameters) differs significantly from theoretical estimates. In some places the coverage was a result of repeatedly reflected signals or signals reflected by various objects either improbable or difficult to map, like trees, billboards, trains, trucks, etc. Other places showed a lack of the coverage despite of relatively minor obstacles between BS and a client terminal, thus creating coverage white spots (fig. 2). Our measurements also showed that even a very small displacement (20 m horizontal and/or 3 m vertical) of a client station can result in a dramatic degradation of the transmission parameters - from the best possible modulation and coding profile (QAM64 3/4) to the complete loss of connectivity. This effect makes a WiMAX system design a very difficult task, requiring empirical measurements to validate the project.



Fig. 2. WiMAX coverage hole effect – measured coverage of WiMAX base station in 3.5 GHz band. Stable modulation profile chosen by client terminal: white X - no communication, "1" – BPSK, "2" – QPSK, "3" – QAM16, "4" – QAM64 1/2, "5" – QAM64 3/4).

The described "coverage white spots" effect shows difficulties in precise prediction of a real system coverage and system parameters. If a service provider is interested in a complete and continuous coverage of an area it can lead to higher system deployment costs. Also, there is no efficient way to validate this kind of coverage.
In case of a mobile operator the problem becomes even more serious, because mobile terminals can loose and regain connectivity as they move. Such effects can be especially laborious in WiMAX, because in this technology each network entry procedure is complicated and consumes significant network (bandwidth, BS processing power) and client terminal (battery) resources.
Summarizing measurement results we can state, that while WiMAX NLOS capability indeed makes it fit for highly urbanized areas, it is not without disadvantages and requires a careful design and troublesome practical validation.

Network design considerations

In the case of simple WiFi (IEEE 802.11 [9]) - wireless local area network - installations we use test measurements to appropriately design a system. Its usual 50-300 m range makes such approach possible. In case of WiMAX technology, where the

range is counted in kilometers such solution seems to be highly impractical, as it is almost impossible to compile a full, empirical coverage map, by measuring all meaningful points within system's range. In a dense metropolitan area with WiMAX NLOS capability we would have to measure an extremely thick layout of measurement points. Also as we pointed out in the previous section we would need a resolution of under 20 m horizontally. Moreover we should not limit such measurements to a 2-dimensional case, because there are significant variations of effective signal strength related to a client station vertical placement , especially prominent near ground level.

Computational propagation models can help us in highlighting potential trouble-spots and suggest important measurement points. They offer a great support during the design process. Regrettably, their application can be costly, because they usually require detailed 3-dimensional digital maps, which may be expensive or even unavailable for the area of interest [10].

Furthermore, commercial products, based on ray-tracing and ray-launching models [2], are not well suited to detect coverage anomalies as small as the described "coverage hole effect". Our research shows that in order to detect them we must employ a very high resolution of modeling – often higher than popular 3-dimensional map resolution [3]. In case of modeling at such resolution, the simplifications common for these models no longer work in our favor. This leads to the need for much higher computational power and longer modeling time and still does not guarantee detection of all significant anomalies.

Wherever we are able or not to detect the coverage holes, to provide consistent coverage of the area we need much higher number of BSs than we can expect from theoretical modeling. Also, in many places such coverage holes are almost impossible to eliminate without installation of economically impractical number of BSs.

There are at least two basic approaches that could be proposed as possible solutions to this problem:

- heterogonous approach - a number of different connectivity technologies are used in order to provide services for the user,
- homogenous approach - only one wireless technology, i.e. WiMAX is employed.

Currently there is a strong trend towards creation of heterogeneous systems, where users can use a variety of connectivity technologies [11].

The emerging IEEE 802.21 standard [12] is devoted to a seamless handover between networks of the same or different types. In this case the best connection (ABC strategy – Always Best Connected) is automatically selected at a given user location and the handover is performed without losing quality of service, if possible [13].

This approach takes into account several different wireless technologies and we will not consider it here. We will concentrate on the homogenous approach, limited to WiMAX technology, considering WiMAX mesh architecture, as a promising solution for coverage issues.

WiMAX-Mesh mode

In WiMAX-Mesh mode, there is no prominent BS, but SSs communicate directly with their neighbors forming a dynamic, self-organizing, multi-hop network. In this way a client station does not need to be in range of one of relatively few BSs, but it is sufficient to be in range of any other participating client station and number of these devices is usually much higher (fig. 3). Moreover, with correctly designed control protocols and effective methods of joining the network by new stations, its available capacity can be increased instead of going down.

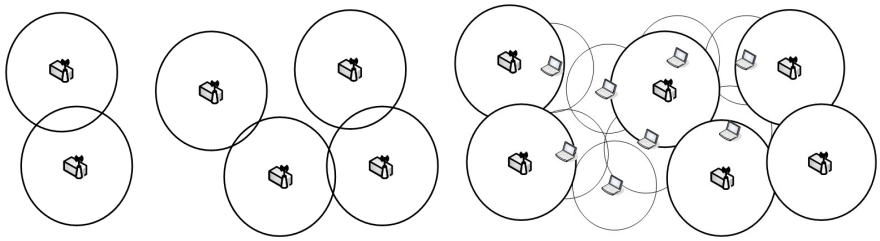


Fig. 3. Coverage comparison of PtMP and mesh mode with equal number of operator provided base stations/nodes.

Unfortunately such network architecture requires much more advanced support mechanisms than a simple PtMP setup, where a single entity (BS) sees and controls all the network activity. In case of wireless ad-hoc mesh architecture, these mechanisms (medium access control, security…) have to be significantly extended and be able to operate in a distributed environment. Also new mechanisms (listed below) not required in PtMP setup (which utilizes star architecture), are necessary:

- Topology control – selects logical network node neighborhood based on its physical neighborhood. Running in all network nodes, it is responsible for overall network topology and vast number of derived characteristics (path lengths, bandwidth available, network capacity, transmission delay, error rates…).
- Route discovery – set of mechanisms able to find a route through network nodes to any required destination within and outside wireless mesh. In case of WiMAX, it should be able to provide paths that able to provide specified QoS guarantees.
- Data forwarding – responsible for retransmitting received traffic addressed to remote nodes, with accordance to routing information obtained from discovery mechanisms and QoS guarantees.

In majority of research works and test implementations a mesh network utilizes short range wireless technologies (WLANs or sensor networks) to ensure wide area coverage and high reliability. We claim that, also in case of a wireless metropolitan network (WMAN) of much higher basic range WiMAX-based ad-hoc mesh architecture can provide required functionalities and become practical and economically viable solution [14].

Due to relatively high complexity the WiMAX-mesh mode is not yet specified in the IEEE 802.16 standard and there are still ongoing research and design works. This fact (lack of detailed specification) gives us a possibility to incorporate into the created

standard new mechanisms which will make 802.16 especially attractive for metropolitan environments - being definitely its prime areas of deployment.

Our research related to IEEE 802.16 and measurements in the test-bed shows that mesh architecture based on WiMAX metropolitan area network is likely to solve the coverage-hole problems, as long as sufficient number of client stations will participate in the network. In such case it will provide much better terrain coverage than reasonably designed PtMP WiMAX installation. Also the costs of infrastructure will be significantly lower.

We predict that in metropolitan environment the number of client station will not pose a problem, with currently observable user demand and manufacturers' support for the technology. It should also be possible to keep prices of mesh-capable subscriber stations similar to classical PtMP WiMAX terminals.

WiMAX-based ad-hoc mesh network can provide much better terrain coverage and its development costs are significantly lower than in corresponding coverage scenario supported only by classical BSs in PtMP mode. Of course, there is still need for a number of operator-provided network nodes as a foundation of the network.

Moreover, mesh architecture can provide high reliability due to high number of redundant network devices, wireless links and paths to most destinations. It will also scale well, because any participating node brings additional resources to an overall network poll.

The coverage hole problem can be solved by sufficiently dense network of mesh nodes and their redundant links, but network control mechanisms need to be able to deal with rapidly changing mesh topology. In a static mesh configuration (as in static PtMP scenario), the problem is not dangerous, as in most cases the placement of network nodes can be optimized, but in case of mobile mesh nodes even small movement (as mentioned before) can lead to unpredictable breakdowns and reappearances of inter-node links.

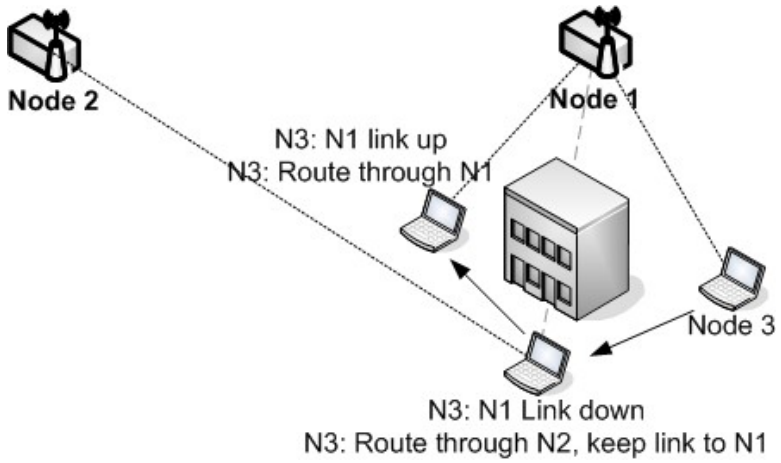


Fig. 4. Short-period link losses and possible solution with redundant path routing and coverage hole aware topology control.

Such environment greatly lowers efficiency of network mechanisms leading to, for example, frequent activation of ad-hoc routing protocol's discovery mechanisms, which flood the network with control traffic. Also QoS guarantees are extremely difficult to maintain in such environment as fast and frequent, short-term connectivity losses can occur.

Fortunately massive redundancy which exists in a sufficiently dense mesh network can be used to offset the effect without losing transmission reliability and QoS guarantees. Furthermore, our observations and measurements tell us, that signal losses and corresponding link breakdowns caused by coverage hole effect are mostly short term events and as such can be efficiently countered with properly designed network control mechanisms.

Our present research leads us to believe, that efficient topology control, taking into account possibility of short-time disappearance of network links, coupled with redundant path routing and stability-aware routing metric can solve the described problem (fig. 4). We are currently working on a simulation model of WiMAX-based self-organizing mesh network, resistant to the topology stability issues, described above.

Conclusions

Based on our theoretical research and practical experiments, we observed a possibly dangerous effect present in wireless networks based on WiMAX technology, resulting in small coverage holes in areas of otherwise good coverage. Such white spots are difficult to predict, even with the use of deterministic propagation models, which are amongst the most popular wireless network design support tools, used today.

Such situation can lead to lower than expected service level, requiring repositioning or installation of additional hardware in case of stationery users and can be especially harmful for mobile users that will experience periodic losses of connectivity.

Despite the fact that the same coverage problem would affect mesh nodes (especially in case of mobile nodes) potentially leading to topology instability, it is possible to design network control mechanisms to counter the effect. That would allow WiMAX mesh networks to provide continuous coverage of a given area eliminating "coverage holes", which is very difficult in classical BS-based architecture without large additional hardware costs.

We propose WiMAX mesh networks as viable method of dealing with coverage difficulties in metropolitan areas as they are simultaneously providing additional crucial advantages, such as: well-scaling high network capacity, high reliability based on multiple redundancy, low cost of deployment etc.

References

1. IEEE, IEEE 802.16-2004: *IEEE Standard for Local and Metropolitan Area Networks Part 16: Air Interface for Fixed Broadband Wireless Access* Systems, 2004.
2. T. K. Sarkar at al., *A Survey of Various Propagation Models for Mobile Communication*, IEEE Antennas and Propagation Magazine, Vol. 45, No. 3, June 2003.
3. J.-P. Rossi, Y. Gabillet, *A mixed ray launching/tracing method for full 3-D UHF propagation modeling and comparison with wide-band measurements,* IEEE Transactions on Antennas and Propagation, Vol. 50, Issue 4, p. 517 – 523, April 2002.
4. ATDI, The WiMax technologies with ICS telecom: Fixed, Nomadic and Mobile!, `http://www.adti.com/` 2007.
5. R. Olejnik, *Methods of designing wireless networks and parametric design method (in Polish)*, Computer Networks – T. 1: New technologies, p. 347-356, Wydawnictwo Komunikacji i Łączności, Warszawa 2007.
6. K. Gierłowski, K. Nowicki, *Test Installation of WiMAX-based Wireless Network on Gdansk University of Technology (in Polish)*, Multimedia in economy and education. Ed..L. Kiełtyka. DIFIN, Warszawa 2006.
7. WiMAX Forum, *WiMAX Forum Whitepapers*, `http://www.wimaxforum.org` 2007.
8. Alvarion, *BreezeMAX Whitepapers*, `http://www.alvarion.com`, 2006.
9. IEEE, IEEE 802.11-1999: *IEEE Standard for Information Technology – Telecommunications and Information Exchange between Systems – Local and Metropolitan Area Networks – Specific requirements – Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications*, 1999.
10. M. T. Hewitt, *Surface feature data for propagation modeling, IEEE Colloquium on Terrain Modelling and Ground Cover Data for Propagation Studies*, p. 4/1 - 4/6, 1993.
11. P. Matusz, J. Woźniak, K. Gierłowski, T. Gierszewski, T. Klajbor, P. Machań, T. Mrugalski, R. Wielicki: *Heterogeneous wireless networks - selected operational aspects*, Telecommunication Review and Telecommunication News No 8-9/2005, pp. 284-295.
12. IEEE, IEEE P802.21/D05.00: *Draft IEEE Standard for Local and Metropolitan Area Networks: Media Independent Handover Services*, 2007.
13. P. Machań, A. Jordan, A. Wieczorkowska, J. Woźniak, *IEEE 802.21 Standard – mobility support mechanisms,* Telecommunication Review and Telecommunication News, No 7/2007, p. 211-216.
14. K. Gierlowski, K. Nowicki, *Wireless Networks as an Infrastructure for Mission-Critical Business Applications*, Network Business Information Systems, Regensburg, Germany 2007.

Proceedings of the International Multiconference on
Computer Science and Information Technology pp. 985–994
ISSN 1896-7094
© 2007 PIPS

Ad-hoc networking with low-cost devices: how to do it right

Pawel Gburzynski[1] and Wlodek Olesinski[2]

[1] University of Alberta, Department of Computing Science,
Edmonton, Alberta, CANADA T6G 2E8
[2] Olsonet Communications Corporation,
51 Wycliffe Street, Ottawa, Ontario, CANADA K2G 5L9

Abstract. Although simple wireless communication involving nodes built of microcontrollers and radio devices from the low end of the price spectrum is quite popular these days, one seldom hears about serious wireless networks built from such devices. Commercially available nodes for ad-hoc networking (somewhat inappropriately called "motes")[3] are in fact quite serious computers with tens of megabytes of RAM and rather extravagant resource demands. We show how one can build practical ad-hoc networks using the smallest and cheapest devices available today. In our networks, such devices are capable of sustaining swarm-intelligent sophisticated routing while offering enough processing power to cater to complex applications involving distributed sensing and monitoring.

1 Introduction

The wast number of academic contributions to the area of ad-hoc wireless networking has left a surprisingly tiny footprint in the practical world. For once, the industry is not much smarter. The problem is that the popular and acknowledged routing schemes, as well as programmer-friendly application development systems, require a significant amount of computing power and are not suitable for small and inexpensive devices. As an example of the latter, consider a key-chain car opener. A networking "node" of this kind is typically built around a low-power microcontroller with $\sim$ 1 KB of RAM driving a simple transceiver. The combined cost of the two components is usually below $5. While it is not a big challenge to implement within this framework a functionally simple broadcaster of short packets, it is quite another issue to turn this device into a collaborating node of an intelligent ad-hoc wireless system.

The plethora of popular ad-hoc routing schemes proposed and analyzed in the literature [14, 1, 13, 10, 8, 12, 16, 6], address devices with a somewhat larger resource base. This is because those schemes require the nodes to store and analyze a non-trivial amount of information to carry out their duties. Moreover, none of them provides for "graceful downscaling," whereby a node with a smaller

[3] See for example `http://www.xbow.com/`.

than "recommended" amount of RAM can still fulfill its obligation to the network, but, perhaps, at a reduced (albeit manageable and acceptable) quality. With such systems, hardware resources must be overdesigned (i.e., wasted in typical scenarios), as an overrun leads to a functional breakdown.

The most popular commercial scheme originally intended for building ad-hoc networks is Bluetooth. Its two fundamental problems are: 1) a large footprint and, consequently, non-trivial cost; 2) arcane connection discovery and maintenance options, which render true ad-hoc networking cumbersome. Even though some attempts are still being made to build actual ad-hoc networks based on Bluetooth [4], it is commonly agreed that the role of this technology is reduced to creating small personal area hubs. ZigBee® (based on AODV [15]) comes closer; however, despite the tremendous industrial push, it fails to catch on. The reason, we believe, is its isolation from the wider context of application development issues combined with the limited flexibility of AODV as a routing scheme.

If there is a place in the realm of low-end microcontrollers where the adjective "ad-hoc" is well applicable, it is to software development. Typically, the software (firmware) designed for one particular project is viewed as a "one-night stand," and its reusability, modularity, and exchangeability are not deemed interesting. This is because convenient, modular, and self-documentable programming techniques based on concepts like multi-threading, event handling, synchronization, object-orientedness are considered too costly in terms of resource requirements (mostly RAM) to be of merit in programming the smallest microcontrollers. Even TinyOS [9], which is the most popular operating system for networked microcontrollers, has many shortcomings in these areas. Moreover, its evolution (as it usually happens with systems driven by large communities and consortia), leans towards larger and larger devices.

Serious efforts to introduce an order and methodology into programming small devices often meet with skepticism and shrugs. The common misconception is that one will not have to wait for long before those devices disappear and become superseded by larger ones, capable of running Linux or Windows®. This is not true. Despite the ever decreasing cost of microcontrollers, we see absolutely no reduction in the demand for the ones from the lowest end of the spectrum. On the contrary: their low cost and low power requirements enable new applications and trigger even more demand.

In this paper, we outline our comprehensive platform for rapidly building wireless *praxes*, i.e., low-cost applications based on ad-hoc networking. This platform comprises an operating system, a flexible and auto-scalable ad-hoc forwarding scheme, and an emulator for testing the praxes in a high-fidelity virtual environment. We show how one can built well structured multithreaded programs operating within a trivially small amount of RAM and internetwork them in a truly ad-hoc fashion.

2 The operating system

The most serious problem with implementing multitasking within tiny RAM is the stack space, which must be rigidly pre-allocated to every task. Notably, this space is wasted from the viewpoint of the application: it is merely a "working area" needed to build the high-level structure of the program. One radical solution, e.g., adopted in TinyOS [7, 9], is simply to eliminate multitasking. Essentially, TinyOS defines two types of activities: event handlers and *tasks*, which are simply chunks of code that cannot be preempted by (and thus cannot coexist with) other tasks.

Our operating system, dubbed PicOS, does implement multitasking, whereby tasks explicitly declare preemption points in the form of *states*. A PicOS task looks like a finite state machine (FSM) that changes its states in response to events. The CPU is multiplexed among the multiple concurrent tasks, but only at state boundaries.

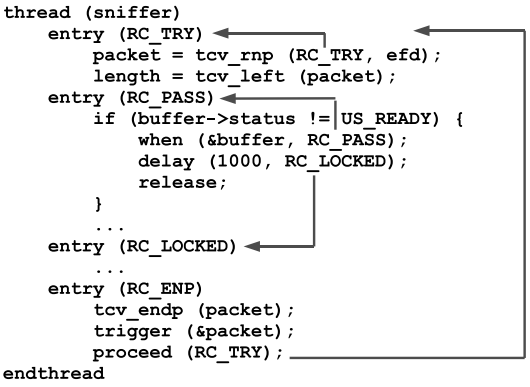


Fig. 1. A sample task (strand) code in PicOS.

For illustration, consider the sample task code shown in Figure 1. Its different states are marked by the `entry` statements. Whenever a task is assigned the CPU, its code function is called in the *current state*, i.e., at one specific entry point.

Task blocking always involves an explicit state indicating where the task should be resumed when the blocking condition disappears. For example, function `tcv_rnp` in state `RC_TRY` attempts to acquire an input packet for processing. If no packet is available, the function does not return: the process loses the CPU and will be resumed in state `RC_TRY` when a packet becomes available. The sequence in state `RC_PASS` (`when`, `delay`, `release`) exemplifies a more explicit state transition scenario. With `when`, the task declares that it wants to be resumed in state `RC_PASS` upon the occurrence of an event represented by the address of a data object (`buffer`). Such events can be signaled with `trigger`, as illustrated in state `RC_ENP`. The `delay` operation sets up a timer to go off after 1000

milliseconds, which will resume the task in state `RC_LOCKED`. Finally, `release` explicitly gives up the CPU: the task becomes blocked until any of the awaited conditions occurs.

The above paradigm of organizing tasks in PicOS has proven very versatile, expressive, structural, and self-documenting. The inherent FSM layout of the praxis comes for free and can be mechanically transformed, e.g., into a statechart [3], for easy comprehension or verification. Owing to the fact that a blocked task needs no stack space, all tasks in the system can share the same single global stack. The programmer-controlled preemptibility grain practically eliminates all synchronization problems haunting traditional multi-threaded applications.

So far, PicOS has been implemented on the MSP430 microcontroller family and on eCOG1 from Cyan Technology. The size of the task control block (PCB) needed to describe a single PicOS task is adjustable by a configuration parameter, depending on the number of events E that a single task may want to await simultaneously. The standard setting of this number is 3, which is sufficient for all our present applications and protocols. The PCB size in bytes is equal to $8 + 4E$, which yields 20 bytes for $E = 3$. The number of tasks in the system (the degree of multiprogramming) has no impact on the required stack size, which is solely determined by the maximum configuration of nested function calls. As automatic variables are not very useful for tasks (they do not survive state transitions and are thus discouraged), the stack has no tendency to run away. 96 bytes of stack size is practically always sufficient. In many cases, this number can be reduced by half.

3 Communication

Most routing protocols for ad-hoc wireless networks assume point-to-point communication, whereby each node forwarding the packet on its way to the destination sends it to a specific neighbor. Regardless of whether the scheme is proactive [14, 1] or reactive [13, 10, 8, 12, 16, 6], its primary objective is to determine the exact sequence of nodes to forward the packets from point A to point B. Despite the fact that the wireless environment is inherently multicast, this free feature is rarely exploited during the actual forwarding of session packets, although all protocols necessarily take advantage of broadcast transmissions during various stages of route discovery (e.g., the periodic HELLO messages broadcast by all nodes to announce their presence in the neighborhood). For example, in AODV [15], a node S initiating packet exchange with node D broadcasts a request to its one-hop neighbors to start the so-called path discovery operation. Based on its current perception of the neighborhood and cached information collected form previous path discoveries, a node receiving such a request may decide to forward it elsewhere, or, possibly, respond backward with a path information intended for the initiating node S. At the end, a single path between S and D has been established. A problem arises when the path is broken, because such a mishap effectively demolishes the entire delicate structure. When that happens, a new path discovery operation is essentially started from scratch.

On top of the susceptibility to node failures and disappearance, this generic approach requires the nodes to store a potentially sizable amount of elaborate routing information, which, typically, cannot be made fuzzy. For example, if a node is unable to store the identity of the next-hop neighbor for a particular session, then it will simply not be able to carry out its duties with respect to that session (thus breaking the path). It may confuse the network by offering a service that it is unable to deliver, resulting in stalled path discovery and, ultimately, communication failure.

The idea behind our solution, dubbed TARP (for Tiny Ad-hoc Routing Protocol), is to embrace fuzziness as a useful feature and to to take full advantage of the inherently broadcast nature of the wireless medium. Traditional schemes view this nature as a rather serious problem and try to defeat its negative consequences (hidden/exposed terminals) via MAC-level handshakes intended to facilitate point-to-point transmission [11]. TARP, in contrast, turns it into an advantage.

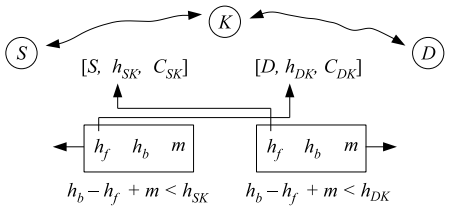


Fig. 2. The rule for selective packet discard (SPD).

Suppose that node S wants to send a packet to node D. With TARP, S simply transmits (broadcasts) the packet to its neighbors. A neighbor may decide to drop the packet (if it believes that its contribution to the communal forwarding task will not help) or retransmit it. This process continues until the packet reaches the destination D. An important property of this generic scheme (otherwise known as flooding) is that a retransmitted packet is never specifically addressed to a single next-hop neighbor. Needless to say, to make it useful, measures must be taken to limit the number of retransmissions to the minimum at which the desired quality of service is maintained. This part comes as a series of rules that determine when a node receiving a packet should rebroadcast it, as opposed to dropping. Some ideas for such rules are obvious, e.g., discarding duplicates of already seen packets and limiting the maximum number of hops traveled by a packet. The key to the success of our variant of flooding is the rule that brings the paths traveled by forwarded packets down to a narrow (but intentionally fuzzy) stripe of nodes along the shortest route.

Consider the three nodes shown in Figure 2. K is contemplating whether it should re-broadcast an "overheard" packet sent by S and addressed to D. Suppose K knows this information: h_b—the total number of hops traveled by some

packet that has recently reached S arriving from D (in the opposite direction); h_f—the number of hops traveled so far by the current packet; h_{DK}—the number of hops separating K from D. If $h_b < h_f + h_{DK}$, K can suspect that the packet can make it to D via a shorter path leading through another node. This is because, apparently, packets can make it from D to S in fewer hops than the combination of whatever the packet has already gone through with the hops it still must cover if forwarded via K. Thus, in such a case, K may decide to drop the packet.

The requisite information can be collected from the headers of packets that K overhears as the session goes on. To make it possible, the packet header should carry the current number of hops traveled by the packet as well as the number of hops traveled by the last packet that arrived at the sender from the opposite end. As a duplicate packet is always discarded at the earliest detection, a non-duplicate packet arriving at a destination makes it along the fastest (and usually the shortest) path. Until the network learns about a particular session (understood as a pair of nodes that want to communicate), the forwarding for that session may be overly redundant. We assume that a node that does not have enough information to make an educated decision, forwards the packet (unless the obvious rules say otherwise).

Note that this mode of operation is not fundamentally less economical than one based on point-to-point forwarding (in a traditional ad-hoc routing scheme). This is because before a point-to-point system can commence forwarding, it has to learn the routes one way or the other. In a traditional scheme, there are distinct phases of knowledge acquisition and recovery from its decay, which are separate from the actual forwarding. In our approach, the acquisition and maintenance of the requisite knowledge happens together with forwarding (as its byproduct) and acts constantly to preserve and improve its quality.

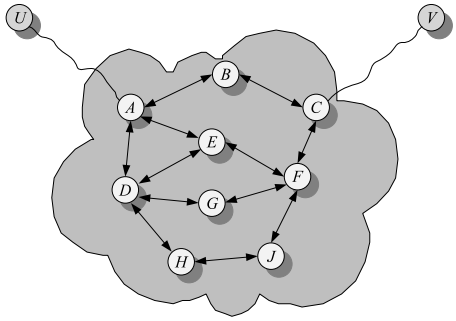


Fig. 3. A smooth handoff in TARP.

Owing to the inherent imperfections of the ad-hoc wireless environment, K should not be too jumpy with negative decisions. TARP includes a slack parameter m in the inequality (as shown in Figure 2). When $m > 0$, the rule will allow

the node to forward the packet when the path passing through it appears to be slightly longer (by up to m hops) than the currently believed shortest path. This way, m introduces controllable redundancy into the scheme. To see how it helps the network cope with node dynamics (mobility, failures), consider the scenario shown in Figure 3. Packets traveling between U and V are forwarded within the clouded fragment of the network. Suppose that the arrows represent neighborhoods. In a steady state, the path A–B–C (of length 2) is the shortest route through the cloud. Let m be set to 1. This means that nodes E and F will also retransmit the packets because the route through them incurs a 1-hop increase over the best path. The worst thing that can happen is the disappearance of node B, which is a critical component of the current best path. Note, however, that this disappearance will not disrupt the traffic because the second best path through the cloud, i.e., A–E–F–C, is also being used. The net outcome of this disappearance will be that a would-be duplicate arriving at A or C (from E or F), will be now *bona fide* received and forwarded towards the destination. After a short while, as the destinations update their h_b values in response to the increased number of hops along the best path, the nodes within the cloud will learn that A–E–F–C is the best path at the moment.

To cope with more drastic changes in the network configuration, every once in a while, a routing node forwards the packet to be otherwise dropped (based on the inequality in Figure 2). This happens with frequency inversely proportional to the margin by which the inequality holds.

An important property of the rule-based routing system of TARP is that the way all rules are implemented makes them automatically scalable to the amount of RAM available at the node. This is because their behavior is driven by caches, and the lack of information in the cache forces the rule to decide that the packet *should not* be dropped on its account. Thus, a node with too little memory may make suboptimal decisions, i.e., forwarding more packets than it absolutely should, but it will never break the connectivity.

4 Praxis structure

The interface of a PicOS application (praxis) to the outside world is governed by a powerful module called VNETI (for Virtual NETwork Interface). VNETI offers to the praxis a simple collection of API, independent of the underlying implementation of networking. To avoid the protocol layering problems haunting small-footprint solutions, VNETI is completely layer-less and its semi-complete generic functionality is redefined by plug-ins. For example, TARP is just a plug-in to VNETI. The actual connection to the physical transceiver module is encapsulated and abstracted into a PHY—an easily exchangeable module with a minimalistic program interface. Multiple plug-ins and physical interfaces can coexist within the same system configuration.

The interaction of all system components is shown in Figure 4. VNETI provides built-in means for buffering packets, moving them among different queues, and locating the relevant components of their payloads. A plug-in may intercept

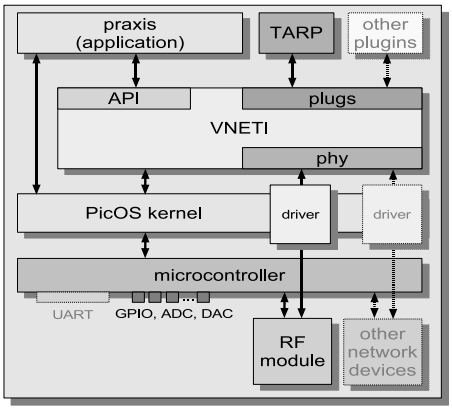


Fig. 4. System structure.

some or all packets arriving from the network or from the application, modify them, insert new packets into queues, and so on. All this happens in a way that preserves the built-in API and PHY interfaces, which are formally independent of the configuration of plug-ins. In contrast to the traditional layered approach, a VNETI plug-in acts in a "holistic" manner, which means that it can see and affect the *whole* of the packet's structure and fate. The key to power, convenience, simplicity, and safety (resilience to errors) of this approach lies in the careful design of the available tools.

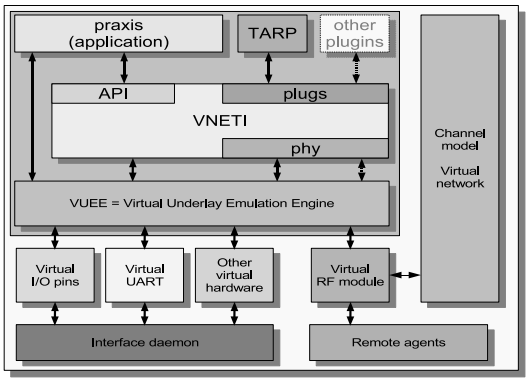


Fig. 5. The structure of a VUE^2 model.

An important stage of the application design process is its testing, which is greatly facilitated by VUE^2 (Virtual Underlay Emulation Engine) based on SMURPH [5]. VUE^2 makes it possible to execute the complete application in an artificial environment, where multiple nodes communicate over virtual wire-

less channels and, possibly, talk to real-life OSS agents. The latter means that the complete functionality of the target networked application can be verified virtually, including its collaboration with real external programs.

The structure of a VUE^2 model is depicted in Figure 5. Owing to the fact that PicOS's programming paradigm directly descends from SMURPH [2], it is reasonably easy for VUE^2 to understand the structure and behavior of a PicOS program and simulate its execution. In particular, exactly the same VNETI (the same source code) executes under PicOS and VUE^2. That source code, together with the praxis program, are mechanically transformed into a SMURPH program, then compiled and linked to the libraries providing the models of standard components, including wireless channels and peripheral devices, like UARTs, A-D converters, etc. If the network is not excessively large, the model is able to execute in real time, meaning that the observed time flow coincides with its flow in real life. A slow-motion factor can be applied to models of large networks.

5 Summary

Using the technology described in this paper, we have been able to build several practical ad-hoc networks, including serious industrial deployments. By a "practical network" we understand one that works and meets the expectations of its users.

Customary, we extend the notion of practicality onto technologies, e.g. we say that Ethernet technology is practical, even if some of its botchy specimens fail. Networks acquire practicality via technological progress and industrial acceptance, but this acquisition need not be universal. Ethernet or IP networks are indisputably practical, so are ATM and Bluetooth, even if their cases illustrate the fact that practicality not always follows common sense. On the other hand, IP extensions (meant to make it a "one for all" choice), should, after all these years, be denied practicality. So, we are afraid, must some wireless ad-hoc networking schemes, notably ZigBee®, despite powerful industrial sponsors behind them. In the latter case, most of the harm is inflicted by confusing a general scheme with a complete solution, of which the scheme is merely a (likely quite suboptimal) part.

We have presented here a technology that, in our opinion, makes ad-hoc networks practical, i.e. functional and deployable in a variety of industrial frameworks. While we do not claim that ours is the only possible approach to practical ad-hoc networking, we couldn't find a better one despite honest attempts. Our present library of application blueprints (working, demonstrable, open-ended data-exchange patterns) makes us confident that the combination of tools and methodologies comprising our platform is powerful enough to handle many practically interesting cases of distributed sensing, monitoring, industrial process control, and so on.

References

1. T-W. Chen and M. Gerla. Global state routing: a new routing scheme for ad-hoc wireless networks. In *Proceedings of ICC'98*, June 1998.
2. W. Dobosiewicz and P. Gburzynski. From simulation to execution: on a certain programming paradigm for reactive systems. In *Proceedings of the First International Multiconference on Computer Science and Information Technology (FIMCSIT'06)*, pages 561–568, Wisla, Poland, nov 2006.
3. D. Drusinsky, M. Shing, and K. Demir. Creation and validation of embedded assertions statecharts. In *Proceedings of 17th IEEE International Workshop Rapid Systems Prototyping*, pages 17–23. IEEE CS Press, 2006.
4. D. Dubhashi et al. Blue pleiades, a new solution for device discovery and scatternet formation in multi-hop Bluetooth networks. *Wireless Networks*, 13(1):107–125, 2007.
5. P. Gburzynski and I. Nikolaidis. Wireless network simulation extensions in SMURPH/SIDE. In *Proceedings of the 2006 Winter Simulation Conference (WSC'06)*, Monetery, California, dec 2006.
6. M. Gunes, U. Sorges, and I. Bouazizi. ARA—the ant-colony based routing algorithm for manets. In *Proceedings of International Workshop on Ad-hoc Networking (IWAHN)*, Vancouver, British Columbia, Canada, August 2002.
7. J. Hui. Tinyos network programming (version 1.0), 2004. TinyOS 1.1.8 Documentation.
8. D. B. Johnson and D. A. Maltz. Dynamic Source Routing in ad hoc wireless networks. In Imielinski and Korth, editors, *Mobile Computing*, volume 353. Kluwer Academic Publishers, 1996.
9. P. Levis et al. TinyOS: An operating system for sensor networks. In W. Weber, J.M. Rabaey, , and E. Aarts, editors, *Ambient Intelligence*, pages 115–148. Springer, 2005.
10. J. Li, J. Jannotti, D. De Couto, D. Karger, and R. Morris. A scalable location service for geographic ad hoc routing. In *Proceedings of the ACM/IEEE International Conference on Mobile Computing and Networking (MOBICOM' 00)*, pages 120–130, 2000.
11. X. Ling et al. Performance analysis of IEEE 802.11 DCF with heterogeneous traffic. In *Proceedings of Consumer Communications and Networking Conference*, pages 49–53, Las Vegas, NV, January 2007.
12. V.D. Park and M.S. Cors, on. A performance comparison of TORA and ideal link state routing. In *Proceedings of IEEE Symposium on Computers and Communications '98*, June 1998.
13. C. Perkins, E. Belding Royer, and S. Das. Ad-hoc On-demand Distance Vector Routing (AODV), February 2003. Internet Draft: draft-ietf-manet-aodv-13.txt.
14. C. E. Perkins and P. Bhagwat. Highly dynamic Destination-Sequenced Distance Vector routing (DSDV) for mobile computers. In *Proceedings of SIGCOMM'94*, pages 234–244, August 1993.
15. C. E. Perkins and E. M. Royer. Ad-hoc On-demand Distance Vector Routing (AODV). In *Proceedings of the IEEE workshop on Mobile Computing Systems and Applications (WMCSA)*, pages 90–100, 1999.
16. C-K Toh. A novel distributed routing protocol to support ad-hoc mobile computing. In *Proceedings of IEEE 15th Annual International Phoenix Conf. on Comp. and Comm.*, pages 480–486, March 1996.

Proceedings of the International Multiconference on Computer Science and Information Technology pp. 995–1004
ISSN 1896-7094
© 2007 PIPS

Problems When Realizing Ad Hoc Networks: How a Hierarchical Architecture Can Help

Stefan Bouckaert, Dries Naudts, Ingrid Moerman, and Piet Demeester

Ghent University – IBBT vzw – IMEC vzw
Department of Information Technology (INTEC) – IBCN
Gaston Crommenlaan 8/201 – B-9050 Gent—Belgium.
sbouckae@intec.ugent.be – WWW: http://www.ibcn.intec.ugent.be

Abstract. Despite the fact that many electronic devices are equipped with wireless interfaces and a lot of publications on wireless ad hoc and mesh networking exist, these networks are seldom used in our everyday life. A possible explanation is the fact that only few of the numerous theoretically promising proposals lead to practical solutions on real systems. Currently, wireless network design is mostly approached from a purely theoretical angle. In this paper, common theoretical assumptions are challenged and disproved, and key problems that are faced when putting theory to practice are determined by experiment. We show how these problems can be mitigated, and motivate why a heterogeneous hierarchical wireless mesh architecture can help in making wireless ad hoc networking a reality.

1 Introduction

Wireless ad hoc networks based on the IEEE802.11 standard have been a popular research topic for many years. Despite the fact that a rising number of electronic devices are being equipped with wireless interfaces as technology gets cheaper, ad hoc networks are rarely used in our everyday life.
How can this be explained, even though it is easy to think of scenarios that could benefit from ad hoc technology?

In [1], the authors answer this question by suggesting that most ad hoc network design focuses on military or specialized civilian applications, making the solutions impractical for everyday life. This is an important observation, however, this paper addresses another, perhaps more fundamental problem. We feel that a lot of research gets stuck in a crucial phase of development: while there are a massive number of initiatives to design wireless ad hoc solutions, few ideas are implemented on actual systems. Unfortunately, as promising as some ideas may be, they do not always lead to good or practical solutions. We have found that, using a purely theoretical top-down approach where implementation is the last step in designing wireless network protocols, architectural decisions are often made which, after months of research, turn out to be impossible to realize because of unforeseen implementation problems.

Wireless research focused on single-interface homogeneous ad-hoc networks for a long time. Recently, an evolution in wireless networking research is observed: more researchers start focusing on multiple interface nodes in mesh topologies. Additionally, several aspects of cross-layer research, such as power control, are gaining popularity. While there is a growing awareness within the research community that simulation of protocols might not be sufficient in order to declare wireless networking protocols working and stable [2, 3], a lot of assumptions are still made while studying old and new topics in wireless research. In this paper, we will verify the validity of common assumptions by experiments, and formulate lessons learned while evaluating several experimental set-ups and real life implementations over the past months.

2 Observations and Experiments

When collecting data from test set-ups using any wireless driver, one should always keep in mind that the driver could be causing errors at a node. We have tried, to the extent possible, to exclude driver-caused errors from the observations below. The observations follow from several real-life experiments and testbeds, using a broad range of hardware. A first testbed consists of 18 Linux nodes, equipped with D-Link AG520 wireless a/b/g PCI cards and external antennas. In addition, a lot of research on mesh networks at our labs is done using several modified 4G Meshcubes with up to 4 wireless mini-PCI a/b/g cards per node. Other tests are performed using Linksys WRT54GL wireless routers with modified firmware. All devices can be powered using batteries, allowing to test real mobility.

2.1 Single Frequency, Single Hop Networking

***Assumption**: In an isolated situation, an 802.11 wireless link between two nodes will be of better quality if transmission power is increased.*

***Observation**:*

Consider a test set-up where three identical network nodes, each equipped with a single wireless network interface are stacked on top of each other in a rack. The external antennas are positioned in a triangle, the antennas about 1.5 meters apart. Although at some times data can be sent at the theoretical speed limit, results are very unpredictable. Even if a link seems to be stable for a certain period of time, data rates can drop below one third of the stable rate, seemingly without a reason. In addition, links are not always symmetrical: changing the direction of the traffic flow can result in degraded throughput.

In this specific setup, one solution seems to solve most of these problems. *Reducing* the transmit power of the wireless cards results in highly increased stability. In this case, a transmit power of $10mW$ gives the best results.

***Experiment**:*

When putting single interface Meshcubes to test in an RF-shielded box [4], the same observation is made in a controlled environment: reducing power when sender and transmitter are at close distances, increases networking quality.

Placing the antennas relatively close to each other, as in this case, might seem artificial—there are however a lot of situations imaginable where two single interface nodes are placed at comparable distances: e.g. a user can carry a wireless-enabled PDA and a laptop, or, during meetings, there is a high laptop density.

2.2 Multiple Frequencies, Multihop Networking

Assumption*: when configuring the different wireless interfaces of a multi-interface integrated node to theoretically non-overlapping channels, these different links will not interfere. Capacity of a wireless network can be increased dramatically by adding multiple interfaces.*

Observation*:*

Reading through literature, lots of innovative solutions involving wireless networking can be found, and numerous protocols are designed to support systems with multiple network interfaces [5, 6]. Many of these solutions are based on the assumption that multiple theoretically non-interfering channels can be operated simultaneously at a node's different interfaces. This seems obvious when considering theory and simulations. Unfortunately, when these solutions are deployed in a testbed, it turns out that this assumption is not necessarily true.

When a two-hop path is created using a traditional single-interface approach, the wireless medium has to be shared between two links. The maximum reachable throughput of the total path thus roughly halves because of a single additional hop. Adding a second wireless interface to the middle node and choosing orthogonal frequencies for the first and second link solves this problem, in theory, at the cost of adding a single extra interface. As simple as this may be, different results can be observed: even when two interfaces are available at the middle node and a two-hop path is constructed using two theoretically non-overlapping frequencies, the throughput does not rise considerably.

Still, in theory, devices supporting the 802.11b and 802.11g (802.11a) standard should be able to use, depending on the region, three (twelve) fully non-overlapping frequencies. Recently, other researchers described the same effects and concluded that wireless nodes with multiple interfaces can suffer severely from self-generated interference between the different interfaces [7].

Experiment*:* Measurements done at our lab (cf. Fig. 1) show that these effects of self-generated interference can be severely reduced by limiting the transmit power used at the different interfaces and physically separating the antennas of a node. On the other hand, these results also reveal the sad truth that—at least when using off the shelf hardware—a single wireless interface using high transmit power can severely degrade the performance of the other interfaces and surrounding nodes, even when they are set to operate on non-overlapping channels.

When using integrated 802.11a devices such as the Meshcubes in a two-hop test, interference problems are still observed, even when lowering transmit power. The first and second link can be set up separately on orthogonal channels with

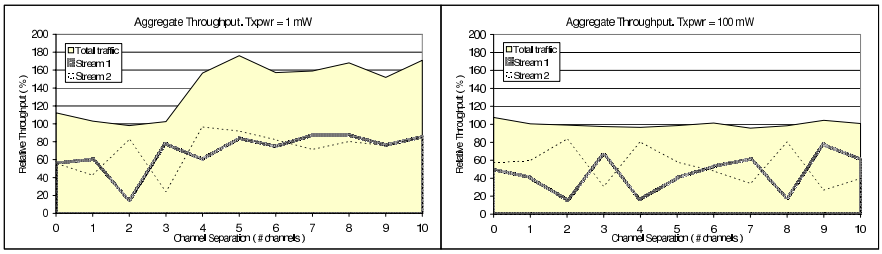


Fig. 1. Throughput measurement. Using two distant groups of two, 1.5 meter separated, ethernet-linked WRT54GL routers-representing two nodes with two interfaces each-, two connections are set up in parallel. The Y axis shows the throughput, relative to the maximum throughput of a single, non-interfered stream.

a goodput of over $30Mbps$. However, if both links are used at the same time in a multihop configuration, the goodput drops to about $16Mbps$ even though the processor of the middle node can handle the packet forwarding. We have found that in this case, the problem essentially follows from the fact that the Meshcubes are highly integrated: it's not only the fact that two mini-PCI cards are located right on top of each other that causes problems, but especially the fact that the distance between the antennas is too small. This is not surprising: the antennas in the integrated devices are located very close to each other. At a frequency of $5GHz$, the wavelength used for transmission is about $6cm$, thus in our integrated devices, up to 4 antennas are separated by less than a single wavelength, resulting in a very unpredictable system. Increasing the distance between two antennas to about $15cm$ alleviates this intra-node interference and goodput rises considerably. In [7], the authors conclude that these interference effects can be solved by providing an antenna separation of $35dB$. However, this separation is hard or even impossible to obtain in mobile devices with a small form factor.

Figure 2 shows additional measurements using WRT54GL routers at a transmission power of $1mW$, quantifying the effect of antenna distance on interference between two non-overlapping 802.11g channels: channel 1 and 11. In scenario A, two separate UDP streams (starting at 30Mbps, decreasing until packet loss is minimal) with a UDP packet size of 1470 bytes are set up, sequential at first, then simultaneous. The graphs with circles and squares show the according sum of the average throughput observed at the receiving interfaces. The individual throughput is not shown on the graph, as bandwidth is equally divided between the two flows. The figure shows that the sum of throughputs is reduced by 16.8% when the flows are set up simultaneously with a distance d of 1 meter between the antennas. At a distance of $5cm$, throughput is reduced by over 45% compared to non simultaneous transmission. The sum of throughputs reduces both in the sequential and parallel tests when the antenna distance is decreased.

In scenario B of Fig. 2, a single UDP stream with same characteristics is set up. The test packets are now transferred over wire between interface B and D, recreating a typical multihop situation where every receiving interface

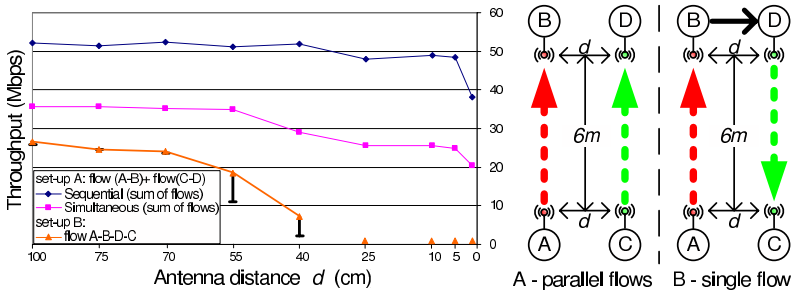


Fig. 2. Two scenarios quantifying the effect of the antenna distance on throughput. Link A-B operates on channel 1, link C-D on channel 11. Measurements at $d = 1cm, 5cm, 10cm + k \cdot 15cm, k = 0..6$. Txpower $= 1\ mW$.

has a sending interface nearby. Looking at the graph with the triangles, a first observation is that the end-to-end throughput is higher than half the aggregate throughput from scenario A, for antenna distances d larger than $55cm$: the single flow is transferred more efficiently than two traffic flows originating separately. However, performance drops faster with the decrease of antenna distance. It is also observed that unlike in the previous situation, throughput is very unstable when the antennas are moved more closely to each other: moving the antennas slightly can result in a serious performance drop or increase. For this reason, the graph with the triangles shows the maximum values that where obtained from a large amounts of tests. Minimum throughput results are shown with error bars. When the distance between antennas drops below about $40cm$, communication at reasonable throughputs is no longer possible. It is clear that the consequences of interference between neighboring interfaces with nearby antennas are worse in scenario B than in scenario A. The receiving interfaces are disturbed by a signal of the nearby transmitter, even though it is sending on a non-overlapping channel, rendering successful reception of the UDP test packets impossible. Note that in both situations, 802.11ACK messages are sent in the opposite direction of the UDP flows. However, these small packets suffer less from the interference of neighboring interfaces.

This observation raises questions about using multiple interfaces in integrated devices: even if perfect algorithms for using multiple interfaces on devices can be thought of and simulated, it is very likely that the result will not be as expected when deploying them in real integrated systems. We also believe that a testbed with 'full size' computers and wireless PCI cards with external antennas, can not fully represent an integrated end-user device with multiple interfaces.

2.3 Power Adaptation

From previous paragraphs, we have learned, that changing power levels can lead to a more reliable link, but increasing transmit power does not necessarily increase communication quality. Furthermore, it was shown that theoretically non-interfering channels will interfere when using off-the-shelf hardware at mid

to high transmit power, and that integrated systems with multiple interfaces will suffer from self-generated interference if there is no adequate antenna separation. Consequently, a single device transmitting at a relatively high output power may render all surrounding communication virtually impossible.

Lowering transmission power is often considered as a measure of freedom, in order to decrease interference and thus increase the number of possible simultaneous transmissions in a certain area, or to increase the lifetime of battery powered devices [8]. Our experience shows that when using today's 802.11 hardware as a base for multi-interface nodes, power adaptation is not a measure of freedom but rather a necessity in order to guarantee network operation. End-user hardware can and probably will improve in quality over time, however, we predict that it is very unlikely that in the near future palm-size systems will be able to fully take advantage of using multiple interfaces at relatively high output powers, if the interfaces are tuned to neighboring channels. As frequency regulations only allow a small part of the spectrum to be used for unlicensed civilian WLAN communication, it is not easy to provide the required channel separation. When using only two interfaces, putting one interface in the $2.4GHz$ range and the other one in the $5GHz$ might turn out effective with some types of hardware, but it is a mere palliative as this solution can currently not be scaled.

2.4 Hardware Issues

A recurring problem faced during tests with hardware from different vendors, is that changing an interface to some specific channels can result in a wireless link of bad quality. As communicating on those channels is possible using hardware from a different vendor, and bad channels stay bad when replacing hardware with an identical spare, the problem is most likely hardware related.

In general, there is a big difference in stability and performance between hardware from different vendors. In Fig. 3, the spectrum of two mini-pci cards from different vendors, operated at the same power level and frequency are shown. The figure clearly shows that the spectrum of the first card suffers a lot more from frequency leakage than the second card. Consequently, when building a testbed with hardware of the first type, test results will be much more pessimistic than test results with hardware from the second vendor, especially when operating at multiple theoretically non-interfering frequencies.

Although these problems can be solved by replacing faulty hardware with hardware from a different vendor, ad hoc networking protocols will only become successful if a large group of end users can use them instantly without problems, regardless of their choice of vendor. Unfortunately, in a traditional ad hoc network where nodes can join freely, it is wrong to assume that all nodes will react identically to a specific algorithm's action. If the quality of the hardware cannot be guaranteed, control loops should be provided within algorithms to verify whether a certain action has the desired effect. These effects are very hard to model in a simulator and can only be discovered by putting algorithms to test on real-life testbeds.

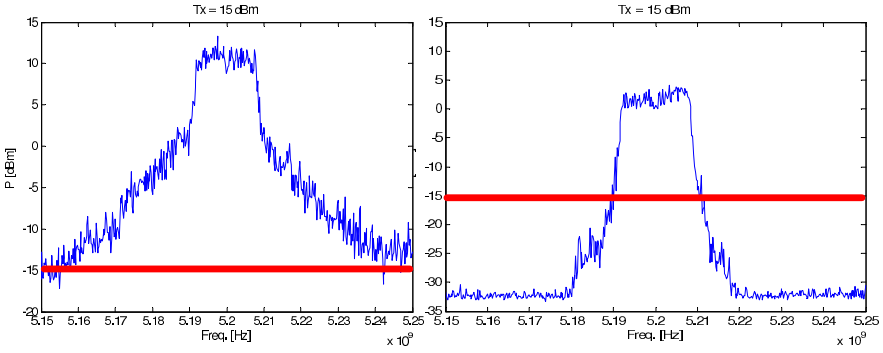


Fig. 3. Spectrum measurements of two different 802.11a mini PCI-cards operating at same Txpower (15dBm), both using channel 40.

2.5 A Lab Environment is not a Real Environment

Assumption*: (i) If it works in simulation, it will work on a testbed. (ii) If it works on a testbed, it will work in real life.*

From previous paragraphs, it is clear that designing algorithms and protocols for wireless systems should preferably not be done solely by considering theory or simulations. Creating a wireless test environment in a laboratory is not an easy task. Not only does it require a lot of -sometimes costly- hardware and space, it is also time consuming. On the other hand, creating these test environments and implementing developed algorithms on actual hardware forces the researcher to develop a system close to reality.

However, there is always a risk that the testing environment itself will lose its value as 'real test case', as over time wireless systems could be tuned—unintentionally—to work great in the testing environment only. This way, a solution that evaluates positive in a testing environment can at the same time be useless when deployed in a random situation. This is a frustrating experience that was witnessed before at our lab: a demonstrator which transmits video over a self-forming and self-recovering multihop mesh/relay network was developed. After the demonstrator had proved to be working perfectly in the lab environment—even when moving the battery fed relay stations through the building—it was taken to a large hangar. Surprisingly, even with relatively short distances between the relaying hardware and line of sight communication, link breaks occurred frequently and maximum throughput was low. In this case, the set-up probably suffered from the absence of the waveguide effect described in [9]: there are circumstances where a wireless signal does not degrade as fast when using devices indoor, compared to using them in an open space.

This example, amongst others, shows that a system should not be declared stable based on a single test environment, and certainly not based on simulations. We believe that wireless ad hoc networking protocols and systems will only be used in our everyday life if their use is not limited to a specific scenario or environment—however, today, a lot of algorithms are evaluated only in simulators using a very specific test scenario.

3 A Heterogeneous Hierarchical Architecture

As shown in previous paragraphs, a lot of problems were observed while bringing wireless ad hoc and mesh networking algorithms to real systems. Some of these problems are hardware related, and one might argue that it is not the task of networking algorithms to correct them. However, wireless systems will always be more unreliable than their wired counterparts, and therefore, algorithms must be able to detect anomalies and react appropriately.

First, transmission power should be chosen wisely: neither too low nor too high. While in a static set-up, transmission power can be set manually by trial and error, there is need for automatic tuning in dynamic environments.

Second, because of interference and hardware related issues, the channel choice has a great impact on the wireless link quality. Problems occur at various layers of the protocol stack when wireless links break due to changing channel conditions or failing hardware. Cross-layer interactions might be required to cope with these complexities.

Third, as small devices with multiple interfaces suffer from self-generated interference, we should focus our research on an architecture which takes this fact into consideration. An algorithm which presupposes perfect separation between multiple interfaces at end-user nodes will most likely never be able to achieve its claimed results when used in real systems.

In a heterogeneous architecture, devices have distinct capabilities and technologies. In a hierarchical architecture, different nodes can belong to different logical groups, for example, backbone nodes and clients. Heterogeneous hierarchical architectures (Fig. 4) have been described in the past, however, we believe that their true potential has not been discovered yet. In [10], the authors describe a (hybrid) wireless mesh architecture. In a wireless mesh network (WMN), two types of nodes are distinguished: mesh routers and mesh clients. Mesh routers hold superior properties concerning processing power, interfaces, available power and memory, enabling them to perform more complex functions. In addition, they have limited mobility compared to the clients, resulting in a wireless mesh backbone. Mesh routers can be added or removed at any time and act as a gateway to other networks such as the internet. In a *hybrid* WMN, mesh clients can connect to the backbone network either directly, or by using a multi hop path through other clients.

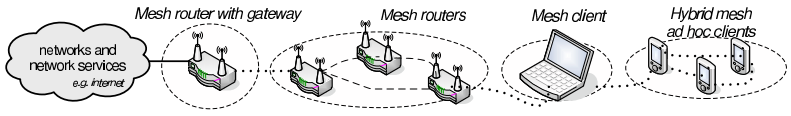


Fig. 4. Heterogeneous hierarchical mesh network, i.c. a Hybrid Wireless Mesh Ad Hoc network. Clients connect either directly or through another client to a mesh backbone.

Some benefits of heterogeneous hierarchical networks have been described in the past, such as an increase in coverage, or the (theoretical) ease of set-up.

However, we believe that there are more reasons why hierarchical heterogeneous architectures can help to realize robust wireless networks, and that a conscious choice of networking architecture can help to make certain assumptions that are invalid for homogeneous wireless networks become valid.

Most small and mobile end user devices such as PDAs or smartphones will probably only have a single (high speed) wireless network interface, using the unlicensed bands, enabled at a time, as adding interfaces is suboptimal due to the described interference problems and other limitations such as power and cost. On the other hand, the mesh routers in the backbone can and should have multiple interfaces: they can be bigger in size and antennas can adequately be separated, thereby reducing the interference problems.

Faulty hardware will always be used within a cooperative wireless network, resulting in decreased performance and satisfaction for the end-users. In a traditional ad hoc network, even if one user invests in high-quality hardware, he can still experience bad performance if the person he is connecting through uses faulty hardware. In a heterogeneous architecture, end users can e.g. connect to a mesh backbone which is constructed with hardware of better quality. The nodes which are higher in the hierarchy can be more expensive, as less nodes of higher hierarchy are needed.

In a hierarchical network architecture, operators can invest in high quality wireless backbone nodes. In addition to increasing the number of interfaces, more expensive network nodes could also use alternative technologies such as WiMAX. By using more interfaces or better technologies at higher hierarchical layers, wireless networks become more scalable.

Changing protocols or interface configuration on all end user devices, constructed by different vendors, is harder to achieve than making changes to a smaller group of devices at a higher hierarchical layer, all the more since it is more likely that wireless devices at a higher hierarchical layer are controlled by a single administrator. For example, multi-interface wireless 802.11a backbone routers with (proprietary) cross-layer optimizations can easily provide wireless coverage within a building, or at a fair or festival. By adding an extra wireless interface to every backbone router, configured as an 802.11g access point, end users are able to connect to this backbone with hardware from any vendor, without compromising the quality of the backbone network. If a user has the right hardware and chooses to function as a relaying node and extend the network, he can do this by voluntarily installing the required protocols.

4 Conclusion

In this paper, assumptions that are commonly made when researching wireless ad hoc networking protocols were challenged. It was shown that, whether installing a single interface or multiple wireless interfaces at a node, real-life performance is always worse than can be expected from theoretical models or simulations. We raised questions about the usefulness of embedding multiple interfaces of the same type in a palm-size device, and argued that, in contrast to what is

believed in many research papers, adjusting transmission power is not a measure of freedom but a necessity. We described how a choice of hardware affects the efficiency of algorithms, and how this influences the stability of wireless networks.

In order to test the robustness of algorithms, testing on one or multiple testbeds is a necessity. However, one must keep in mind that positive testbed results do not always imply a stable system under all circumstances. Finally, we argued that a heterogeneous hierarchical wireless mesh network architecture can help solving the observed problems, by reducing the need for miniaturization and providing incentives for network operators and businesses. We believe that, if protocols are developed closer to reality, and more realistic architectural choices are made at the start of a design process, the usability of ad hoc and mesh networks can drastically be improved.

Acknowledgment

S. Bouckaert thanks the Institute for the Promotion of Innovation through Science and Technology in Flanders (IWT-Vlaanderen). This research is also partly funded through the IBBT-Cisco Eye-Sense project.

References

1. Bruno R., Conti M., Gregori E.: *Mesh networks: Commodity multihop ad hoc networks,* IEEE Communications Magazine (2005) 123–131.
2. Tschudin, C. F. Gunningberg P., Lundgren H., Nordström E.: *Lessons from experimental manet research,* Ad Hoc Networks **3** (2005) 221–233.
3. De P., Raniwala A., Sharma S., Chiueh T.: *MiNT: a miniaturized network testbed for mobile wireless research,* INFOCOM 2005. 24th Annual Joint Conference of the IEEE ComSoc. Proceedings IEEE **4** (2005) 2731–2742.
4. Qosmotec: Air Interface Simulator. `http://www.qosmotec.com/`
5. Tang J., Xue G., Zhang W.: *Interference-aware topology control and qos routing in multi-channel wireless mesh networks,* In: Proceedings of the 6th ACM international symposium on Mobile ad hoc networking and computing, Urbana-Champaign, IL, USA (2005) 68–77.
6. Kyasanur P., Vaidya N. H.: *Routing and interface assignment in multi-channel multi-interface wireless networks,* Wireless Communications and Networking Conference, 2005 IEEE **4** (2005) 2051–2056 Vol. 4.
7. Robinson J., Papagiannaki K., Diot C., Guo X., Krishnamurthy L.: *Experimenting with a multi-radio mesh networking testbed,* In: WiNMee—1st workshop on Wireless Network Measurements, Italy (2005).
8. Jung E. S., Vaidya N. H.: *Power aware routing using power control in ad hoc networks,* Technical report, CSL, University of Illinois, Urbana (2005).
9. Lu D., Rutledge D.: *Investigation of indoor radio channels from 2.4 ghz to 24 ghz,* In: IEEE Antennas and Propagation Society International Symposium. (2003) 134–137.
10. Akyildiz I. F., Wang X., Wang W.: *Wireless mesh networks: a survey,* Computer Networks **47** (2005) 445–487.

1st Workshop on Advances in Programming Languages (WAPL'07)

Programming languages are programmers' most basic tools. With appropriate proramming languages one can drastically reduce the cost of building new applications as well as maintaining existing ones. In the last decades there have been many advances in programming languages technology in traditional programming paradigms such as functional, logic, and object-oriented programming, as well as the development of new paradigms such as aspect-oriented programming. The main driving force was and will be to better express programmers' ideas. Therefore, research in programming languages is an endless activity and the core of computer science. New language features, new programming paradigms, and better compile-time and run-time mechanisms can be foreseen in the future.

The aims of this event is to provide a forum for exchange of ideas and experience in topics concerned with programming languages and systems. Original papers and implementation reports are invited in all areas of programming languages. Major topics of interest include but are not limited to the following:

- Automata theory and applications
- Compiling techniques
- Domain-specific languages
- Formal semantics and syntax
- Generative and generic programming
- Grammarware and grammar based systems
- Knowledge engineering languages, integration of knowledge engineering and software engineering
- Languages and tools for trustworthy computing
- Language theory and applications
- Language concepts, design and implementation
- Markup languages (XML)
- Metamodeling and modeling languages
- Model-driven engineering languages and systems
- Practical experiences with programming languages
- Program analysis, optimization and verification
- Program generation and transformation
- Programming paradigms (aspect-oriented, functional, logic, object-oriented, etc.)
- Programming tools and environments
- Proof theory for programs
- Specification languages
- Type systems
- Virtual machines and just-in-time compilation
- Visual programming languages

As a concept, wireless ad-hoc networking has been around for at least 20 years, but, despite the extensive coverage in academic literature, its practical success so far has been less than spectacular. Most contemporary attempts at exploiting the ad-hoc

paradigm in real-life applications fall into the realm of sensor networks, where it promises low-cost, ease of deployment, flexibility, reliability, resilience and ubiquity. However, few practical deployments of sensor networks are actually ad-hoc based: there seems to be a considerable reluctance in the industrial world to embrace "mesh networking," and most actual solutions are based on asymmetric schemes involving various flavors of base stations. The purpose of this workshop is to get to the roots of that reluctance. We solicit papers addressing the practical aspects of ad-hoc wireless networking, with stress on case studies and empirical verification of ideas. We shall not shy away from negative conclusions, as long as they are well supported.

The list of topics includes (but is not limited to):

- Routing protocols for ad-hoc wireless networks
- Medium access control schemes
- Modeling methodologies for wireless channels
- Simulation systems and techniques
- Performance studies and comparisons
- Case studies (ad-hoc networks at work)
- Security issues in ad-hoc networking
- Asymmetric wireless networks with ad-hoc components
- Cooperation issues at the MAC and network layers
- Pricing and incentive arbitration in ad hoc mesh topologies

International Programme Committee

- Zoran Budimac, University of Novi Sad, Serbia
- Fei Cao, Microsoft, USA
- Thomas Cleenewerck, Vrije Universiteit Brussel, Belgium
- Johan Fabry, INRIA, France
- Pedro Henriques, University of Minho, Portugal
- Jan Kollar, Technical University Kosice, Slovakia
- Tomaz Kosar, University of Maribor, Slovenia
- Alexander Legalov, Siberian Federal University, Russia
- Ivan Lukovic, University of Novi Sad, Serbia
- Pablo Martinez Lopez, National University of La Plata, Argentina
- Prabhat K. Mahanti , University of New Brunswick, Canada
- Jan Madey, Warsaw University, Poland
- Mykola Nikitchenko, National Taras Shevchenko University of Kyiv, Ukraine
- Terence Parr, University of San Francisco, USA
- Samir Ribic, University of Sarajevo, Bosnia
- Guenter Riedewald, University of Rostock, Germany
- Gilles Roussel, Université de Marne-la-Vallée, France
- Vladimir Safonov, St.Petersburg University, Russia
- Adnan Salihbegovic, University of Sarajevo, Bosnia
- Jose Luis Sierra, Universidad Complutense de Madrid, Spain
- Boatjan Slivnik, University of Ljubljana, Slovenia
- Vladimir Sulov, University of Economics Varna, Bulgaria

- Maria João Varanda Pereira, Instituto Politécnico de Bragança, Portugal
- Dániel Varró, Budapest University of Technology and Economics, Hungary
- Jurgen Vinju, Centrum voor Wiskunde en Informatica, The Netherlands
- Xiaoqing Wu, University of Alabama at Birmingham, USA

Organizing Committee

Marjan Mernik, University of Maribor, Slovenia

Proceedings of the International Multiconference on
Computer Science and Information Technology, pp. 1009 – 1020
ISSN 1896-7094
© 2007 PIPS

Post object-oriented paradigms in software development: a comparative analysis

Adam Przybyłek

Department of Business Informatics, University of Gdańsk,
Piaskowa 9, 81-824 Sopot, Poland
adam.przybylek@univ.gda.pl

Abstract. The object-oriented (OO) paradigm has been popularised as a solution to the problems encountered with the structured paradigm. It facilitates the understandability, extensibility, reusability and maintainability of systems. However, years of experience and analytical studies have shown that this is only partially true, and that there are still issues which have never been successfully resolved in the OO paradigm. These issues arise whenever programmers need to deal with peripheral requirements which spread over a system. A simple modification of these requirements typically involves intensive changes in code. Over the last decade interesting and worthwhile work has been done on the subject of implementing peripheral requirements. Perhaps the most successful outcomes have been obtained by aspect-oriented programming and composition filters. The main goals of the paper are: (1) to present open problems for the OO paradigm; (2) to analyse two post-OO paradigms involved in confronting these problems; (3) to indicate a possible application of these paradigms.

1 Introduction

For many years now theorists have agreed that the best way to manage the complexity of the system is to separate the system concerns [15, 19, 23, 24]. A concern is a specific requirement or matter for consideration that must be addressed in order to satisfy the overall system goal [2, 7, 10, 12, 13, 16, 20, 23]. In this paper concerns are categori sed according to whether they (1) capture the main business requirements or (2) capture peripheral requirements or technical-level issues that affect the system as a whole. The first category of concern is relevant to the application domain and is termed a “core concern”. The important property of core concerns is their ability to be separated at implementation level from other concerns of the same kind. On the other hand, “crosscutting concerns” play a supporting role and are required by some other concerns. Although they can be identified as distinct concerns, they are usually spread over the system and cannot be modularised into classes. Typical examples of crosscutting concerns are authentication, logging, synchronisation, transaction processing, failure handling, reliability, memory management, optimisation, performance tracking, real time constraints, storage management, data persistence, resource pooling and remoting. Figure 1 gives a visual depiction of concerns in a hypothetical application.

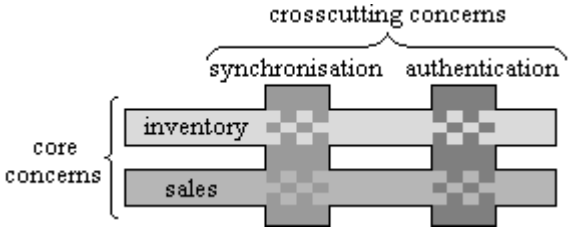


Fig. 1 . Intertwined concerns

"Separation of concerns" (SoC) refers to the ability to identify, encapsulate and manipulate parts of the software in order to deal with one important issue at a time [2, 9, 15]. It reduces software complexity and enhances understandability, adaptability, reuse and traceability throughout the development process [9, 19, 24]. Dijkstra [15] and Parnas [25] have suggested, that the best way to achieve SoC is through modularisation, the process of breaking the system into loosely-coupled modules that hide their implementation from each other. Individual modules can be relatively easily composed so as to meet the system requirements [10, 20, 23, 24]. This process, known as composition, is performed according to explicit statements about how the concerns should be put together.

The evolution of a software development paradigm is driven by the need to achieve a better SoC [15]. In software engineering the term "paradigm" is widely used to refer to the essence of certain software development processes. In spite of the fact that a paradigm refers to all the phases of system development, initial research usually originates in programming languages. New paradigms, therefore, occur in terms of programming, as in aspect-oriented programming (AOP).

Various paradigms have been studied that deal with system complexity and divide the systems into modules. Currently the most popular is the object-oriented (OO) paradigm. This provides techniques for reducing software complexity by introducing concepts such as abstraction, encapsulation, aggregation, inheritance and polymorphism. However, modularisation of each concern in an extensible manner cannot be achieved using the OO paradigm [10, 12, 20, 23]. Attempts to provide a solution to this problem have included AOP and composition filters (CFs). Each of these paradigms builds on all the advantages of the OO paradigm and overcomes some OO weaknesses. The term "post-OO paradigm" has therefore been introduced to refer to them.

The remainder of the article is structured as follows. Section 2 continues analysis of the motivations for post-OO paradigms. Section 3 and 4 introduce the principles of AOP and CFs respectively, both paradigms being illustrated with examples which focus on software reuse. Section 5 compares the paradigms under consideration and presents similarities and differences between them. Section 6 outlines the future work. Section 7 concludes the paper.

2 Weaknesses of the object-oriented paradigm

The OO paradigm was created from a desire to have constructs to represent real world objects. It has been claimed that: (1) OO systems support software evolution,

extension, and modification; (2) OO systems are reusable and can be easily constructed from existing components; (3) OO systems are understandable to domain experts and developers [13, 17, 18, 26]. However, years of experience have revealed that the OO paradigm fails to deliver all its claimed benefits [21].

Classes are a powerful way of encapsulating and localising changes to the core concerns, but they fall short in addressing the crosscutting concerns which affect multiple places in the system. Symptoms of implementing crosscutting concerns in an OO environment are "code scattering" and "code tangling" [10, 12, 20, 23]. Scattering occurs when multiple fragments of code that all do the same thing (or that do closely related things) are found in more then one class. The term "code tangling" is often used to describe the situation where a class contains logic pertaining to more than one concern. Code tangling and code scattering together negatively affect software design and development in many ways: poor traceability, low productivity, low code reuse, poor quality and difficulty in evolution [23].

3 Aspect-oriented programming

In the OO paradigm everything is considered as an object, but not everything is an object either in the real world or in system development. For example, synchronisation, transaction processing and reliability are not objects [21]. They are crosscutting concerns that cannot be effectively modularised in classes. AOP was introduced to address these shortcomings. It does not replace the OO but instead complements and builds on what the OO already offers. The AOP allows crosscutting concerns to be implemented separately from core concerns and then merges both into an executable code [10, 11, 12, 20, 22, 23].

Crosscutting concerns, together with weaving rules, are implemented in a new unit of modularity called an "aspect". Aspects centralise functionality, which is distributed over the whole system in the OO model [20]. Weaving rules specify how to integrate core and crosscutting concerns in terms of pointcuts and joinpoints. A pointcut is a language construct used to define a set of joinpoints. Joinpoints are well-defined locations in the program flow at which a crosscutting concern needs to be merged with core concerns. Typical examples of pointcuts could be a method being called, an exception being thrown, or an object being created.

The code that is inserted upon reaching a particular joinpoint is called an advice. An advice represents the body of a crosscutting concern. There are three types of advice: "before", "after", and "around". A before advice executes a designated code just before matched joinpoints. An after advice runs after the execution of matched joinpoints. An around advice executes a defined code instead of the code in the joinpoints [11, 20]. Moreover, an advice is able to access parameters of any method in any class. The actual process of concern merging, also known as "weaving", uses aspects and classes to compose the final system. Core concerns no longer embed the logic of crosscutting concerns, which are isolated into aspects [23].

The AOP has moved into the forefront through the work [22] of Gregor Kiczales and his team at Xerox's PARC. The most mature aspect-oriented implementation available today is AspectJ, which is an extension to Java.

For the purpose of illustration, an implementation of a tracing concern is pr esented. A learning management system (LMS) has been deployed at a college. After

some time, the suspicion is formed that unauthorised persons know lecturers' passwords and are modifying data in the system. As a result, it is decided that all data-changing SQL statements have to be watched. In order to do this, it is proposed that the existing application be extended by logging every DML statement that modifies the records, together with the date of the incident and the user name.

The original version of the application uses JDBC to access the database (Fig. 2). The key elements of JDBC API (in terms of the example presented) are the Connection interface, the Statement interface and the DriverManager class. DriverManager manages all the details involved in establishing the connection to a specified database. The established connection is returned by DriverManager::getConnection(..), which is a static method. In a typical application scenario the next step is creation of the Statement object. The Connection::createStatement() method is called upon to do this. The Statement object is associated with an open connection and used to send SQL statements to the DBMS. DML statements such as INSERT, UPDATE, DELETE are usually executed using the method Statement::executeUpdate(..).

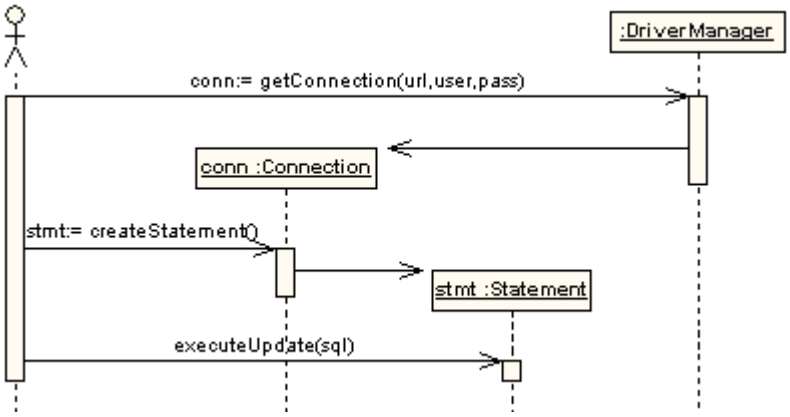


Fig. 2 . A typical usage scenario for accessing a database

In the conventional OO model, the best solution to the problem presented is based on a delegation-based reuse. A LogStatement class has been created to trace queries. This class implements the Statement interface and aggregates the statement object. All messages specified by the Statement interface are delegated to the statement object. Those which are able to modify the records (i.e. execute, executeUpdate) are redefined in order to implement an additional concern (tracing). The definition of the LogStatement class and an example of its application is presented on Listing 1.

```
public class LogStatement implements Statement {
  private Statement delegate;
  publ ic LogStatement(Statement st) {
    delegate = st;
  }
  private boolean isDML(String sql) {
    String tmp = sql.toUpperCase();
    return tmp.indexOf("UPDATE")>=0 ||
     tmp.indexOf("INSERT")>=0 || tmp.indexOf("DELETE")>=0;
  }
  private void log(String sql) throws SQLException {
    String user =
     delegate.getConnection().getMetaData().getUserName();
    System.out.println("["+new Date()+"]; "+user+"; "+sql);
  }
  public void cancel() throws SQLException {
    delegate.cancel();
  }
  public void close() throws SQLException {
    delegate.close();
  }
  public Connection getConnection() throws SQLException {
    return delegate.getConnection();
  }
  public ResultSet executeQuery(String sql) throws SQLException
{
    return delegate.executeQuery(sql);
  }
  public boolean execute(String sql) throws SQLException {
    if (isDML(sql)) log(sql);
    return delegate.execute(sql);
  }
  public boolean execute(String sql, int autoGeneratedKeys)
   throws SQLException {
    if (isDML(sql)) log(sql);
    return delegate.execute(sql, autoGeneratedKeys);
  }
  public boolean execute(String sql, int[] columnIndexes)
   throws SQLException {
    if (isDML(sql)) log(sql);
    return delegate.execute(sql, columnIndexes);
  }
  public boolean execute(String sql, String[] columnNames)
   throws SQLException {
    if (isDML(sql)) log(sql);
    return delegate.execute(sql, columnNames);
  }
  public int executeUpdate(String sql) throws SQLException {
    if (isDML(sql)) log(sql);
    return delegate.executeUpdate(sql);
  }
  //other methods specified by the Statement interface
}
```

Listing 1. LogStatement class definition

The suggested implementation limits the number of changes in the existing source code. However, in order to fulfil the new requirement, every statement object returned by Connection::createStatement() has to be wrapped with the LogStatement. The LogStatement implements 40 methods. Of these 1 is a constructor, 8 methods perform tracing and delegating, 29 methods are used for delegating only and 2 methods implement tracing only. It is apparent that the tracing concern (marked code) is scattered through and tangled with the business logic. A better solution can be obtained by using AOP and implementing a new functionality as an aspect (Listing 2).

```
public aspect DMLmonitoring {//1
  pointcut DMLexecute(String sql, Statement st)://2
   call( * Statement.execute*(..) )
   && args(sql,..) && target(st);
  after(String sql, Statement st) returning (Object o):
   DMLexecute(sql,st) {//3
    String user = "";
    try {
      user=st.getConnection().getMetaData().getUserName();
    } catch (SQLException e) {};
    String tmp = sql.toUpperCase();
    if( tmp.indexOf("UPDATE")>=0 || tmp.indexOf("INSERT")>=0
     || tmp.indexOf("DELETE")>=0 ) {
      if (o instanceof Integer) {
        int i = ((Integer)o).intValue();
        sql += "; ("+i+")";
      }
      tmp = user + "; " + sql;
      System.out.println("[" + new Date() + "]; " + tmp);
    }
  }
}
```

Listing 2. DMLmonitoring aspect definition

The fundamental difference between a conventional and an aspect-based solution is modularisation of the tracing concern. Instead of mixing concerns in a single class, the aspect is defined (1). The joinpoints at which weaving should be made are identified as the places where the update methods are called (2). An after-returning advice (3) runs after a successful return from the execution of a matched joinpoint. The statements inside this advice implement the tracing concern and are applied whenever the records are updated, deleted or inserted.

4 Composition Filters Object Model

The Composition Filters Object Model (CFOM) was defined in the late 1980s by Aksit and Tripathi [1, 3] as a modular extension to the OO model. It offers composable techniques for specifying the behaviour of the class [4]. The motivation for the CFOM was difficulty in expressing any kind of message co-ordination in the conventional OO model [14].

The CFOM can be presented as two partitions: an interface layer and an impleme ntation layer [4]. The implementation layer can be thought of as a conventional OO model. It can be extended by the interface layer. To achieve this, the interface layer contains an arbitrary number of input and output filters. Incoming messages pass through the input filters and the outgoing through the output filters [5, 8, 14]. Each message has to pass through all the filters defined until it is dispatched. The filters are able to inspect and manipulate messages; messages may, for example, be discarded or redirected to other objects. A filter definition specifies a filter type and filter guards. A filter accepts a message if any of the filter guards matches the message. If not, the message is said to be rejected. Each guard has the form:

condition => selector_expression

A guard is matched if (1) the condition evaluates to true, and (2) the selector ex pression matches the message. The filter type determines the semantics associated with acceptance and rejection of messages [4], [5].

This paper presents the CFOM on a classical producer-consumer problem. In a producer-consumer dilemma two processes (or threads), one known as the “producer” and the other called the “consumer”, run concurrently and share a common, fixed-size buffer. The producer generates items and places them in the buffer. The consumer removes items from the buffer and consumes them. However, the producer must not place an item into the buffer if the buffer is full, and the consumer cannot retrieve an item from the buffer if the buffer is empty. Nor may the two processes access the buffer at the same time to avoid race conditions. The producer-consumer problem can be generalised to have multiple producers and consumers. The UML's sequence diagram in Fig. 3 presents a typical interaction between producer, consumer and buffer.

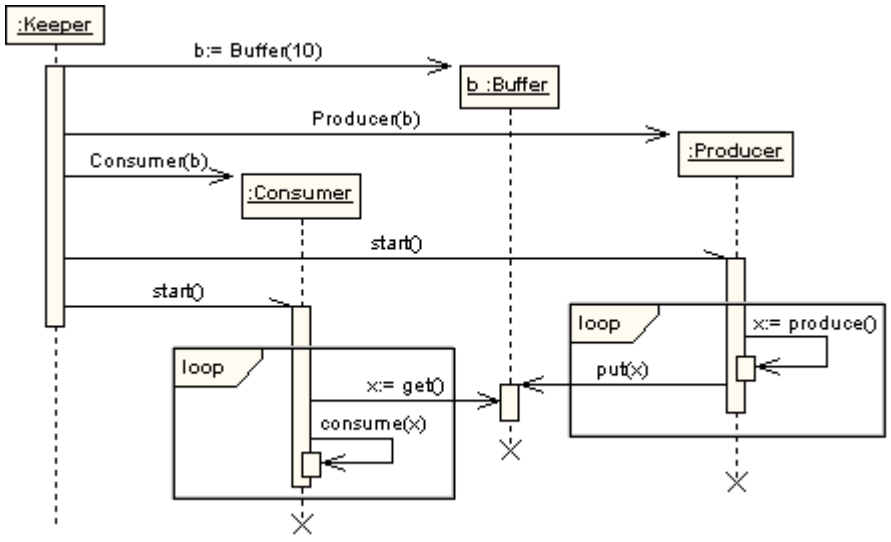


Fig. 3 . The producer-consumer interaction

Listing 3 gives a definition of a buffer class in Java. The OO solution tangles business logic together with the synchronisation concern. Moreover, the synchronisation code

is scattered through the methods responsible for accessing the buffer. In the result the put(..) and get() methods contain fragments of code for cooperation synchronisation.

```
public class Buffer {
  private int[] buf;
  private int counter;
  public Buffer(int capacity) {
    buf = new int[capacity];
  }
  public synchronised void put(int x) {
    while (counter == buf.length) try {
      wait();
    } catch (InterruptedException e) {
      System.err.println( e.getMessage() );
    }
    buf[counter++] = x;
    notifyAll();
  }
  public synchronised int get() {
    while (counter == 0) try {
      wait();
    } catch (InterruptedException e) {
      System.err.println( e.getMessage() );
    }
    int tmp = buf[--counter];
    notifyAll();
    return tmp;
  }
}
```

Listing 3. Buffer definition in the OO model

The alternative implementation is shown on Listing 4. The buffer is defined in pseudo-Java code using a simplified version of CFs.

```
public class Buffer {
  private int[] buf;
  private int counter;
  public Buffer(int capacity) {
    buf = new int[capacity];
  }
  public void put(int x) {
    buf[counter++] = x;
  }
  public int get() {
    int tmp = buf[--counter];
    return tmp;
  }
  filter Wait {
    (counter==buf.length) => put(int),
    (counter==0) => get()
  };
  filter Dispatch { true => this.* };
}
```

Listing 4. Buffer definition in the CFOM

A message arriving at a Wait filter is evaluated according to the filter specification. The first guard matches the put(int) message if the buffer is full (i.e. the current number of elements in the buffer is equal to the capacity of the buffer). The second guard matches the get() message if the buffer is empty. When the message is rejected, it proceeds immediately to a Dispatch filter. Otherwise the message is put in a queue, and delayed until the correspondent condition is satisfied. Re-evaluation of a Wait filter occurs at least at every change of the object state. In order to provide mutual exclusion, the pre-processor puts a synchronised block around every method for which a Wait filter is specified. However, while a message is blocked, the other messages can be processed.

The last filter in a filter set is always a Dispatch filter [4, 5]. When a Dispatch filter rejects a message, an exception is raised. In case of acceptance, the message is dispatched to the object that corresponds to the target of the first matching guard [4]. The pseudovariable "this" refers to the implementation layer. The wildcard "*" matches all messages which are in the signature of the target [5]. The definition "true => this.*" declares that all public methods in the implementation layer are unconditionally allowed to execute.

The next example illustrates that, CFs are also more effective than the OO paradigm, in adapting software to changes in requirements. The class definition in Listing 5 uses CFs to provide the tracing concern described in Section 3. LogStatementCF implements 11 methods. Of these, 1 is a constructor, 8 methods perform tracing and delegating, and 2 methods implement tracing only. No methods are used for delegating only, because delegation is achieved by the Dispatch filter. However, it should be noticed that in this example CFs implementation mixes the concerns.

```
public class LogStatementCF implements Statement {
  private Statement delegate;
  public LogStatementCF(Statement st) { delegate = st; }
  private boolean isDML(String sql) {
    String tmp = sql.toUpperCase();
    return tmp.indexOf("UPDATE")>=0 ||
     tmp.indexOf("INSERT")>=0 || tmp.indexOf("DELETE")>=0;
  }
  private void log(String sql) throws SQLException {
    String user =
     delegate.getConnection().getMetaData().getUserName();
    System.out.println("["+new Date()+"]; "+user+"; "+sql);
  }
  public boolean execute(String sql) throws SQLException {
    if (isDML(sql)) log(sql);
    return delegate.execute(sql);
  }
  //other execute(..) methods
  public int executeUpdate(String sql) throws SQLException {
    if (isDML(sql)) log(sql);
    return delegate.executeUpdate(sql);
  }
  //other executeUpdate(..) methods
  filter Dispatch { true => this.*, true => delegate.* };
}
```

Listing 5. LogStatementCF definition in the CFOM

5 Comparative analysis of AOP and CFs

For the purpose of this paper an evaluation study of two post-OO paradigms was performed. The examined paradigms share the same goal, which is to modularise crosscutting concerns for better reusability and easier system maintenance. In order to achieve this goal AOP introduces new units of modularity, whereas CFs enrich the class module. The guidelines for determining which paradigm is more effective for which issue are outlined in Table 1.

Table 1 . AOP vs. CFs.

paradigm / **issue**	**AOP**	**CFs**
implementing core concerns		√ √
implementing crosscutting concerns when they: a) cut across a single core concern b) cut across multiple core concerns	√ √ √ √	√ √
adjusting a system to new requirements	√ √	√
prototyping	√ √ √	√
tool availability	√ √	

Although AOP is more powerful th an CFs, it is slightly controversial. It breaks encapsulation and modifies flow control, making the source code hard to understand. AOP should therefore be used only for maintaining the system, rather than being the choice of convenience for developing the initial version of the system. On the other hand, CFs extend the OO paradigm in a natural way and so can be used like the classical OO mechanism. CFs also improve delegation-based reuse and allow the developer to avoid composition anomalies. In contrast to CFs, AOP is a completely non-invasive technique, allowing developers to introduce the new functionality without making any change to the core concerns, and is therefore an ideal prototyping tool because new concerns are easily added or removed as new requirements are explored. AOP is also appropriate for implementing development concerns such as tracing, profiling and testing. Development concerns are of interest only to the development team and have to be removed from a system prior to its release into the preproduction environment. Moreover, AOP is invaluable for implementing those crosscutting concerns which cut across multiple core concerns. On the other hand, crosscutting concerns that cut across a single core concern can be implemented by using AOP as well as by CFs. CFs are, however, still a theoretical concept unsupported by mainstream programming languages.

6 Future work

CFs potentially have the ability to work well together with the OO paradigm. However, the benefits of CFs are still in the realm of speculation, and more research is required to evaluate the practical usability of this approach. In particular the current OO languages should be enhanced to support CFs.

Table 1 shows that the presented post-OO paradigms are complementary in some areas. Combining the ideas behind AOP and CFs might be a valuable contribution to both paradigms. Therefore it is worth considering the development of tools that support the integration of these paradigms. Moreover, the way in which the post-OO paradigms support modularising crosscutting concerns at analysis and design level is an area that has still not been thoroughly explored.

7 Summary

The essential problem with the OO paradigm is the lack of proper mechanisms to separate the implementation of crosscutting concerns from the implementation of core concerns. Although it is possible to implement the crosscutting concerns in the conventional OO fashion, the presented examples show that such implementation produces classes with low cohesion and high coupling. Poor software quality causes that slight changes in requirements result in invasive modification of the source code.

The l imitations of the OO paradigm can be overcome by AOP and CFs, which, as outlined, indicate the direction in which software development should evolve. Although both post-OO paradigms form suitable candidates for implementing crosscutting concerns, one of them may be more suitable for some issues than the other. Moreover, these paradigms complement each other so that their respective weaknesses can be addressed, while maintaining their desirable properties. AOP modularises crosscutting concerns into separate aspects, making complete SoC possible. However, it violates encapsulation and should be used only as a last resort where other mechanisms introduce code tangling or scattering. On the other hand, the CFOM is consistent with the OO model and can improve its quality, making CFs appropriate for developing the initial version of the system. CFs are also useful in reusing modules without unnecessary redefinitions.

The main contribution of this paper lies in the guidelines for determining which paradigm is appropriate for which software development issue. Moreover, the author has presented the problems he encountered in OO system development and, in accordance with good practice, has explained how to solve these problems with the post-OO paradigms.

References

1. Aksit M.: *On the Design of an Object-Oriented Language Sina,* Dept. of Computer Science, University of Twente (1989).
2. Aksit M.: Composition and Separation of Concerns in the Object-Oriented Model. ACM Computing Surveys, vol. 28(4), ACM Press, New York (1996).

3. Aksit M., Tripathi A.: *Data Abstraction Mechanisms in Sina,* ACM Sigplan Notices, vol. 23(11). ACM Press, New York (1988) 267–275.
4. Bergmans L.: *Composing Concurrent Objects – Applying Composition Filters for the Development and Reuse of Concurrent Object-Oriented Programs,* Ph.D. thesis, University of Twente (1994).
5. Bergmans L., Aksit M.: *Composing crosscutting concerns using composition filters.* Commun. ACM, vol. 44(10). ACM Press, New York (2001) 51–57.
6. Bergmans L., Aksit M.: *Analyzing Multi-Dimensional Programming in AOP and Composition Filters,* In Proceedings of the OOPSLA'99 workshop on Multi-Dimensional Separation of Concerns (1999).
7. Bergmans L., Aksit M.: *Composing Crosscutting Concerns Using Composition Filters,* Communications of the ACM, vol. 44(10). ACM Press, New York (2001) 51–57.
8. Bergmans L., Aksit M., Tekinerdogan B.: *Mapping Aspects to Components,* University of Twente (1999).
9. Brito I., Moreira A.: *Integrating the NFR framework in a RE model,* In Proceedings of the 3rd Workshop on Early Aspects, 3rd International Conference on Aspect-Oriented Software Development, Lancaster (2004).
10. Clarke S., Baniassad E.: *Aspect-Oriented Analysis and Design: The Theme Approach,* Addison Wesley, Boston (2005).
11. Colyer A., Clement A.: *Aspect-oriented programming with AspectJ,* IBM Systems Journal, vol. 44(2). IBM Corp., Riverton (2005) 301–308.
12. Colyer A., Clement A., Harley G., Webster M.: *Eclipse AspectJ: Aspect-Oriented Programming with AspectJ and the Eclipse AspectJ Development Tools,* Addison Wesley, Boston (2004).
13. Constantinides C., Skotiniotis T.: *Reasoning about a classification of crosscutting concerns in object-oriented systems,* Second Workshop on Aspect-Oriented Software Development of the German Information Society. Bonn (2002).
14. Czarnecki K., Eisenecker U.: *Generative Programming: Methods, Techniques, and Applications,* Addison-Wesley, Boston (2000).
15. Dijkstra E. W.: *A Discipline of Programming,* Prentice Hall, Englewood Cliffs (1976).
16. Filman E. E., Elrad T., Clarke S., Aksit M.: *Aspect-Oriented Software Development,* Addison Wesley, Canada (2004).
17. Findler R. B., Flatt M.: *Modular object-oriented programming with units and mixins,* ACM SIGPLAN Notices, vol. 34(1). ACM Press, New York (1999) 94–104.
18. Fischer G., Redmiles D., Williams L., Puhr G., Aoki A., Nakakoji K.: *Beyond Object-Oriented Technology: Where Current Object-Oriented Approaches Fall Short,* Human-Computer Interaction, vol. 10. (1995) 79–119.
19. France R., Georg G, Ray I.: *Supporting Multi-Dimensional Separation of Design Concerns,* AOSD Workshop on AOM: Aspect-Oriented Modeling with UML, (2003).
20. Gradecki J. D., Lesiecki N.: *Mastering AspectJ: Aspect-Oriented Programming in Java,* Wiley, Canada (2003).
21. Harrison W., Ossher H.: *Subject-Oriented Programming: A Critique of Pure Objects,* ACM SIGPLAN Notices, vol. 28(10). ACM Press, New York (1993) 411–428.
22. Kiczales G. et. al.: *Aspect-Oriented Programming,* In Proceedings of the European Conference on Object-Oriented Programming (ECOOP). LNCS, vol. 1241. Springer, New York (1997) 220–242.
23. Laddad R.: *AspectJ in Action,* Manning (2003).
24. Mili H., Elkharraz A., Mcheick H.: *Understanding separation of concerns,* In Proceedings of the 3rd Workshop on Early Aspects, 3rd International Conference on Aspect-Oriented Software Development. Lancaster (2004).
25. Parnas D. L.: *On the criteria to be used in decomposing systems into modules,* Communications of the ACM, vol. 15(12). ACM Press, New York (1972) 1053–1058.
26. Wrycza S., Marcinkowski B., Wyrzykowski K.: *UML 2.0 in information systems modelling,* Helion, Warsaw (2005).

Proceedings of the International Multiconference on
Computer Science and Information Technology pp. 1021–1024
ISSN 1896-7094
© 2007 PIPS

Slicing wxHaskell modules to derive the User Interface Abstract Model

Daniela da Cruz[1] and Pedro Rangel Henriques[1]

University of Minho - Department of Computer Science,
Campus de Gualtar, 4715-057, Braga, Portugal
{danieladacruz,prh}@di.uminho.pt

Abstract. In this paper, we discuss an experience slicing an Haskell program to separate its GUI developed with the graphic toolkit wxHaskell. The slicing operation was implemented in Strafunski, an Haskell-centered software bundle for generic programming and language processing. To analyze the user interface and extract the abstract model, we first parse the source Haskell program, and build its Abstract Syntax Tree (AST). Then a strategic traversal of the AST is used to slice the GUI wxHaskell code, and build the graph of dependencies between widgets. In this way it becomes possible to understand the execution flow of the program.

Reusing and modifying legacy systems are complex and expensive tasks, because the required program comprehension is a difficult and time-consuming process. Thus, the need for methods and tools that facilitate program comprehension is urgent and strong. Reverse engineering aims at analyzing the software in order to represent it in an abstract format that makes easier to understand *what it does* and *how it works*. Reverse engineering tools provide means to support this task. Nowadays those software systems offer a rich interaction with the user, and their architectures are oriented towards a layered organization, separating at least the interface layer from the business core and data persistence modules.

In this context—system maintenance and comprehension of software with graphical user interfaces—we present the results of a research work on the application of *strategic programming* and *slicing techniques* to the reverse engineering of interactive functional applications. Holmes, the tool so far developed and described in the paper, is capable of deriving the *user interface abstract model* for interactive functional programs. Since wxHaskell became the most famous graphic toolkit to build Graphical User Interfaces (GUI) for Haskell, we have investigated slicing techniques specialized for wxHaskell modules.

In order to extract the user interface model from a Haskell/wxHaskell program we need to isolate the subset of functions responsible for the interaction. *Slicing techniques* are appropriate for that task. To build Holmes it is necessary to construct a slicing function [1, 2] that isolates the wxHaskell subprogram from the entire Haskell program. At a first glance, the obvious and easiest way is to define a recursive function to traverse the Abstract Syntax Tree (AST) of the program and return the wxHaskell subtree (maybe a forest). However, that approach forces the programmer to have full knowledge of Haskell grammar, and to write a complex and long set of mutually recursive

functions. *Strategic programming* is based on the use of a set of generic tree-walker functions that visit any AST, enabling the programmer to concentrate just on the relevant nodes; different traversal strategies (e.g. top-down, bottom-up, ...) can also be used. So this approach seems very convenient for our specific purpose.

Classic software systems have a *hierarchical graphical front-end* (the upper layer) that interacts with user reading his input, and writing system messages to him (results or general info). We can say that the interactive layer produces, deterministically, graphical output from *user or system events*, according to the system internal state. A graphical user interface (GUI) is built up from *widgets* (graphical objects), each one with a fixed set of attributes. At any time during the execution of the GUI these attributes have discrete values (the *widget properties*), the set of which compose the *state* of the GUI.

External (user originated) or internal (system originated) events are processed by the other layers and produce changes in the attribute values (alter the widget properties), implying the transition to a new state. So the behavior of the user interface can be described by a sequence of state transitions. A *dependence-graph*—a *directed-graph where nodes are that states, and events label the arcs*—is a perfect *abstract model* to describe GUIs.

To create that GUI abstract model from a given Haskell program we need: to identify *data entities* that denote widgets, and *actions* that denote events; and to discover the *relationships* that interconnect them. For that, we have defined a small set of abstract operations that describe the interactions between the user and the system, i.e., the events responsible for the state changes: *User Selection*–any choice that the user can make between several different options; *User Action*–an action that is performed as the result of user input or selection; *System Output*–any communication from the application to the user. Then we studied wxHaskell library to identify functions and data types that characterize the GUI module. Those identifiers will enable us to slice the user interface component in Haskell programs, isolating the tree nodes that correspond to the GUI. An Haskell GUI is described by types: `Frame` and `Button`, that denote graphical objects, widgets; `text` and `layout`, that represent attributes (functions that applied to widgets return values); `Event`, special attributes that can be transformed into an attribute using the `on` function (the value of an event attribute is normally an IO action that is executed when the event happens); etc.

After the contextualization of the our work and introduction of the basic concepts, we will describe the steps of our approach, and how they were implemented in Holmes.

First Step: Building the AST.

To extract the abstract model for a given source Haskell program, we build the respective AST. It will be our internal representation of the source system that we use in order to isolate the interface layer (wxHaskell sub-program) from the whole application (Haskell program). Any Haskell parser could be used to implement this phase build the AST. As there are many Haskell parsers available, we do not want to develop a new one from scratch. In this project, we use the libraries `Language.Haskell.Parser` and `Language.Haskell.Syntax`. At moment, only a subset (a realistic one) of wxHaskell components is being considered by the Holmes. Our first objective was to explore this approach and to assess its feasibility and usefulness. In the future, we will

extend our tool to handle more complex user interfaces. There is no reason to preview any kind of problems in that extension.

Second Step: Slicing Haskell with Strafunski

Given the AST, the next step is to isolate the wxHaskell sub-trees from the whole Haskell tree. As said in the beginning, we decided to follow a *strategic programming* approach to implement this phase, overcoming the disadvantages of a traditional approach to prune the tree.

Strategic programming is a generic programming style for processing compound data such as terms and object structures. Strategic programming was initiated in the setting of term rewriting, but has been transposed to other programming paradigms, most notably functional and object-oriented programming. With strategic programming, one gains full control over the application of basic actions, mainly full traversal control. Using a combinator style, traversal schemes can be defined, and actual traversals are obtained by passing the problem-specific ingredients as parameters to suitable schemes. In this style of programming, there is a set of generic traversal functions that traverse any AST. These strategic functions can traverse into heterogeneous data structures while mixing uniform and type-specific behavior [3]. In this project, Holmes was implemented with Strafunski library [4]: an Haskell library for generic programming and language processing.

As stated above, we have identified the set of types that will help us to recognize the *data entities*, the *actions* and the *relationships* between them necessary to build the GUI abstract model. A deeper study of the grammar, allowed us to know precisely what we need to look for in the list of nodes corresponding to the declarations in the AST. In order to prune the required nodes from whole tree and collect the correct information, we have written in Strafunski a strategy that performs a full traversal (`full_td`, i.e. visits every subterm), in a *top-down* way. To execute a type analysis, the strategy programmed is *type unification* (`applyTU` in Strafunski). The result of pruning the AST with this strategic function is a list of statements of type `HsGenerator`.

Third Step: Building the dependency-graph (the Abstract Model desired).

After slicing the AST, we are able to build the dependency-graph that will model the interaction layer of the given program.

To implement this phase in Holmes, we developed 3 functional strategies: To extract the source node, we explore the information from the first expression in the context of a `HsApp`, until we find a node with type `HsVar`; To extract the action, we look for the name of the widget that appears in the context of a `HsApp` (more precisely, in context of the first `HsVar`); To extract the target node, we consider the node that appears in the context of a `HsQualifier`, and we explore until we find a node of type `HsInfixApp` (but returning the right side of the expression).

This approach has proved to be flexible and broader enough to be applied in real cases of software engineering. The use of functional strategic combinators, make us able to extract not only the wxHaskell sub-program from a Haskell program, but also modules programmed with another GUI toolkits. Actually, we can generalize the functions above referred to other language constructors, by the reason that the basic concepts of strategic programming are independent of the programming paradigm.

Holmes prototype was tested with Haskell programs of small and medium complexity. In all cases, it produced the correct model.

References

1. Tip F.: *A survey of program slicing techniques*, Technical report, Amsterdam, The Netherlands, The Netherlands (1994).
2. Xu B., Qian J., Zhang X., Wu Z., Chen L.: *A brief survey of program slicing*, SIGSOFT Softw. Eng. Notes **30** (2005) 1–36.
3. Lämmel R., Visser E., Visser J.: *The Essence of Strategic Programming*. 18 p.; Draft; Available at `http://www.cwi.nl/ ralf` (2002).
4. Lämmel, R., Visser, J.: *A Strafunski Application Letter*, In Dahl V., Wadler P., eds.: Proc. of Practical Aspects of Declarative Programming (PADL'03). Volume 2562 of LNCS., Springer-Verlag (2003) 357–375.

Proceedings of the International Multiconference on
Computer Science and Information Technology pp. 1025–1036
ISSN 1896-7094
© 2007 PIPS

Pattern-based Program Visualization

Daniela da Cruz[1], Pedro Rangel Henriques[1], and Maria João Varanda Pereira[2]

[1] University of Minho - Department of Computer Science,
Campus de Gualtar, 4715-057, Braga, Portugal
{danieladacruz,prh}@di.uminho.pt
[2] Polytechnic Institute of Bragança
Campus de Sta. Apolónia, Apartado 134 - 5301-857, Bragança, Portugal
mjoao@ipb.pt

Abstract. The aim of this paper is to discuss how our pattern-based strategy for the visualization of data and control flow can effectively be used to animate the program and exhibit its behavior. That result allows us to propose its use for Program Comprehension. The animator uses well known compiler techniques to inspect the source code in order to extract the necessary information to visualize it and understand program execution.
We convert the source program into an internal decorated (or attributed) abstract syntax tree and then we visualize the structure by traversing it, and applying visualization rules at each node according to a pre-defined rule-base. No changes are made in the source code, and the execution is simulated.
Several examples of visualization are shown to illustrate the approach and support our idea of applying it in the context of a Program Comprehension environment.

1 Introduction

PCVIA, Program Comprehension by Visual Inspection and Animation, is a research project looking for techniques and tools to help the software engineer in the analysis and comprehension of (traditional or web-oriented) computer applications in order to maintain, reuse, and re-engineer software systems.

To build up a Program Comprehension environment we need tools to cope with the overall system, identifying its components (program and data files) and their relationships; complementary to those, other kind of tools is also necessary in order to explore individual components. These tools—that are our concern along the paper—deal with single programs instead of the complete set of programming units (the application), and their purpose is to extract and display static or dynamic data about a program to help the analyst to understand its structure and behavior.

Depending on the actual program facet we want to explore, different approaches to inspection and visualization can be followed. We are experiencing that in the context of PCVIA. We are developing a tool that does not modify the source program and uses abstract interpretation techniques, aiming at an easy and systematic adaptation to cope with different programming languages. In the Section 2 of this paper, we are going to discuss this approach and the generated visualizations. To attain such an objective, we parse the source program and build a decorated syntax tree and a symbol table,

that are kept in memory to support all the other subsequence operations (visualization and animation), while the program itself is discarded and is not compiled (it will not be executed in the target machine). Following this approach we implemented Alma, a program animation system.

We show some visualizations created by Alma. Alma facilitates the representation of abstract program concepts and can be useful in circumstances where there are not specific tools to visualize programs written in the language. The produced visual representation contains information about instructions and data, that will allow the user to get the perception of the program's execution behaviour and the changes in the value of variables.

1.1 Related work

During our study of the state of the art we found several software handling tools: classic Program Comprehension (PC) tools; software visualization tools that can be also seen as program understanding tools; development environments that use visual or textual representation to help the programmer; tools that are used just in some specific tasks of software maintenance; graph visualization tools that can be used for some program visualization tasks; and teaching tools.

Almost all of these tools were constructed for some specific language and are totally dependent of that language. Most of them use parsers automatically generated, and compiler techniques to process information. These parsers are used to transform the source code in order to instrument it with inspection functions or special data types. They can be also used to construct an internal representation of the program. This representation can be then systematically used to generate explanations, statistics, structured information, visualization or animation of programs.

Some examples of tools that create and use internal representation as the core of the tool are: Moose [1], CANTO [2] or Bauhaus [3]. In Moose (a reengineering tool) the information is transformed from the source code into a source code model. Moose supports multiple languages via the FAMIX languages independent meta-model. In almost all cases a parser is constructed to directly extract information to generate the appropriate model. The CANTO environment has a front-end (for C) which parses the source code and creates an intermediate file with structural, flow and pointer information. Then a flow analysis tool is used to compute flow analysis on the code. The front-end also creates an abstract syntax tree that is used by an architectural recovery tool which recognizes architectural patterns. The Bauhaus system has tools that use compiler techniques which produce rich syntactic and semantic information creating a low level representation of programs.

Alma follows this kind of approach — well-known language processing techniques are applied to visualization and animation. Alma uses a parser to construct an internal representation of the program and then uses a set of pattern based rules to inspect the code.

A first difference is that Alma can be easily prepared to cope with a new language; it is simply a matter of building a map between language concepts and Alma nodes, and nothing more is needed. Another difference is that Alma is not a tool to analyze an application (a set of modules) and extract information for its comprehension. Instead of

that, Alma aims at aiding to understand a program by visual inspection of its structure and by animation. Of course, one of its handicaps is that more complementary tools are needed to comprehend an application. Another disadvantage is that the visual representations can be not so beautiful as those produced by tools dedicated to a language or a problem class; but, on the another hand the system is more generic.

TKSee [4] or SeeSoft [5] are some examples of tools that collect statistic information about the source program and then this information is shown in a structured way without changing the source code. TKSee permits users to search the whole system for files, routines or identifiers whose name or lines match a certain pattern and build hierarchies to organize the information. SeeSoft also extracts statistical information from a variety of sources (like version control systems, programmer purpose and static and dynamic analysis) and shows the information using different coloured lines.

2 DAST Approach

In this section, we discuss the approach to program inspection and visualization followed in the context of Alma, one of the PCVIA tools under development. Although not a classic tool for program comprehension, we believe that it can truly contribute to this task, at the program understanding level (as argued in the Introduction).

Alma is a system for program visualization and animation. The purpose of such a family of tools is to help the programmer to inspect *data* and *control flow* for a given program (a *static view* of the algorithm realized by the program — **visualization**), and to understand its *behavior* (a *dynamic view* of the algorithm — **animation**).

The core of such tool is language independent; it is similar to a compiler's *back-end* that takes as input an abstract representation, and implements the visualizer and the animator components in a systematic way. To process a concrete programming language, the tool is specialized providing a dedicated *front-end* that converts the input programs into that internal abstract representation. As an intermediate representation, between the *front-end* and the *back-end*, we chose a DAST— Decorated Abstract Syntax Tree.

In this paper we do not want to introduce or explain Alma in detail; our purpose is just to discuss *the information we need to extract* from the source program, *how do we do it*, the format under which this *information is represented*, and *how is it visualized* to help the user to understand the program.

3 Patterns: the information to extract

In contrast to the most common animators, we are looking forward to building a more generic system, in the sense that it can animate any algorithm and that it can be easily adapted to work with different programming languages. To go in that direction, it is essential to find out a set of program patterns that we know how to deal with (display and rewrite). That is, we need to discover the information, common to a set of programming languages, that describes the structure and semantics of the program, and that we know how to store and to display (we intend to create a set of rules to systematically visualize those patterns).

An analysis of the programming languages, belonging to the universe we want to deal with, allows us to state that all of them have common entities, like: *literal values* and *variables* (atomic or structured), *assignments*, *loops* and *conditional statements*, *write/read statements* and *functions/procedures*.

After the common entities have been identified, we must find a way to describe them at an abstract level, in order to establish a generic set of rules to handle them in a language independent way. The solution is a set of *elementary programming patterns*.

In this paper, we consider that a **pattern** is a tree that represents an abstraction of a programming concept; it is composed by two parts: a structural component (given by a grammar production) and a semantic component (given by a set of attribute occurrences affected to the symbol that labels each tree-node).

These patterns capture the abstract syntax of each entity (value or operation) in order to preserve and keep, via attributes, the necessary information to express its static semantics.

This set of patterns can be compared with the instruction set of a machine. At compile time, the statements of a program, in the source language, are mapped into the proper instructions of the target machine (translation from high-level to low-level, or machine-level). In the same way, with our approach, the statements of a program are translated into patterns (in this case, from high-level to an abstract-level). For example, in an high-level language, the *reference to a variable* in an expression means the *access to its stored value*. The corresponding machine instructions have to load the variable value into the stack or accumulator; in the assembly language of a stack machine these instructions are something like: PUSH var_address, followed by LOAD. Similarly, in our approach, this operation will be mapped, in our internal representation, to the pattern that matches a variable; similar to assembly language, we get the value of this variable from the attribute where it was stored at parsing time.

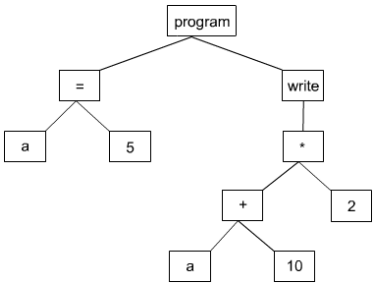
4 Program representation: Pattern Tree

Once we decided the information that we need to extract from a source language, we must now find a way to represent it. The internal representation chosen to store these patterns is a DAST [6][7] that describes the meaning of the program we intend to represent and visualize, being separated from any particularity of a source language. This DAST is specified by an abstract grammar independent of a concrete source language. This DAST is intended to represent the program in each execution point.
Consider the following program in some imperative source language:

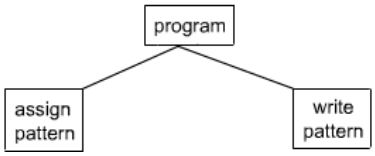
```
a=2;
write((a+10)*2);
```

Clearly, we have two different statements: an *assignment* and an *I/O statement (write)*.

One possible representation for it could be the syntax tree shown in Figure 1(a): Figure 1(b) shows the pattern tree (DAST) chosen in our approach. Each node in a concrete DAST will match and instantiate a specific pattern. These tree nodes are implemented with attributes, whose values are obtained during the information extraction phase, and describe the characteristics of the source program to preserve.



(a) Syntax Tree representation of the program



(b) Pattern Tree representation of the program.

Looking at Figure 1(a), we see that assignment node has two children—a *variable name*, and a *value*—and an implicit *type*. So, the corresponding DAST pattern will use three attributes: `name`, `value` and `type`. Bellow, we list the patterns considered in our approach, as well as some of the attributes used in each one.

- **Constants** — `value`, `type`;
- **Variables** — `name`, `value`, `type`;
- **Assignments** — left-side: `variable name`, right-side: `expression`;
- **Arrays** — particular case of variable, have one more attribute: `dimension`;
- **Conditional If/Then** — `boolean expression` (1) to evaluate, `set of statements` to execute in case (1) is true;
- **Conditional If/Then/Else** — `boolean expression` (2) to evaluate, `set of statements` to execute in case (2) is true, `set of statements` to execute in case (2) is false;
- **Loops** — `boolean expression` (3) to evaluate, `set of statements` to execute in case (3) is true;
- **Read** — (variable) `name`, `value`, `type`;
- **Write** — `expression`;
- **Functions/Procedures** — `table` for local variables, `arguments`, `set of statements`, `return value` (for functions).

The visualization and animation are internally supported by trees. At first, the *program tree* is constructed, representing a static visualization of the entire source program. Then, an execution tree is constructed representing the dynamic facet of the program. The rewriting and visualizing processes are applied precisely to this second tree; the first one will be only used as a repository of nodes. For example, when an instruction is executed three times, three instances of the corresponding nodes will be copied from the *program tree* to the *execution tree*.

In order to simulate the execution, all the pattern instances have one common attribute: `isEvaluated`. This attribute is mainly used to control the *rewrite process* (necessary for *program animation*) —it indicates if the tree had *already* been evaluated or *not yet*.

In next section, we show how we implement the patterns and how the DAST is built.

5 Pattern extraction and implementation

To extract information from a concrete source program its is necessary to parse it. This operation will be responsible for by a *front-end* built specifically for the concrete language under consideration. The *front-end* will be in charged of identifying the source language constructs and map them to the DAST patterns. To develop such a *front-end* we will use a compiler generator based on an attribute grammar. Our choice was LISA [8, 9]. LISA is implemented in the programming language Java, following an object-oriented approach either in its internal implementation or in the attribute grammar it accepts. To generate a *front-end* for a specific source language, we use the syntax and static semantics of that language specified by its grammar, and then we add to each production new attribute evaluation rules (`computing statements` in LISA's metalanguage) to build the internal representation of the corresponding DAST pattern.
For example, consider a grammar derivation rule (production) to define the assignment statement in some imperative language. Its definition in LISA's metalanguage using attribute evaluation *templates* is:

```
rule extends Assign
  {
    ASSIGN ::= DESIGNATOR \= EXPR \; compute
      {
        ALMA_ASSIGN_VAR<ASSIGN,
                        DESIGNATOR.name,
                        EXPR.value,
                        DESIGNATOR.type>
      };
  }
```

ALMA_ASSIGN_VAR is a *template* and has the three attributes mentioned in subsection 4 — the variable `name`, `value` and `type`.
Each template is previously defined in an Alma library and has the generic form shown below:

```
template<attributes X_in, Y_in, Z_in,...> compute NODETYPE
  {
    X_in.dast = new Node(Y_in, Z_in,...);
  }
```

Notice that NODETYPE identifies the type of the DAST node to be built corresponding to the pattern found (one of those listed in page 1029).

As we are using the LISA tool to automatically produce the extractor, we also decided to implement the patterns (subsection 3) reusing some LISA classes. It was very easy to identify and understand the data structures and methods used by the LISA system to process a given attribute grammar specification or a source program — they are properly encapsulated in classes with attributes and methods. So the coding of patterns became straightforward, due to the reuse of CSyntaxTree and CTreeNode classes to build the internal tree representation. As an immediate consequence, all the facilities provided by LISA to manipulate the attributed tree became available to process the DAST. This consequence made the development of the Alma *back-end* (another Java class that implements the visualizer and animator) much easier and faster; to code that class, we kept the object oriented approach followed in LISA.

6 Pattern visualization

At this point, we had already decided which information to extract, how to extract it, how to represent it, thus making how to visualize it is a natural consequence of the previous decisions.

Once we have a pattern tree as the intermediate representation between the *front-end* and the *back-end*, the DAST will be used to construct a visual representation for the source program. Each pattern will be extended with one additional attribute: `vr`, that contains the corresponding *visualization rule*. Thus, the visualization of a program is obtained by making a *top-down* traversal over the DAST, applying the specified rule to each node instance. The first traversal produces a picture of the entire program before execution. So, the animation of a program will be done by multiple *top-down* traversals to the DAST, until program is totally rewritten.

Figure 1 shows the visualization of a program, using the pattern tree corresponding to the source program shown in the bottom-left sub-window.

Figure 1 and figure 2 show the animation of a LISS program. LISS [10] is a language where all variables are initialized at declaration time (with explicit values or default ones). Figures 3, 4 and 5 (subsection 6.2) are related with C language.

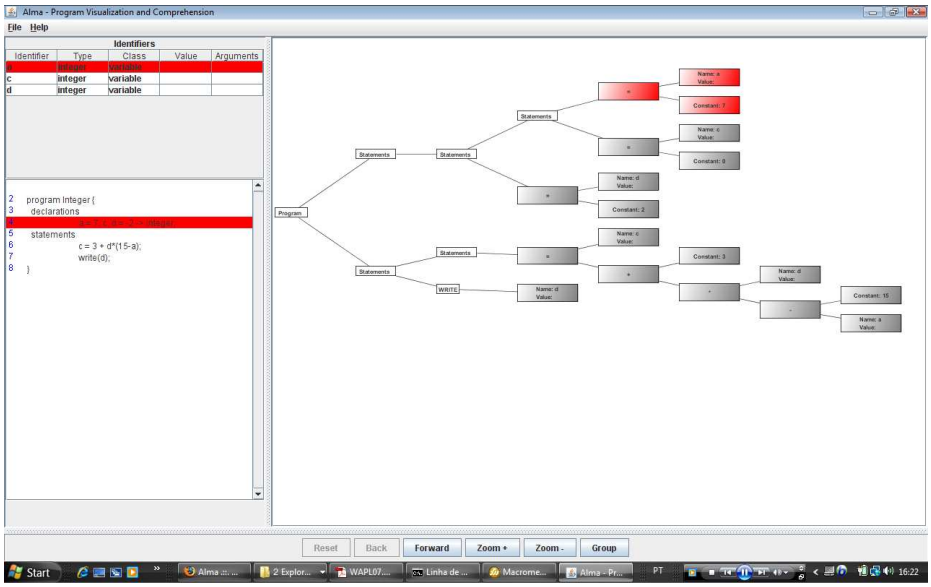


Fig. 1. Global visualization of the source program

6.1 Arrays and Structured Data Types

For arrays and structured data types we chose a different way to show their initialization. The initialization of variables of this kind is usually more difficult to understand; when a variable of one of those data types is initialized, a new sub-window is shown, giving the user the opportunity to see in detail the attribution of values to each component or to skip all the steps at once.

Figure 2 shows the sub-window to initialize a structured variable. After a variable is initialized in the respective row of the identifiers table, will appear a link "*See table*" to the current value of this variable. In the case of an array, will appear the values at each index (figure 2). In the case of a structured data type will also appear a local identifiers table.

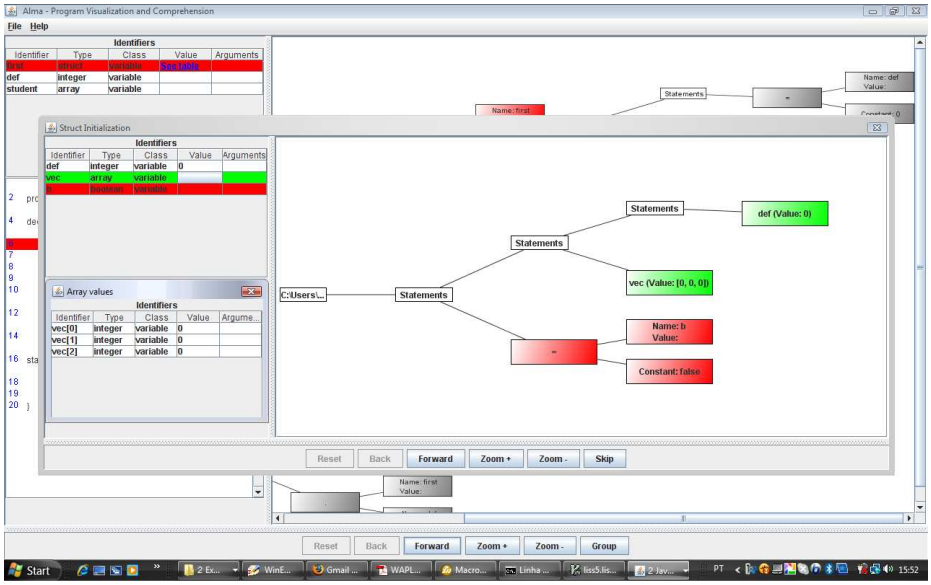


Fig. 2. Struct initialization and local table with values of an array variable

6.2 Function and procedures

To animate functions or procedures, a new window is opened each time the subprogram is invoked. This window is divided into parts: the first one is used to animate the parameters passing process; and the second one, to animate the execution of the function/procedure body. In case of subprograms without parameters, the first window (related with parameters passing) is omitted.

To illustrate the *function call mechanism*—suspending the execution of the invoker, evaluating and passing actual values to the function formal parameters, executing the

function body, returning a value and resuming the invoker execution—we include Figures 3, 4 and 5 that are concerned with the animation of a classic program written in C language. The program under consideration in this example is composed by two func-

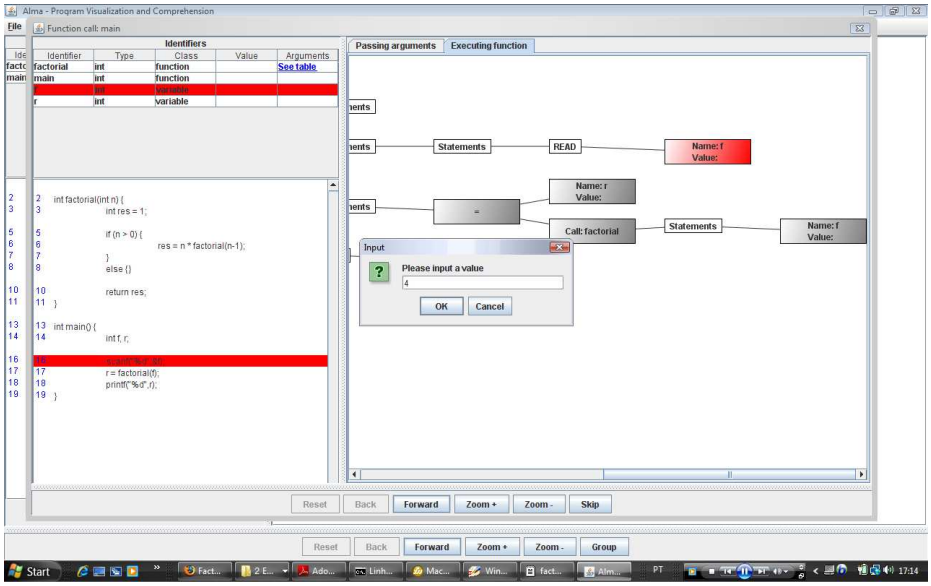


Fig. 3. Invoking the main function

tions: `main()` that prints the factorial of a given integer, invoking a function to do the computation; and `factorial()` that receives a parameter and computes recursively its factorial. The first screen displayed by Alma for this example, corresponds directly to the program tree, and shows that program global structure.

Automatically the Animator invokes function `main()`, opening a second window to show inside it the `main()` execution. Figure 3 is a screen-shot of the `main()` state corresponding to the execution of the `read` statement; notice the `input window` that appears in the middle of the screen to get a new value from the user. This picture also illustrates the mapping of a concrete C instruction, `scanf(%d,&f)`, into the Alma's abstract pattern `read`. The next two figures (Fig. 4 and Fig. 5) illustrate the invocation of `factorial()` function. The first describes the parameter passing (immediately after executing the `call` statement, a new window is opened and the animation of the evaluation and assignment of the actual parameter is displayed). The second (Fig. 5) corresponds to the execution of the third recursive invocation of `factorial()`. Notice that a new window is opened for each function invocation; a new identifier table and a new tree are displayed in order to animate that new execution process.

For each function, a local identifier table is created, and it is possible to map each row of this table to the visualization of the function body execution. When the function

execution ends, the local table disappears, and the return value is transferred to the previous identifier table.

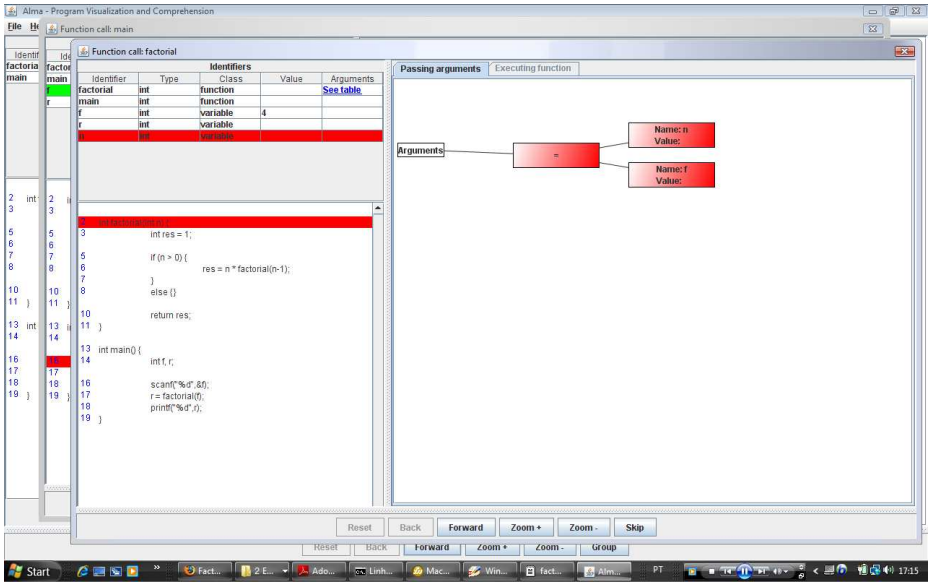


Fig. 4. Parameters passing to factorial function

7 Conclusion

To help the software engineer to understand the behavior of a given program (in the context of program comprehension environments), it is necessary to extract and collect from it *static data*—concerned with variable/type declarations and statement structure—and *dynamic data*—concerned with the data and control flows.

The objectives of our approach are two-fold: to show the program structure (the hierarchy of the statements); and to illustrate the execution flow and how it affects the program state. For that purpose we just have to parse the source program in order to: collect the information that defines its state (values and variables); and to find out its structure. A symbol table (or definition table) and an abstract syntax tree are enough to store this information. The visualization process is then performed by a systematic tree traversal, applying straightforward rules to each tree pattern, and to the correspondent row in the symbol table and line in the source text.

In our approach, we no longer need the source program; furthermore, after extracting information and building the DAST, we are able to give visual details helpful to get easily an *operational view* of the program. This approach does not modify the source program, and relies upon a visualization/animation engine (the Back-End of the tool)

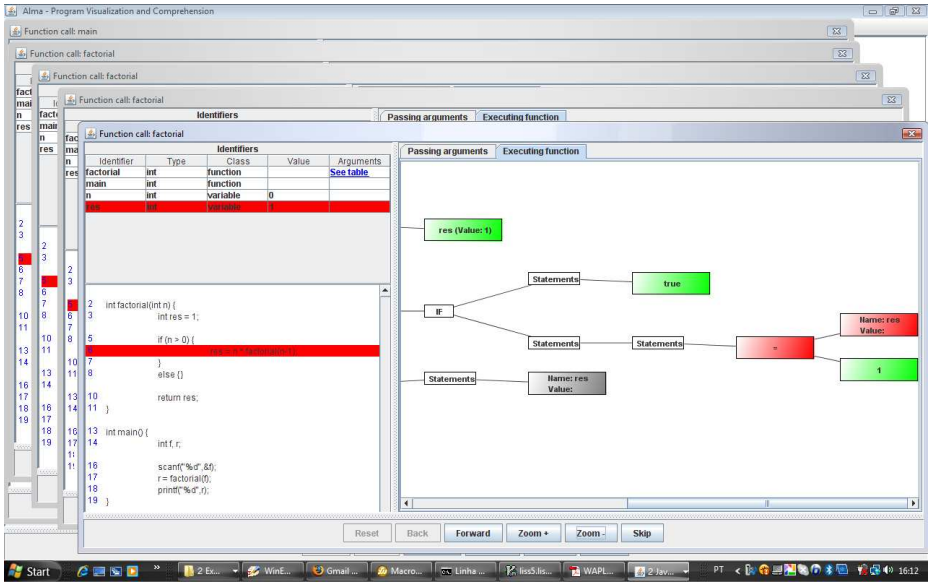


Fig. 5. Calling recursively the factorial function – Executing the function

that is independent of the source language (and, of course, of the algorithm); thus, tuning the tool to analyze programs in different languages is not an hard task.

In order to use Alma for a new language we just have to construct a *front-end* for that language. This *front-end* will map each source language concept to an Alma pattern. To start Alma development, we have created a *front-end* for LISS language, enabling us to begin the tests. Recently, we have followed a similar systematic process to construct another *front-end*, this time for C language. Using an attribute grammar (based on a public CFG for C) and LISA generator, this *front-end* was developed very fast.

We also believe that Alma can be very useful to visualize more declarative languages, like functional/logic programming languages, or specification languages, but we will work out this point as future work.

Hence Alma patterns correspond to the Turing machine basic operations, we argue they suffice to deal with the common imperative programming languages.

To cope with other paradigms, as referred above, possibly it will be necessary to upgrade the patterns library to include some others that can contribute to a more clear understanding of their specificities.

Alma system can be particulary useful for domain specific languages and other special languages that don't have any kind of PC tool implemented. For these languages a new PC tool would be constructed from the scratch. As we already said using Alma a specific PC tool can be easily prepared.

We are convinced that present Alma animations really help on program understanding — they show a program execution simulation with data and control flow information. However, this statement will be measured in the near future, via usability tests.

References

1. Ducasse S., Gîrba T., Lanza M.: Moose: an agile reengineering environment. In: ESEC-FSE'05, Lisbon, Portugal (2005).
2. Antoniol G., Fiutem R., Lutteri G., Tonella P., Zanfei S., Merlo E.: Program understanding and maintenance with the canto environment. In: IEEE International Conference on Software Maintenance (ICSM'97), Bari, Italy (1997).
3. Raza A., Vogel G., Plodereder E.: Bauhaus—a tool suite for program analysis and reverse engineering. Technical report, Department of Programming Languages, Institute for Software Technology, University of Stuttgart (2006).
4. Herrera F.: A usability study of the tksee software exploration tool. Master's thesis, University of Ottava (1999).
5. Eick S., Steffen J., Jr E. S.: Seesoft—a tool for visualizing line oriented software statistics. IEEE Transactions on Software Engineering **18** (1992) 957–968.
6. Moher T. G.: PROVIDE: A process visualization and debugging environment. In: IEEE Transactions on Software Engineering. Volume 14. (1988) 849–857.
7. Reiss S.: PECAN: Program development systems that support multiple views. IEEE Transactions on Software engineering (1985).
8. Mernik M., Zumer V., Lenic M., Avdicausevic E.: Implementation of multiple attribute grammar inheritance in the tool lisa. ACM SIGPLAN not. **34** (1999) 68–75.
9. Mernik M., Lenic M., Avdicausevic E., Zumer V.: Compiler/interpreter generator system LISA. In: IEEE Proceedings of 33rd Hawaii International Conference on System Sciences. (2000).
10. da Cruz D., Henriques P. R.: Liss—the language and the compiler. In: Proceedings of the 1.st Conference on Compiler Related Technologies and Applications, CoRTA'07—Universidade da Beira Interior, Portugal. (2007).

Proceedings of the International Multiconference on
Computer Science and Information Technology, pp. 1037 – 1039
ISSN 1896-7094
© 2007 PIPS

Machine Code Can Be Representation of Source Code With Optimization

Samir Ribić, Adnan Salihbegović

Elektrotehnički fakultet, Zmaja od Bosne bb,
71000 Sarajevo, Bosnia and Herzegovina
{samir.ribic,adnan.salihbegovic}@etf.unsa.ba

Abstract. In the recent times the authors of this paper have been doing research on possibility of developing programming language, which would be neither compiler, nor interpreter. The concept is based on holding complete program in native machine code, while the specialized editor can decompile machine code and display it in high level language. The displayed code can be reedited and saved again as pure machine code. This paper investigates the possibility of optimizing generated code, while still retaining decompilability features.

Rather then traditional approach for converting programs from high level languages to machine code using compilers and interpreters, the authors of this paper presented one more approach, in [1] and [2], where the source code does not exist as separate entity from executable (similarly to interpreted languages), while the program is executed in native machine code without aid of additional interpreter (similarity with compiled languages). The role of editor, compiler and decompiler is given to only one program, which displays the high level program representation from native code program. ASCII representation exists only during editing process.

Bidirectional transformation between high level language code and machine code is established by generating the code in such a way that it can always be decompiled. For example, to access the integer variable, in the native code the instruction LEA EBX,[EBP-nnnn] is executed, followed by MOV EAX,[EBX]. Each subroutine has special header, (this header does not slow down program execution), that provides information to editor about positions and data types of the variables on the stack. Hence, during decompilation for the purpose of listing or editing the program from the sequence mentioned, it is possible to restore that it is accessing the variable. From the offset nnnn it is possible to restore the variable name. Bidirectionally recognizable sequences are simple, but they are far from optimal when combined with larger language sentences. For example, the statement vb=va+2; in high level notation is represented in machine code as

538D1DNNNNNNNN8B03538D1DMMMMMMMM
8B0350B8020000005B03C35B890390

where NNNNNNNN and MMMMMMMM represent memory addresses of the global variables vb and va. In assembly code these instructions are

```
lea ebx,[vb]/ mov eax,[ebx] /push ebx/ lea ebx,[va] / mov eax,[ebx] /
push eax/ mov eax,2/ pop ebx /add eax,ebx/ pop ebx / mov [ebx],eax.
```

Such a long sequence results from translation symbol by symbol. When access to the variable is recognized, for addition it is enough to know value of the variable, for the assignment only the address is required, while pointer operator (*) needs both. Until we reach next symbol, the worst case sequence is generated. Another reason for longer code than necessary, is in requirement that functionally the same, but logically different sequences (like testing expression value in **if** and **while** statements) need to be represented differently in machine code, to make decompilation easier, even if this means a longer sequence than necessary.

One solution to this can be with the table which would contain sequences in shorter and expanded form, with the modification of the compiler part. In our example, the shorter sequence is `A1NNNNNNNN05020000008905MMMMMMMM`, meaning `mov eax,[va] / add eax,2 / mov [vb],eax.` Before the code is generated, it is first required to look into the table to find a shorter sequence equivalent to the originally planned generated sequence. If such a sequence exists, the compiler part of the editor produces shorter sequence. On the other hand, during code recognition, the decompiler part will first check if the sequence exists in the table, and if it does, it will repeat the recognition over larger version and therefore find the decompiled version. The most important sequences to be optimized are: assigning constant or variable to variable, arithmetic operations between two variable or variable and constant, negative constants, unary operations over variables, accessing to the array element, comparison of variable and constant (or two variables) in a condition part of **if**, **while**, **for** and **do/while** statements, comparison and assignments of two array elements etc.

Another problem is need for changing the editor itself if new optimization sequences are added. This problem can be solved by holding optimization sequences in a separate file (for example Windows.DLL, or Linux.so) which keeps track not only on optimized and non-optimized sequences, but also about versions of each translation. The version of the optimizer is saved somewhere inside user executable. For embedded systems which contain the compiling/decompiling editor in ROM, this table is an integral part of the editor.

References

1. Samir Ribić: *Concept and implementation of the programming language and translator, for embedded systems, based on machine code decompilation and equivalence between source and executable code,*. Proceedings 13th Working Conference on Reverse Engineering (WCRE 2006), 23-27 October 2006, Benevento, Italy. pp. 307-308
2. Samir Ribić, Adnan Salihbegović: *High Level Language Translator With Machine Code As Representation Of The Source Code,* Proceedings 29th International Conference on Information Technology Interfaces (ITI 2006), 25-28 June 2007, Cavtat, Croatia. pp. 777-782
3. Eldad Eilam, *Reversing: Secret of reverse engineering,* Wiley, 2005.

Proceedings of the International Multiconference on Computer Science and Information Technology pp. 1041–1053
ISSN 1896-7094
© 2007 PIPS

Executable form of IEC 61131-3 ST language programs in CPDev environment

Dariusz Rzońca[1], Jan Sadolewski[1], and Bartosz Trybus[1]

Rzeszów University of Technology, Division of Informatics and Control
Wincentego Pola 2, 35-959 Rzeszów, Poland
{drzonca, js, btrybus}@prz-rzeszow.pl,
Website: http://www.kia.prz-rzeszow.pl/

Abstract. A prototype compiler of the ST language (Structured Text), its operation and internal structure is presented. The compiler is principal part of CPDev engineering environment for programming industrial controllers according to IEC 61131-3 standard. The CPDev is under development at Rzeszów University of Technology. The compiler generates an universal executable code as final result. The code can be interpreted on different platforms by target-specific virtual machines. Sample platforms include AVR, ARM, MCS-51, PC.

1 Introduction

The main goal of the IEC 61131-3 standard introduced in 1998, was to increase quality of programmable controllers software [3]. By defining special programming languages and procedures it frees designers from using general purpose languages like C, focusing instead on implementation of control algorithms.

This paper describes a propotype environment for programming industrial controllers according to the IEC 61131-3 standard[1]. The environment, called CPDev (*Control Program Developer*) is built around the Microsoft .NET Framework [6], using C# language. Its main component is the compiler which transforms programs written in ST, one of five IEC 61131-3 languages [3] to a universal executable code. ST is a high level language, similarly to Pascal, Ada and C. Specification of the universal executable code has been prepared in such a way, that the resulting code can be executed on different target platforms, so small microcontrollers or larger microprocessors, via target-specific virtual machines. The machines operate as interpreters of the universal code. Generally speaking, it resembles the concept of Java virtual machines designed for implementation on different platforms [5]. The basic machine is written in industry standard C, for easy adaptation for various C compilers.

2 IEC 61131-3 standard and ST language

The IEC 61131-3 standard defines five programming languages – LD, IL, FBD, ST and SFC. *Instruction List* IL and *Structured Text* ST are text languages,

[1] This research has been supported by MNiSzW under the grant R02 058 03

Ladder Diagram LD, *Functional Block Diagram* FBD and *Sequential Function Chart* SFC are graphical. LD and IL are fairly simple, so appropriate mainly for small applications. FBD, ST and SFC are recommended for medium- and large scale projects. Kasprzyk's book [4] is a good source to learn IEC programming. According to [7], familiarity with the languages by engineering staff looks as follows: LD 90%, FBD 60%, IL 35%, ST 30%, SFC 15%. Computer and control engineers with experience in structural programming usually prefer ST.

Common components of the five languages are names (identifiers), data types, constants and variables. Twenty elementary data types of IEC 61131-3, together with memory sizes and ranges in the CPDev environment are shown in Tab. 1. In practice `BOOL`, `INT`, `REAL` and `TIME` are most common. Examples of corresponding constants are `FALSE`, `13`, `-4.1415` and `T#1m3s`. The IEC standard defines three access levels to variables, i.e. `LOCAL`, `GLOBAL` and `ACCESS`. `LOCAL` variables are available in the program or function block. `GLOBAL`s can be used in the whole project, but programs or blocks must declare them as `EXTERNAL`. `ACCESS` variables exchange data between different systems.

Table 1. Elementary data types of IEC 61131-3, their size and range in the CPDev environment

Data type	Memory size and range	Data type	Memory size
`SINT`	1B (-128 .. 127)	`TIME`	4B
`INT`	2B (-32768 .. 32767)	`DATE`	4B
`DINT`	4B (-2^{31} .. $2^{31}-1$)	`TIME_OF_DAY`	4B
`LINT`	8B (-2^{63} .. $2^{63}-1$)	`DATE_AND_TIME`	8B
`USINT`	1B (0 .. 255)	`STRING`	Variable length
`UINT`	2B (0 .. 65535)	`BOOL`	1B (0, 1)
`UDINT`	4B (0 .. $2^{32}-1$)	`BYTE`	1B
`ULINT`	8B (0 .. $2^{64}-1$)	`WORD`	2B
`REAL`	4B IEEE-754 format	`DWORD`	4B
`LREAL`	8B IEEE-754 format	`LWORD`	8B

Functions, function blocks and programs are components of the IEC projects. Function blocks, designed for multiple reuse in different parts of the program, are essential. Typical block involves input and output variables, and employs values from previous executions. The IEC standard defines a small set of standard blocks, such as flip-flops, edge detectors, timers and counters. Four of them are shown in Fig. 1.

Of the five IEC languages, ST is particularly suitable for implementation of nonstandard or complex algorithms. Most development systems recommend ST

as a default language for defining user function blocks. Therefore it has been chosen as a base language for the CPDev environment.

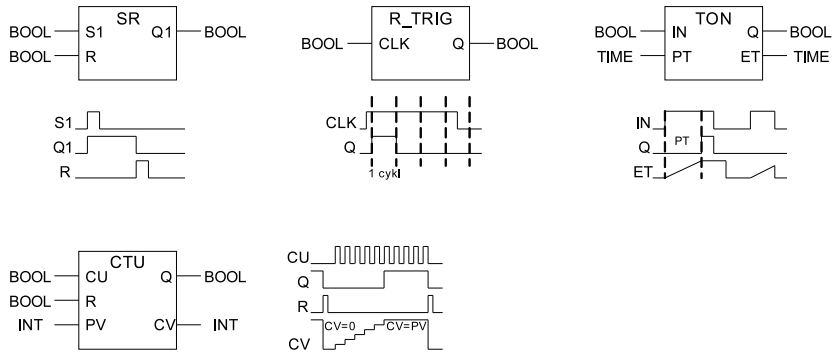


Fig. 1. IEC 61131-3 standard blocks: SR flip-flop, rising edge detector R_TRIG, timer TON, counter CTU

Initial part of ST program involves declarations of variables and block instances written between `VAR` and `END_VAR` keywords. The declarations are followed by a sequence of program instructions. The instructions contain expressions which involve operators such as (in descending priority): bracket, function call, negation, arithmetical operators, comparison, boolean operators. Similarly as in Pascal, the symbol `:=` denotes assignment. An example for starting or stopping an engine has the following form:

```
engine := (start OR engine) AND NOT stop AND NOT alarm;
```

Four types of control instructions are available in ST:

- conditional branches: `IF`, `CASE`,
- loops: `FOR`, `WHILE`, `REPEAT`,
- brake or stop: `RETURN`, `EXIT`, `END`,
- function block call.

Example of the last one may look as follows. After declaring `Timer1: TON` as an instance of the `TON` block (Fig. 1), one can call it by

```
Timer1(IN:=engine1, PT:=t#2.5s);
```

Output `Q` of the timer (Fig. 1) can be assigned to a variable in the following way

```
engine2:=Timer1.Q;
```

Alternatively, both input and output parameters may be included in a single function block call, i.e.

```
Timer1(IN:=engine1, PT:=t#2.5s, Q=>engine2);
```

3 The CPDev environment

Engineering environments are integrated tools for development, debugging, deployment and maintenance of control software [10]. Structure of the CPDev environment is shown in Fig. 2. It involves logical and physical layers what simplifies programming and compiling for different hardware platforms.

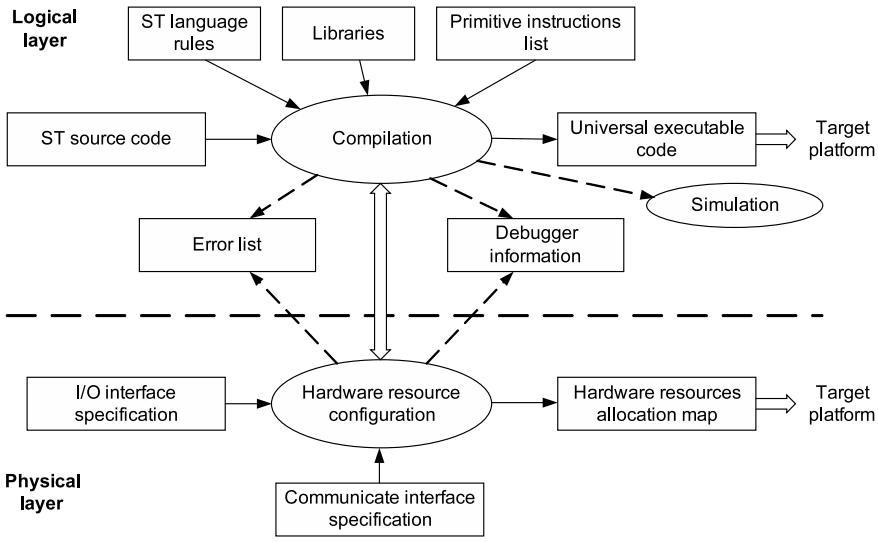


Fig. 2. Structure of the CPDev development system

ST compiler is an essential part of the logical layer. It translates ST programs into the universal executable code interpreted later by a virtual machine on the target platform. The compiler employs ST language rules, function block libraries and a set of primitive instructions implemented in the virtual machine. Single ST instruction is translated into one or several primitive instructions. Logical layer is also responsible for debugging, deployment information, simulation and list of errors.

Hardware resource configuration at the physical layer involves specifications of memory, input/output interfaces and communications. The specifications describe memory types and areas, inputs and outputs, addresses and types of communications channels, failure indicators, etc. Hardware allocation map (Fig. 2) is a table which assigns symbolic names from the ST program to physical addresses. By using this, the compiled code can be assembled for a particular platform to create final, universal executable code. Hardware platforms differ only in hardware allocation maps while compiled code is identical (before assigning addresses).

Three basic windows of the CPDev system user interface are shown in Fig. 3 [9]. They involve:

- trees of project structure, hardware configuration and program resources – left windows, interchangeable,
- program in ST language – center,
- message list – bottom.

The windows can be moved across the screen or minimized, frames adjusted and the contents scrolled. Window of the *StartStop* project (Fig. 3) contains the program `PRG1` which consists of `CTD`, `CTU`, `RS`, `SR` function blocks and the main task `DEF_TASK`. The main program involves global variables `START`, `STOP` to `OUT3` with addresses `%MX0000`, `%MX0001` up to `%MX000F`, respectively (directly represented variables). According to IEC 61131-3, the `%MX` prefix indicates variable stored in memory (`M`) and occupying a single bit (`X`). The first instruction of the program activates output `OUT0`, similarly as control of the engine in Sec. 2. Next, if `OUT0` is on, the outputs `OUT1`, `OUT2` and `OUT3` are activated every 2s. The `PRG1` program is executed every 200ms.

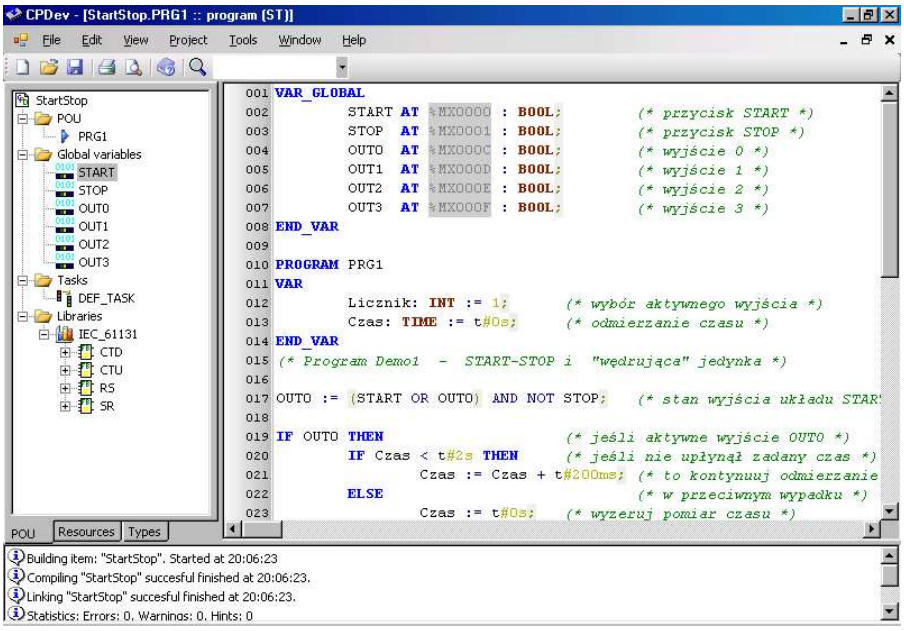


Fig. 3. User interface of the CPDev system

Hardware configuration tree, which would replace the project structure (left window) represents controller hardware of a controller (inputs/outputs, communication modules etc.) or a group of controllers, if a distributed system is

designed. Program resource tree contains lists of global and system variables, linked libraries and list of user-defined variable types (arrays, structured, etc.). `TASK_CYCLE` and `TIME_COUNT` are system basic variables.

In practice a typical ST-language program is a sequence of function block calls, where outputs from the previous block become inputs to the next one. Two libraries are available to the user in the CPDev system, i.e.:

- IEC 61131-3 standard blocks (Sec. 2 and Fig. 1),
- general purpose blocks (flip-flops, signal processing, on/off control, PID, set point generator, positioner, drive control, sequencer and some others).

Additional blocks can be programmed by the user and stored in additional libraries. To create a new block, one should begin with `FUNCTION_BLOCK` keyword in the upper window (Fig. 3). Blocks from all linked libraries can be accessed in the program. Simple programs of three standard blocks are presented in Tab. 2. The flip-flop `SR` and edge detector `R_TRIG` are self-explanatory (`CLKp` denotes previous value of `CLK`). Operation of `TON` has been illustrated in Fig. 1. Input `PT` denotes preset time, while output `ET` is the elapsed time starting from activation of the input `IN`. `ET` is incremented by `TASK_CYCLE`.

The CPDev environment uses the XML format for libraries, programs, configurations etc. in accordance with [11].

4 ST compiler components

The ST compiler contains three basic components i.e., *tokenizer*, *parser* and *code generator* (Fig. 4).



Fig. 4. ST compiler components

The tokenizer extracts tokens from input text (character stream). The parser is build according to top-down scheme with syntax-directed translation [2]. It recognizes the tokens and checks validity of token constructions. It is combined with the code generator which produces universal executable code for the virtual machine or intermediate mnemonic code, where primitive instructions are stored in a special text format. Modules of the mnemonic code can be collected together and converted into the executable format.

The tokenizer analyzes single characters in the input stream and groups them into tokens. Types of the tokens recognized by the compiler are shown in Tab. 3.

The tokens are stored in a list passed to the parser (Fig. 4). The parser utilizes built-in elementary data types, operators, and a set of primitive instructions of virtual machine. Examples of the instructions are shown in Tab. 4. Derived data

Table 2. Programs of SR, R_TRIG and TON standard function blocks

Function blocks: SR, R_TRIG	Function block: TON

FUNCTION_BLOCK SR
VAR_INPUT
S1: BOOL;
R: BOOL;
END_VAR
VAR_OUTPUT
Q1: BOOL;
END_VAR
 	
Q1 := S1 OR (NOT R AND Q1);
 	
END_FUNCTION_BLOCK
	FUNCTION_BLOCK TON
	VAR_INPUT
	IN: BOOL;
	PT: TIME;
	END_VAR
	VAR_OUTPUT
	Q: BOOL;
	ET: TIME;
	END_VAR
	VAR
	END_VAR
	
	IF NOT IN THEN
	Q := FALSE;
	ET := t#0ms;
	ELSE
	IF(ET < PT) THEN
	Q := FALSE;
	ET := ET + TASK_CYCLE;
	PT_x := PT;
	ELSE
	Q := TRUE;
	ET := PT_x;
	PT_x:= CUR_TIME();
	END_IF
	END_IF
	END_FUNCTION_BLOCK

FUNCTION_BLOCK R_TRIG VAR_INPUT CLK: BOOL; END_VAR VAR_OUTPUT Q: BOOL; END_VAR VAR CLKp: BOOL := FALSE; END_VAR Q := CLK AND NOT CLKp; CLKp := CLK; END_FUNCTION_BLOCK	

types [1, 6], functions and function blocks stored in additional libraries can be also included.

The parser generates text sequence of primitive instructions. Each instruction is represented by its mnemonic followed by operand names. Code generator replaces mnemonics and variable names with appropriate number identifiers (indexes). While processing an instruction, the generator extracts some information from libraries, e.g. operand size, type and passing method. Number identifier can be interpreted as a pointer to variable or as actual value.

The essential components of the compiler are designed as classes in the C# language [1]. Each unit of the ST language is encapsulated into an object of corresponding class (Fig. 5). The classes inherit from an abstract `STIdentificator` class. During compilation, identifiers are collected into lists. The lists employ predicates for finding appropriate identifiers and eliminate the need for hash

Table 3. Token types recognized by the compiler

Type name	Token example	Type name	Token example
identifier	`PRG1`	integer constant	`50`
keyword	`FUNCTION`	operator	`+`
typed constant	`DINT#1722211`	delimiter	`,`
comment	`(* assign outputs *)`	directive	`(*$READ*)`
real constant	`18.32`	white space	
string constant	`'Temperature: '`	invalid character	`\`

tables (normally used while developing compilers). There is a list of global identifiers and local lists which store identifiers of functions, function blocks, programs, etc. Identifiers in a list are checked for uniqueness. When identical names are found compilation is stopped and error reported. If local identifier hides a global one, the compiler produces an information.

Instructions for the virtual machine resulting from compilation are represented by instances of `VMInstruction` class. The operand list is also stored as a member of this class. By using lists of operands typical problems with fixed-size operand tables are avoided.

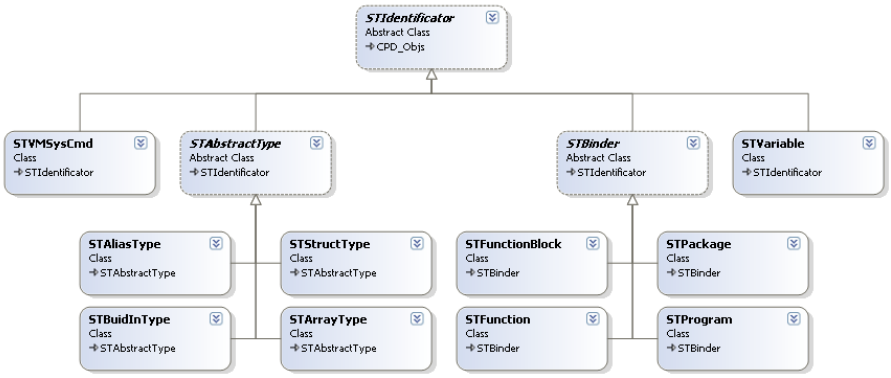


Fig. 5. Object representation of ST language units

Table 4. Primitive instructions of the virtual machine

Function	Meaning	Operator	Function	Meaning
EXPT	Power	**	OR	Logical or
NEG	Negation	- (unary)	XOR	Logical xor
SUB	Subtraction	- (arithm.)	SHL	Shift left
MUL	Multiplication	*	SHR	Shift right
DIV	Division	/	ROL	Rotate left
ADD	Addition	+ (arithm.)	ROR	Rotate right
CONCAT	String join	+ (text)	JMP	Unconditional jump
GT	Greater	>	JZ	Conditional jump
GE	Greater or equal	>=	JNZ	Conditional jump
LE	Less or equal	<=	JR	Relative jump
LT	Less	<	JRN	Conditional relative jump
EQ	Equal	=	JRZ	Conditional relative jump
NE	Not equal	<>	RETURN	Return from a function
AND	Logical and	&	MCD	Constant initialization

5 Compilation stages

The example [8] in Fig. 6 explains the process of compiling an ST program for the case of a function `FF`. The source program (upper left box) contains definition of `FF`, the universal executable code (lower right) for the virtual machine is generated as the final result.

In the first step the tokenizer splits input character stream into tokens. Then, the token list is passed to the parser which removes white spaces and comments. The parser also checks for invalid or unknown tokens and eventually reports an error, giving the file name and position of the bad token. If there are no syntax errors, the compilation begins. In the example of Fig. 6, the token `FUNCTION` is recognized as a keyword. The next token (`FF`) is treated as a new identifier and stored in the global identifier list as an object of `STFunction` type (Fig. 5). The returned value type (`REAL`) is also registered. The declaration clauses beginning with `VAR` are interpreted by the parser according to its syntactical diagram [8]. At that stage the compiler also defines memory areas for parameters, returned values and local variables.

Compilation of the function code section begins when next token does not match any of the declaration clauses. In the example the keyword `IF` indicates a conditional instruction. Therefore the text up to the keyword `THEN` is assumed to be a boolean expression. The result of evaluation of this expression is stored in an

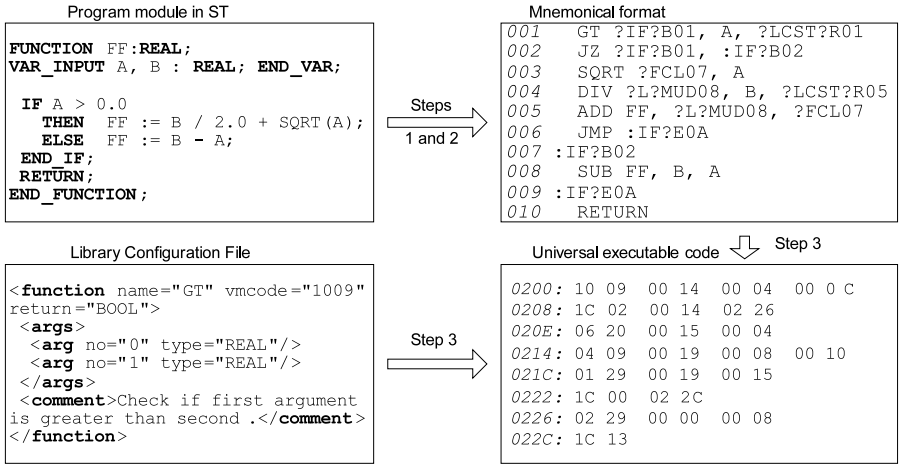


Fig. 6. Compilation steps of the function FF

auxiliary local variable `?IF?B01`, so with the name preceded by quotation mark `?`. This mark is used for every element of the code which has not been explicitly given any name. The expression between `IF` and `THEN` is evaluated according to Tab. 3. As a result, the compiler generates the line 001 of the mnemonic code (upper right box in Fig. 6). This line contains the primitive instruction `GT`, which compares operands (>). The line 002 involves conditional jump `JZ` to the part of the code labelled by `:IF?B02`. The jump is executed if `GT` returns `FALSE`. Otherwise, the lines 003 - 005 evaluate the variable `FF` using square root, division and addition. Unconditional jump `JMP` in the line 006 completes the evaluation. The line 008 contains subtraction `SUB` compiled from the block following `ELSE`. Label `:IF?E0A` is created as the result of `END_IF`. The last line with `RETURN` instructs virtual machine to finish execution of the function.

In the third step the final executable code is created. The compiler links the compiled code with all required modules. The mnemonic code is translated into binary executable by replacing:

- mnemonic function names with number identifiers (opcodes),
- variable names with data area indexes,
- label names with code addresses (absolute or relative).

The last step involves *Library Configuration File* (LCF; Fig. 6, lower left). This is an XML file with additional information required during compilation such as mnemonics of the primitive functions and opcodes, argument order and types. It can also contain some target-specific information, like big- or little-endian byte order. Resulting universal executable code is shown in the lower right box in Fig. 6. As seen, 1009 (hex) is the opcode of the first instruction `GT`. The others follow accordingly.

6 Virtual machine

Software deployment process can be represented as in Fig. 7. The universal executable code is transferred to a target controller where it is executed by the virtual machine.

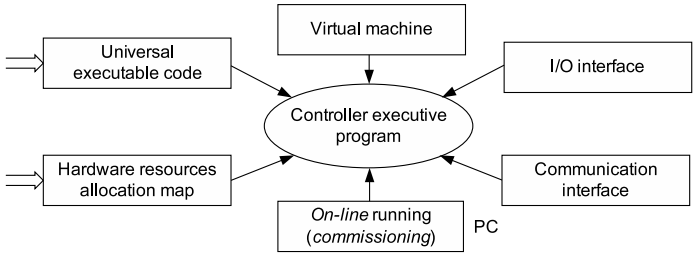


Fig. 7. Software deployment process for different hardware platforms

As indicated before, the machine is dedicated to a particular microprocessor and operates as interpreter. Harvard architecture is used due to security reasons, i.e. code and data are stored in two different memory areas. The virtual machine is an automaton operating according to Fig. 8. Following IEC 61131-3 standard, a task consists of programs executed successively. Universal code of the compiled program contains identifiers of primitive instructions and their operands (Fig. 6). While executing a program, the machine fetches successive instruction, decodes it, fetches the operands, and finally executes the instruction. The machine monitors time cycles of the tasks and alarms if timeouts appear. It also triggers input/output procedures responsible for external variables.

The virtual machine code consists of universal and target platform dependent modules (Fig. 9). The machine maintains program counter with the index to currently executed instruction and with base address to memory area with variables being used. Together with stack emulation, this allows for multiple and concurrent calls of functions and function blocks which employ internal variables. If a real-time multitask operating system is available on the target platform, the machine can execute multiple tasks simultaneously, providing so called *process image* (snapshot) of processed variables to avoid conflicts. The machines provide relatively short execution times due to similarity of primitive instructions interpreted by the machines (Tab. 4 before) to assembler of typical microprocessors, as well as indexing techniques used for interpretation [8].

The virtual machine has been written in the industry standard C language, so it can be directly adapted for different microprocessors. If limited hardware resources are available, an XML configuration file specifies a simplified version of the machine. For example, one can limit the number of elementary data types or define a subset of primitive instructions available. Till now, the machines for

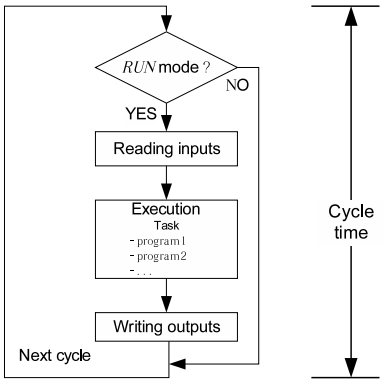


Fig. 8. Phases of the virtual machine cycle

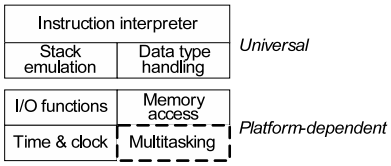


Fig. 9. Virtual machine internal structure

AVR, MCS-51 and PC platforms have been developed. Another one for ARM core-based microprocessors is being considered.

7 Summary

The basic component of the CPDev environment, the compiler of IEC 61131-3 Structured Text language has been presented in the paper. ST programs are translated to specially designed universal executable code. Target-specific virtual machine can execute such code on a controller CPU.

Although similar environments are available from commercial manufacturers, e.g. Step7 from Siemens, Control Builder from ABB, Concept from Schneider, they can handle only particular type of hardware from single manufacturer. The CPDev is oriented towards domestic manufacturers and supports, or will support, a range of popular platforms, such as AVR, MCS-51, ARM or PC. It has been created having in mind both universality and openness, which means that it can be adjusted to actual needs. First applications of the CPDev has already been tested in cooperation with LUMEL Zielona Góra for a central unit in a mini-distributed system for control measurement and monitoring.

References

1. C# Language Specification
   `http://msdn2.microsoft.com/en-us/vcsharp/aa336809.aspx`
2. Cooper K., Torczon L.: *Engineering a Compiler,* Morgan Kaufmann, San Francisco 2003.
3. *IEC 61131-3 standard: Programmable Controllers – Part 3,* Programming Languages, IEC 2003.
4. Kasprzyk J.: *Programming industrial controllers,* WNT, Warsaw 2006 (in Polish).
5. Lindholm T., Yellim F.: *Java Virtual Machine Specification Second Edition,* Java Software, Sun Microsystems Inc., 2004.
6. Ms .NET Framework Developer's Guide
   `http://msdn2.microsoft.com/en-US/library/aa720433.aspx`
7. Pietrusewicz K.: *Biggest study of Polish PLC controllers market,* Control Engineering Polska, pp. 26-38, Feb 2007 (in Polish).
8. Rzońca D., Sadolewski J., Trybus B.: *IEC 61131-3 standard ST compiler into universal executable code,* In: Real-Time Systems. Methods and Applications, WKŁ, Warsaw 2007, pp. 189-198 (in Polish).
9. Stec A., Świder Z., Trybus L.: *Functional characteristic of the prototype system for embedded systems programming,* In: Real-Time Systems. Methods and Applications, WKŁ, Warsaw 2007, pp. 179-188 (in Polish).
10. Trybus L.: *Engineering station functions of control systems for power industry,* VIII KA Conference, Rytro May 2004 (in Polish).
11. XML Formats for IEC 61131-3 ver. 1.01 – Official Release. `www.plcopen.org/`

Proceedings of the International Multiconference on Computer Science and Information Technology pp. 1055–1056
ISSN 1896-7094
© 2007 PIPS

Adapting Service-finding for Component Flexibility in Object-Oriented Languages

Dominik Dahlem and William Harrison

Software Structure Group,
Computer Science Department,
Trinity College Dublin,
Dublin, Ireland
{dominik.dahlem,bill.harrison}@cs.tcd.ie

Abstract. Developers assembling components into larger bodies of software generally make component selection choices early, and use a great deal of latent information in making those choices. In contrast, service-oriented software uses dynamic service-finding facilities, often called "service brokers". The involvement of brokers to provide a service implies that no assumptions can be made about latent information. The broker may supply any service having the characteristics being sought. Using late-bound selection with more information explicitly manifest is vital to coarse-grained business-to-business service interaction. But it would also be beneficial to allow for more flexible and dynamic use of components for in-house software, especially when improvement-over-time and contextual adaptation are important. This work explores applying the "new" operator to either interfaces or classes, to obtain the capabilities of a service broker. This allows conventional object-oriented software to derive the advantages that accrue to more flexible component selection being adopted for service-oriented software.

1 Introduction

Software systems undergo continual change. In most cases, this is due to growth in the availability of alternative choices and adoption of independently developed modules that go into the construction of such a system. But selection of implementation modules is generally made based on the set of modules available at the time a client is written, where the reality is that, over time, greater choice becomes available (e.g. Hashtable to HashMap). In practice, over time, collaborating modules will manifest a growing number of apparent dependencies, and will actually have a growing number of latent ones as well. Ongoing re-evaluation and division to use better alternatives is rare, and if the developers of these modules are responsible for resolving their dependencies an ongoing maintenance challenge is posed to the development team. In response, software developers have increasingly exploited practices, tool support, and language features to modularise software and loosen the coupling between the modules. However, it is generally true that developers assembling components into larger bodies

of software use a great deal of information in choosing the components that is not expressed in the description of their interfaces. This "latent information" makes it harder for later developers to understand the dependencies. On the other hand, developers often over-specify their dependence on a component by declaring more methods than are used or needed in the interface they bind to and thus impede on best practices to avoid highly coupled software structures.

2 A Language Construct for Late-Binding

To support class-finding, we suggest a single Java language extension allowing the "new" operator specify an interface, instead of being restricted to classes. To exemplify this language extension it is allowed to do `x = new Map()` instead of `x = new HashMap()` . In fact it is tolerated by the current definition of the Java class file format [1,2]. Effectively, the class-finder in the Java runtime would provide an object implementing this interface, deferring the decision to make use of available resources (i.e., in the classpath).

3 Implementing Late-Binding of Class Choice

In the Service-Oriented Architecture (SOA) realm, manifest information mostly is associated with the interface specification, which is supplied by the service consuming client. In a Java Virtual Machine (JVM) the facility most analogous to service-finding is the class loader. The responsibility of the class loader then is to resolve this symbolic reference and load the class into memory. The class has to pass all verification rules, including the integrity of the class file and correct inheritance hierarchy [1,2]. Like service-brokers used in service-oriented computing, Java class loaders lend themselves to achieving late-binding in Java applications due to their independence from the software applications. Applications using our proposed language construct require the `ClassFinder` to be configured with the JVM, while retaining their independence from the specifics of the class loader APIs. The `ClassFinder` intercepts all class resolution requests and selects one class implementing an interface from several possible classes present on the classpath. This mapping of interfaces to classes can be provided to the virtual machine as a static mapping generated with a tool and stored in JAR files or once when the JVM starts. Consequently, this extension would enable developers to treat their dependencies as abstract constructs, allowing the Java runtime to reason about the best match.

References

1. Sheng Liang and Gilad Bracha. Dynamic class loading in the java virtual machine. *ACM SIGPLAN Notices*, 33(10):36–44, October 1998.
2. Tim Lindholm and Frank Yellin. *The Java(TM) Virtual Machine Specification (2nd Edition)*. Prentice Hall PTR, April 1999.

Proceedings of the International Multiconference on
Computer Science and Information Technology pp. 1057–1069
ISSN 1896-7094
© 2007 PIPS

OORS: An Object-Oriented Rewrite System with Applications in Retargetable Code Generation and Optimization

Gernot Gebhard and Philipp Lucas

Saarland University, Dept. 6.2 Computer Science,
Building E1.3, Compiler Design Lab,
D-66041 Saarbrücken, Germany
[gebhard|phlucas]@cs.uni-sb.de,
Website: http://rw4.cs.uni-sb.de/~gebhard/projects/oors/

Abstract. Retargeting a compiler's back end to a new architecture is a time-consuming process. This becomes an evident problem in the area of programmable graphics hardware (graphics processing units, *GPUs*) or embedded processors, where architectural changes are faster than elsewhere. We propose the *object-oriented rewrite system OORS* to overcome this problem. Using the OORS language, a compiler developer can express the code generation and optimization phase in terms of cost-annotated rewrite rules supporting complex non-linear matching and replacing patterns. *Retargetability* is achieved by organizing rules into profiles, one for each supported target architecture. Featuring a rule and profile inheritance mechanism, OORS makes the reuse of existing specification possible. This is an improvement regarding traditional approaches. Altogether OORS increases the maintainability of the compiler's back end and thus both decreases the complexity and reduces the effort of the retargeting process. To show the potential of this approach, we have implemented a code generation and a code optimization pattern matcher supporting different target architectures using the OORS language and introduced them in a GPU compiler.

1 Introduction

As the number of different hardware architectures is steadily growing, easily retargetable compilers are most valuable. For instance, this applies to graphics processing units (GPUs). For the last few years the performance of GPUs has been increasing at a much faster rate than that of general-purpose processors and now exceeds the peak performance of high-end CPUs. The amount of transistors on graphics chips has been growing by a factor of 32 every two years [2]; 16 times faster than the CPU transistor count growth, which Moore's Law predicts. Thus, GPUs have become more and more interesting for general-purpose programming [17]. Several high-level languages, such as Brook for GPUs [3] or CGiS [14, 15], have emerged to exploit the vast computational power that GPUs have to offer. Easily retargetable compilers for these languages are necessary, because new

GPU architectures are released at a fast rate (e. g., NVIDIA's NV40 in 2004, G70 in 2005, and G80 in 2006). Naturally, the same applies to compilers supporting embedded systems, where a wide variety of different architectures with common heritage exists (e. g., Freescale's MPC555, MPC565, MPC755 and derivatives).

To decrease the complexity of the retargeting process and to keep the compiler maintainable in the long run, we propose the *object-oriented rewrite system OORS*. The idea of OORS originated from the fact that new architectures share many features with their predecessors, but offer an extended instruction set (apart from other new features). If a new architecture is released with minor differences to an already supported one, only small changes to an existing back end are required. So, the key feature of our approach is to be found in the *reusability* of existing specifications. We realized this by introducing object-oriented language features in the bison-like OORS language. Easily making the reuse of existing specifications possible, the proposed language enables a compiler developer to implement an OORS code generator or code optimizer which is less complex than a hand-written one. Consequently, the OORS implementation is much easier to read and to maintain in the long run. Our experiences in introducing new GPU architectures (NV40/G80) in the CGiS compiler supports this claim.

Basically, an OORS specification describes a pattern matcher that translates attributed input strings into attributed output strings. Given a set of rewrite rules, the pattern matcher processes the input string as follows. First the pattern matcher tries to match the possibly non-contiguous pattern of each rewrite rule against the input string. If multiple rules are applicable, the pattern matcher computes the non-constant costs of each rule to determine the rule to apply. Finally, the selected rule emits an attributed string that is appended to the output string. After the whole input string has been consumed, the pattern matcher terminates.

The aim of this paper is to introduce the OORS language, its syntax and semantics and to demonstrate its applicability in real word applications. Additionally, this paper briefly introduces the pattern matcher generator OORG that compiles an OORS specification into a C^{++} *dynamically retargetable* pattern matcher. We show its applicability by means of a real-life compiler for GPUs.

The remainder of this paper is structured as follows: Section 2 discusses related work. The object-oriented rewrite system OORS is introduced in Section 3. This section covers both the OORS language and the matching process. Section 4 introduces the pattern matcher generator OORG and demonstrates the integration of an OORS pattern matcher in a compiler. Section 5 concludes this paper and discusses future work.

2 Related Work

Numerous other approaches have been suggested for code generator generators. Below we discuss a representative set of such work. More systems are mentioned in [12].

Emmelmann et al. [7] propose BEG, a generator for efficient back ends. Using the description language BEGL, the developer implements tree-pattern matchers for code generation in terms of cost-annotated rewrite rules. In contrast to the OORS language, BEGL offers no rule-inheritance mechanism. The reuse of existing specifications is thus not possible. Additionally, a BEG pattern matcher is dedicated to a single target architecture only. The major difference is to be found in the processed input data. BEG code generators process *trees*, whereas OORS pattern matchers operate on instruction *sequences*.

In [11] Fraser et al. introduce the code generator BURG for the bottom-up rewrite system BURS which is similar to BEG. BURG is able to generate tree-pattern matchers for fast optimal instruction selection. A BURG-generated tree-parser is able to find an optimal parse of an input tree in linear time. As in BEGL, the BURG grammar does not feature any mechanisms that makes the reuse of existing specification possible. Additionally, BURS code generators are only able to generate code for a single architecture only. BURG-generated code selectors are used in the ANSI C compiler lcc [10].

Ferdinand et al. [9] solve the code selection problem with deterministic finite tree automata that are generated automatically from regular tree grammars. In contrast to BURG, the left-hand and the right-hand side of rules are not limited to leafs or nodes with one or two child nodes. In contrast to OORS, the costs of a rule must be constant and thus cannot depend on the matched instructions. Similar to BURG, the developer cannot inherit rewrite rules from each other to easily copy reusable properties. As in the other approaches discussed above, it is not possible to target multiple architectures. Deterministic tree automata have been successfully tested for several architectures, but are not employed in any existing compiler – at least not to our knowledge.

In [1] Alt et al. propose the CoSy model, which provides a framework for flexible combination and embedding of compiler phases to ease the construction of parallel and optimizing compilers. Using three different languages, the compiler developer can implement the different phases of a compiler on a high level of abstraction. Additionally, the developer is able to specify the control flow and the interactions of the compiler phases. Existing implementations of compiler phases can be simply reused. However, if any modifications to the implementation are required, the framework does not prevent copy-and-pasting of that implementation before the developer can change the code. Just as in the approaches discussed above, the code generator is based on tree-pattern matching. A comparison with OORS is hardly possible, because OORS is designed to only implement the code generation and code optimization phase.

Dias and Ramsey [6] propose a recognizer for machine-independent code selection and code optimization. A recognizer is generated automatically from a declarative machine description that describes properties of the target platform. The generated recognizer requires the compiler to represent intermediate code in the form of machine-independent register-transfer lists (RTLs) [5]. By means of a declarative machine description, the recognizer tries to generate better RTLs. The recognizer will continue until no more optimizations can be applied. The

recognizer omits a previously generated RTL, if the new RTL cannot be implemented on the target platform according to the machine description. The authors have successfully generated and tested a recognizer for the x86 back end in the Quick C-- compiler [4]. This approach differs greatly from OORS pattern matchers, as the developer does not have to explicitly implement the code optimizer. The effort is shifted to implement a complete machine description.

Farfeleder et. al [8] describe a similar approach. By means of a new architecture description language (ADL), the authors are able to derive an optimized tree-pattern matching instruction generator, a register allocator and an instruction scheduler. To demonstrate the applicability of the new ADL, the authors have implemented an ADL-generated compiler for the xDSPcore digital signal processor. Again, the effort is shifted to implement a complete machine description.

In [13] Lerner et al. introduce Rhodium, which is a new language for compiler optimizations, whose soundness can be automatically proven. The developer specifies optimizations in terms of transformation rules that are automatically proven to be semantics-preserving. Rhodium optimizations are not bound to a specific target architecture, because they process input programs transformed into a C-like intermediate language. In this way, the optimizations are automatically retargetable. However, the main goal of their approach are the automated soundness proofs of the compiler optimizations. It is unclear how Rhodium optimizations compete with OORS optimizations, because OORS operates on the instruction level. Apart from that, the approach is out of scope for this paper, because OORS was not designed for automated soundness proofs.

OORS differs in many ways from the approaches presented above. One difference concerns the way in which the *subjects of pattern matching* are represented: OORS operates on sequences of instructions, not on trees. This is because OORS is also employed in the code optimization phase. By representing the subject of matching as an instruction sequence, scheduling properties can be expressed alongside with other low-level optimizations (see Section 3.3). Another important point is that OORS features object-oriented language constructs that make the *reuse* of existing specifications easily possible. Apart from CoSy [1], none of the presented approaches was designed with reusability in mind.

3 Object-Oriented Rewrite System

In this section, we introduce the key concepts of the OORS language and discuss the pattern matching process. For the sake of simplicity, we only concentrate on code *generation*. Section 3.3 briefly discusses the changes to the matching process required to realize code *optimization*.

3.1 Rules

As hinted in Section 1, an OORS pattern matcher processes attributed strings, which are sequences of instructions. We assume that each instruction is an in-

stance of a class of the compiler's internal representation (e. g., a binary instruction could be an instance of the *BinaryInstruction* class) with a common base class. The available attributes of each instruction object are then defined by the member functions of the corresponding class. So to speak, OORS rewrite rules determine transformations on sequences of instruction objects. The behavior of each rule is determined by the following four aspects:

search pattern: The search pattern determines constraints on the structure of the input that must be fulfilled before the pattern matcher may apply the rule. A search pattern is a non-empty, ordered or unordered, possibly discontinuous sequence over instruction classes. By using *wildcard patterns*, the developer can specify search pattern with an arbitrary lookahead.

Each symbol of the search pattern may be guarded by a local side condition, which is simply a boolean expression. Using local side conditions, the developer is able to specify non-linear search patterns. For instance, a local side condition could check whether an operand of the currently matched instruction and the target of a previously matched instruction are of the same type (see Example 3).

condition: The condition corresponds to the local side conditions introduced above. The main difference is that the developer implements a boolean function instead of a single boolean expression. An undefined condition function is assumed to return `true`. Condition functions come in handy, when deriving rules from each other (see Section 3.2).

costs: The cost function associates a weight to each matched instance of the search pattern. The computed costs need not be constant and may depend on the matched instruction objects. If multiple rules match the input sequence, the pattern matcher determines the rule to apply according to the associated costs.

replace pattern: The replace pattern determines the generated instruction sequence that the pattern matcher appends to the output sequence when applying the rule. Each element of the replace pattern corresponds to a constructor call of the instruction class with appropriate arguments. It is possible to access previously generated instructions when initializing a new instruction object.

Simple Rules Example 1 demonstrates how to specify a simple code generation rule that compiles a binary instruction for which the target architecture has a direct counterpart. The rule matches any binary instruction object. Thus, there is no need to implement a rule for every single binary instruction. This keeps the specification both readable and maintainable in the long run.

Example 1 (Rule that compiles any abstract binary instruction into its counterpart). Matched instructions objects can be accessed via the `$$` and `$i`-operators like in yacc and related tools. The `$i`-operators enumerate both the matched and the generated operations uniformly.

```
rule binary {
  search:  [ BinOp ]
  cost:    { return 1; }
  replace: [ GPUBinOp($1->opcode,$1->tgt,$1->op1,$1->op2) ]
}
```

Generic rules like the one shown in Example 1 are not always sufficient: Some instructions may require special rules. Example 2 shows how to compile the exponentiation operator for recent GPU architectures.

Example 2 (Code generation rule for a special unary operator). Instruction sets of recent GPU architectures do not feature an exponentiation operator. Instead, their instruction set contains the `EX2` instruction, which computes powers of two. By using the identity $2^x = e^{\ln(2)x}$, we are able to express e^x on the GPU with the rule `exp`. Note that OORS allows the developer to access generated instructions. This is required to initialize the second generated instruction. In this context, the developer can access the target of the generated multiplication instruction via the `$2`-operator.

```
rule exp {
  search:  [ UnOp($$->opcode == OP_EXP) ]
  cost:    { return 2; }
  replace: [ GPUBinOp(OP_MUL, SymReg(TYPE_FLOAT), $1->op,
                      Constant(TYPE_FLOAT, 1/ln(2))),
             GPUUnOp(OP_EX2, $1->tgt, $2->tgt) ]
}
```

Complex Rules In general simple rules as demonstrated above are not sufficient. In some cases, the developer might want to match instructions objects that are not necessarily adjacent to each other in the input instruction sequence. Example 3 demonstrates a typical case.

Example 3 (Complex code generation rule). GPU architectures feature a combined sine-cosine instruction `SCS` [16]. From a single operand c, the instruction writes $\sin c$ and $\cos c$ into two register components. The following rule combines sine and cosine instructions in the intermediate representation into a single `SCS` operation. The *wildcard pattern* (`*`-pattern) denotes that the two instructions need not be adjacent to each other. The curly braces in the search pattern indicate an *unordered* sequence: The instructions may be matched in any order.

```
rule scs {
  search:    { UnOp($$->opcode == OP_COS),
               *,
               UnOp($$->opcode == OP_SIN) }
  condition: { return $1->op == $3->op; }
  cost:      { return 1; }
  replace:   [ GPUUnOp(OP_SCS, SymReg ($1->tgt, $3->tgt),
                       $1->op) ]
}
```

Figure 1 illustrates the effect on the input and output instruction sequence after applying the rule `scs`. The figure demonstrates that the pattern matcher must not apply the rule in every case. For instance, if an instruction in-between the matched instructions modifies the operand or the target of the second matched instruction, the transformation will most likely modify the semantics of the input program and is thus invalid in general. Note that every rule with at least on wildcard pattern in its search pattern is subject to this negative side-effect. Thus, a special, semantics-preserving check is required.

The OORS language enables the developer to implement such a semantics-preserving check. The developer implements a *global implicit condition*, which decides whether the rule in question may be applied. It does so by checking for data dependencies between the matched instructions, which would prevent a reordering. This implicit condition is only checked for rules whose search pattern contains a wildcard pattern.

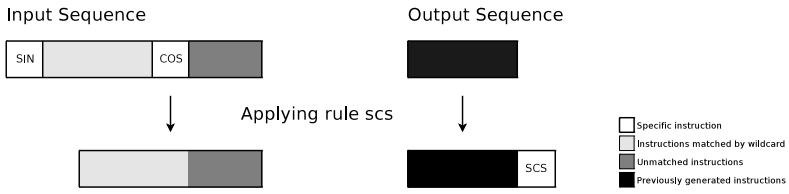


Fig. 1. Input and output instruction sequence before and after applying the rule `scs`.

Under certain circumstances, it might not be sufficient to generate the same sequence of instructions all the time. Some instructions might have to be translated into different instruction sequences depending on the type of their operators or similar side conditions. For this reason, the OORS language allows the developer to guard any sequence of instructions to generate via `if-then-else` statements. This enables the developer to integrate all possible alternatives into a single replace pattern, which keeps the pattern matcher specification readable. Example 4 shows the usage of guards within replace patterns.

Example 4 (Rule that generates different instruction sequences for the same instruction type). On GPU architectures, each register is a float vector comprising four components (called `r g b a`). In the source language, and thus in the intermediate instructions, operations work on scalars or on vectors with a length of at most 4; in general, the native arithmetic instructions also support such vectorial operations. However, some instructions operate only on scalar operands. Thus, special treatment is required if an operand of the correlative abstract instruction is of vector type. To resolve this problem, a sequence of the same scalar instructions has to be generated for each vector component. The rule `sin` generates code for the `SIN`-instruction, which computes the sine of its operand. When generating the corresponding GPU code, the rule has to make sure to select the

correct vector components (e. g., *cmp('r')* directs the instruction object to read from and write to the `r`-component).[1]

```
rule sin {
  search:  [ UnOp($$->opcode == OP_SIN) ]
  cost:    { return $1->target->components; }
  replace: [ if ($1->useComponent('r')) [
               GPUUnOp($1->opcode, $1->tgt, $1->op, cmp('r'))
             ],
             if ($1->useComponent('g')) [
               GPUUnOp($1->opcode, $1->tgt, $1->op, cmp('g'))
             ],
             if ($1->useComponent('b')) [
               GPUUnOp($1->opcode, $1->tgt, $1->op, cmp('b'))
             ],
             if ($1->useComponent('a')) [
               GPUUnOp($1->opcode, $1->tgt, $1->op, cmp('a'))
             ] ]
}
```

To further improve the maintainability and compositionality of the pattern matcher specifications, the OORS language introduces the notion of *intermediate replace patterns*. The developer specifies a replace pattern not on the instruction set of the target architecture, but on the intermediate instructions.[2] These intermediate instructions are then subject to the matching process as usual. The benefit of the intermediate instructions is to allow *compositional* specifications. For example, a vectorial `tan` instruction has to be implemented by sequences of native `sin` and `cos` instructions just like in Example 4, followed by a division; a much more complicated way than just specifying an *intermediate level* `sin-cos-div` sequence and letting the matcher then generate the component-wise native instructions from these intermediate instructions.

3.2 Rule Sets

An OORS pattern matcher specification comprises rules, which are organized into *profiles*. A profile represents a set of rules dedicated to a specific target architecture. During runtime, the host compiler (the compiler using the generated code-generator) selects the corresponding profile to be used for processing the input program(s). In this section, we first describe the matching process with respect to a single profile. Afterwards, we discuss the specifications mechanism concerning multiple profiles and their relationships.

Matching For a given profile, the pattern matcher tries to match the input instruction sequence against the search patterns until every input symbol has

[1] The components need not be consecutive.
[2] This kind of rule is specified by the keyword `intermediate` instead of `replace`.

been covered. In a *greedy* matching mode, the input is processed left-to-right, where always the rule with the lowest costs is selected. In case of a tie, the first specified rule is chosen. The pattern matcher employs backtracking in case the current match cannot be enlarged further while some input instructions remain unmatched.

In contrast to the greedy matching, an *optimal* matching mode investigates all possible matches to select the one with the globally optimal costs. If costs are not negative, we do not need to exhaustively explore the search space, but can prune the search space as soon as it can be determined that the current, incomplete match cannot outpace a previously found match.

Inheritance To keep an OORS pattern matcher specification readable and thus maintainable in the long run, the OORS language comprises rule and profile *inheritance* mechanisms. The developer can derive a new profile from existing ones. The new profile inherits all rules, may add new rules and may omit and modify inherited rules. Example 5 demonstrates the profile and rule inheritance mechanism.

Example 5 (Profile and rule inheritance). The source language features a dot-product operating on vector operands. The target architectures support the dot-product for vectors of three and four components. The newer G80 architecture supports also the dot-product for vectors of two components, whereas before it had to be realized using multiplication and addition. Thus, the NV40 profile comprises two distinct rules to cover all kinds of operands.

```
profile NV40 {
  rule dp2 {
    search:    [ BinOp($$->opcode == OP_DP) ]
    condition: { return $1->target->components == 2; }
    cost:      { return 2; }
    replace:   [ /* MUL, ADD */ ]
  }
  rule dp : extends dp2 {
    condition: { return $1->target->components > 2; }
    cost:      { return 1; }
    replace:   [ GPUBinOp(OP_DP, $1->tgt, $1->op1, $1->op2) ]
  }
}

profile G80 : extends NV40 {
  omit NV40::dp, NV40::dp2;
  rule dp : extends NV40::dp {
    condition: { return $1->target->components > 1; }
  }
}
```

The G80 profile is specified as an extension of the NV40 profile. So, it inherits per default all NV40 rules. The G80 profile omits the two NV40 rules `dp` and `dp2` and specifies a new, general rule as a modification of the inherited `dp`-rule.

3.3 Optimization

The pattern matching approach can also be used for code optimization. In contrast to the generational mode discussed in Section 3.2, there is only one sequence upon which the matching is performed. The pattern matcher searches for instances of the search patterns and literally replaces the matched instructions with the generated instructions. This process is repeated until no more rules can be applied, that is, until a fixpoint has been reached. Thus, there is no need for backtracking.

In this way, it is possible to represent low-level code optimizations in OORS, such as instruction rescheduling (see Example 6, Figure 2) or instruction merging (such as `MUL` and `ADD` into `MAD`).

Example 6 (TriCore instruction rescheduling optimization). On many recent architectures, the instruction order has a major influence on the execution time. For instance, the TriCore architecture [18] is only able to dispatch two instructions at once, if the first instruction will be executed in the arithmetic-logical unit (ALU) and the second instruction will be issued to the load-store unit (LSU). So, the following rule tries to pull a distant memory instruction behind an arithmetic-logical instruction. Although the implicit condition will verify whether possible side effects occur, the developer has to check manually whether it is safe to push the memory instruction in front of the second matched instruction (see the `condition` line; the absence of conflicts with the wildcard is guaranteed by the implicit condition). The cost function favors the match with the greatest distance between the ALU- and the LSU-instruction (for a wildcard pattern, the `$`-operator denotes the number of instructions the pattern has matched). Figure 2 shows this graphically.

```
rule reschedule {
  search:    [ Op($$->isIssuedTo(ALU)),
               Op(!$$->isIssuedTo(LSU)),
               *,
               Op($$->isIssuedTo(LSU)) ]
  condition: { return !$2->conflictsWith($4); }
  cost:      { return -$3; }
  replace:   [ $1, $4, $2 ]
}
```

4 Practice

The integration of OORS pattern matchers works in much the same way as for tools such as yacc or flex (see Figure 3). From the pattern matching specification, OORG generates a C^{++}-file implementing the matcher. This file has to be compiled and linked with the main compiler. The compiler selects at runtime the profile to be used in a particular compilation and calls the generated matcher for each basic block, passing the sequence of intermediate instructions and receiving

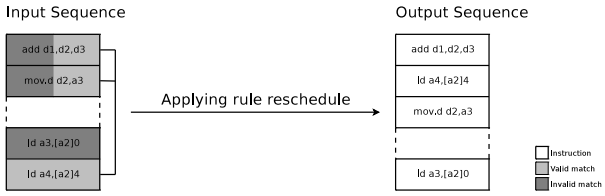


Fig. 2. Valid and invalid match of the `reschedule` rule. The first match (dark-gray) is invalid, because there is a definition-use dependency between `mov.d d2,a3` and `ld a3,[a2]0` (the first instruction reads from and the second instruction writes to register `a3`). The rule's condition prevents this match from being accepted.

the sequence of target instructions. To easily cater for minor variations of the desired rule set, the application program can switch on and off certain rules. For example, different sets of optimizations can be selected in this way.

The CGiS compiler [14, 15] can compile a common input program for GPUs and for SIMD CPUs. OORG-generated matchers are employed in three phases; the actual matchers used in a compilation are selected at runtime depending on the actual target. An early optimizer performs various transformations on intermediate code, which are needed for implementation on SIMD CPUs. A generator transforms the sequence of of intermediate instructions, which are common for GPUs and SIMD CPUs, into a target-specific instruction sequence. To this end, there are three hierarchies of profiles: For various generations of GPUs, SSE, and AltiVec. In a later phase, the GPU code is transformed by an OORG-generated peephole optimizer.

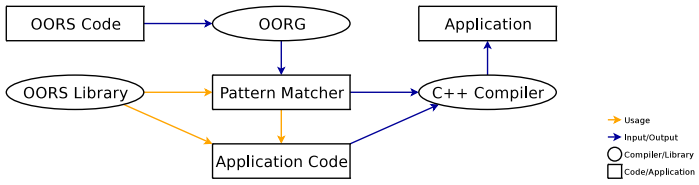


Fig. 3. Integration of OORS into an existing application.

We have implemented an NV40 code generation pattern matcher that comprises 43 rules, and a G80 code generation pattern matcher consisting of 50 rules. The G80 code generation profile inherits most NV40 rules, but replaces some NV40 rules with more specialized ones. The NV40 and the G80 profile of the code optimization pattern matcher comprise 12 rules. Both optimization profiles use the same rule set.

For both profiles, the CGiS compiler spends approximately 10% of the total compile time within the code generation and the code optimization pattern matcher. So, the influence of OORG-generated code generation and code optimization pattern matchers of the overall runtime is negligible.

5 Conclusion and Future Work

In this paper, we have presented the novel object-oriented rewrite system OORS with applications in code generation and code optimization. Using the presented OORS language, a developer is able to implement the code generation and code optimization phase of a compiler's back end in terms of pattern matchers. Retargetability is achieved by organizing rules into profiles, one for each supported hardware architecture. In contrast to other approaches, the OORS language features constructs, such as a rule and profile inheritance mechanism, that make the reuse of existing specification possible. Thus, an OORS pattern matcher specification is maintainable as well as easily retargetable in the long run.

We have additionally introduced the pattern matcher generator OORG that compiles an OORS specification into a C^{++} dynamically retargetable pattern matcher. By means of the CGiS compiler, we have demonstrated the usage of OORG-generated pattern matchers in a real world application. OORG is open source and available for download on our homepage.

The OORS language offers room for further improvements. Currently, OORS pattern matcher process only basic blocks. This restriction decreases the efficiency of certain optimizations, such as dead-code elimination. A dead-code elimination rule is currently not able to determine whether the target register is still live, if e. g., a register is written at the end of a basic block. Thus, we want to extend OORS such that matches over the whole control flow graph are possible. Apart from that, all instructions are assumed to be pushed upwards past wildcard patterns (remember Example 6, where a load-store instruction is pushed upwards). In some cases however, a developer might want to push instructions the other way around. To further improve the expressiveness of OORS, we thus want to introduce a mechanism that indicates the *direction* of a rule. Finally, we plan to improve the performance of the OORG-generated pattern matchers. The current implementation generates pattern matchers that match each rule one after another. This matching method becomes inefficient if the search patterns of two (or more) rules share the same prefix. In such a case, the OORG-generated pattern matcher would match that prefix multiple times. So, to overcome this drawback, we want to improve OORG, such that the generator produces pattern matchers that match all rules in parallel.

Acknowledgements

We thank the anonymous reviewers for their encouraging and helpful comments about this paper. Due to lack of space however, we were not able to add more content regarding further details about OORS or comparisons with other, similar approaches.

References

1. M. Alt, U. Aßmann, and H. van Someren. Cosy Compiler Phase Embedding with the CoSy Compiler Model. In *5th International Conference on Compiler Construction*, volume 786 of *LNCS*, pages 278–293. Springer-Verlag, 1994.
2. M. Breternitz Jr., H. Hum, and S. Kumar. Compilation, Architectural Support, and Evaluation of SIMD Graphics Pipeline Programs on a General-Purpose CPU. In *PACT '03: Proceedings of the 12th International Conference on Parallel Architectures and Compilation Techniques*, page 135, Washington, DC, USA, 2003. IEEE Computer Society.
3. I. Buck, T. Foley, D. Horn, J. Sugerman, K. Fatahalian, M. Houston, and P. Hanrahan. Brook for GPUs: Stream Computing on Graphics Hardware. *ACM Trans. Graph.*, 23(3):777–786, 2004.
4. Quick C-- compiler. `http://cminusminus.org/qc--.html`.
5. J. W. Davidson and C. W. Fraser. Register Allocation and Exhaustive Peephole Optimization. *Software – Practise and Experience*, 14(9):857–865, September 1984.
6. J. Dias and N. Ramsey. Converting intermediate code to assembly code using declarative machine descriptions. In *CC*, pages 217–231, 2006.
7. H. Emmelmann, F.-W. Schröer, and R. Landwehr. BEG – A Generator for Efficient Back Ends. In *PLDI*, pages 227–237, 1989.
8. S. Farfeleder, A. Krall, E. Steiner, and F. Brandner. Effective compiler generation by architecture description. In *LCTES '06: Proceedings of the 2006 ACM SIGPLAN/SIGBED conference on Language, compilers, and tool support for embedded systems*, pages 145–152, New York, NY, USA, 2006. ACM Press.
9. C. Ferdinand, H. Seidl, and R. Wilhelm. Tree Automata for Code Selection. *Acta Informatica*, 31(9):741–760, 1994.
10. C. W. Fraser and D. R. Hanson. *A Retargetable C Compiler: Design and Implementation*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1995.
11. C. W. Fraser, R. R. Henry, and T. A. Proebsting. BURG – Fast Optimal Instruction Selection and Tree Parsing. *SIGPLAN Notices*, 27(4):68–76, 1992.
12. G. Gebhard and P. Lucas. OORS: An Object-Oriented Rewrite System with Applications in Retargetable Code Generation and Optimization. Technical report, University of the Saarland, 2007. To appear.
13. S. Lerner, T. Millstein, E. Rice, and C. Chambers. Automated Soundness Proofs for Dataflow Analyses and Transformations via Local Rules. *SIGPLAN Notices*, 40(1):364–377, 2005.
14. P. Lucas, N. Fritz, and R. Wilhelm. The CGiS compiler—a tool demonstration. In A. Mycroft and A. Zeller, editors, *Proceedings of the 15th International Conference on Compiler Construction (CC)*, volume 3923 of *LNCS*, pages 105–108. Springer-Verlag, 2006.
15. P. Lucas, N. Fritz, and R. Wilhelm. The Development of the Data-Parallel GPU Programming Language CGiS. In *International Conference on Computational Science (4)*, volume 3994 of *Lecture Notes in Computer Science*, pages 200–203. Springer Verlag, 2006.
16. NVIDIA. Nv_gpu_program4. OpenGL Extension 322, 2007.
17. J. D. Owens, D. Luebke, N. Govindaraju, M. Harris, J. Krüger, A. E. Lefohn, and T. J. Purcell. A survey of general-purpose computation on graphics hardware. *Computer Graphics Forum*, 26(1):80–113, March 2007.
18. TriCore microcontroller. `http://www.infineon.com/tricore/`.

Proceedings of the International Multiconference on
Computer Science and Information Technology, pp. 1071 – 1080
ISSN 1896-7094
© 2007 PIPS

Exstensions of Deductive Concept in Logic Programing and Some Applications

Ivana Berković, Biljana Radulović, Petar Hotomski

University of Novi Sad, Technical faculty "Mihajlo Pupin" Zrenjanin, Serbia
berkovic@tf.zr.ac.yu, bradulov@tf.zr.a.yu, hotomski@ptt.yu

Abstract. The paper presents ATP – a system for automated theorem proving based on ordered linear resolution with marked literals, its putting into the base of prolog-like language and some applications. This resolution system is especially put into the base of prolog-like language, as the surrogate for the concept of negation as definite failure. This logical complete deductive base is used for building a descriptive logical programming language LOGPRO, which enables eliminating the defects of PROLOG-system (the expansion concerning Horn clauses, escaping negation treatment as definite failure), but keeping the main properties of PROLOG-language and possibilities of its expansions. Some features of the system when it is used as the base for time-table and scheduling, a technique for the implicational problem resolving for generalized data dependencies and intelligent tutoring system are described.

1. Introduction

Automated reasoning systems are computer programs which have certain "intelligent" components and can be used as: shells of expert systems, dialog systems, human language translators, educational software etc. Inferring the conclusion in such systems is often based on resolution refutation method from a certain set of rules and facts. In order to infer a conclusion, these systems, apart from negating set query, use rules for deducting facts in axiom form and working fact base. Automated reasoning systems give answers to a set query which depend upon the knowledge base and fact base [10]. In this sense, "automated reasoning is concerned with the discovery, formulation, and implementation of concepts and procedures that permit the computer to be used as a reasoning assistant", (Wos, [17]).

The developing of automated reasoning results into the developing of logic programming languages, especially PROLOG. In this paper the advantages and applications of changing one system for automated reasoning by the other are described. The determinate resolution system for automated theorem proving ATP (OL-resolution with marked literals) is especially put into the base of prolog-like language, as the surrogate for the concept of the negation as definite failure (SLDNF resolution) in PROLOG.

2. Ordered Linear Resolution as the Foundation of Automatic Theorem Proving

The most popular method for automatic theorem proving is the resolution method, which was discovered by J. A. Robinson in 1965 ([1, 8, 18]). Since 1965., many resolution forms and techniques have been developed because the pure resolution rule was unable to handle complex problems, [10].

Here is used the general automatic method for determining if a theorem (conclusion) **A** follows from a given set of premises (axioms) **F**:

$$\mathbf{F} \vdash \mathbf{A}.$$

Each formula will be transformed to the *clauses form*. The *clauses* have the form: $L1 \vee L2 \vee \ldots \vee Lm$, where Li are literals. The symbol for disjunction is: $\vee$.

The *literals* Li have the form: P(t1,t2, ... , tn) or $\neg$P(t1,t2, ... , tn), where P is predicate symbol, ti is term, $\neg$ is negation. The literal P(t1,t2, ... , tn) is called *positive* literal, the literal $\neg$P(t1,t2, ... , tn) is called *negative* literal.

Resolution method is a syntactic method of deduction. *Reduction ad absurdum* is in the basis of resolution method:

$$\mathbf{F} \vdash \mathbf{A} \quad \textbf{iff} \quad \mathbf{F} \cup \{ \neg \mathbf{A} \} \vdash \textbf{contradiction}.$$

Resolution rule will be applied on the set of clauses - axioms which was expanded by negating the desired conclusion in clause form.

Ordered Linear (OL) resolution rule with marked literals ([9], [10], [11]) increases efficiency and doesn't disturb completeness of pure resolution rule.

The generating process of OL-resolvent from central clause (**d1**) and auxiliary clause (**d2**):

1. Redesignate variables (without common variables in the clauses).
2. Determine universal unificator Θ for last literal of **d1** and **k**-literal (**k=1,2,...**) of **d2** (if it exists for some k, else it is impossible to generate OL-resolvent for specification clauses).
3. Create resolvent with marked last literal in **d1**Θ and add the rest of clause **d2**Θ without **k**-literal (**d1**Θ and **d2**Θ are clauses, which were formed by universal unificator Θ applied on **d1** and **d2**, respectively).
4. Eliminate identical non-marked literals and tautology examination (tautologies are not memorized).
5. The Shortening Operation (delete all ending marked literals)
6. The Compressing Operation (delete the last non-marked literal, which is complemented in relation to negation, with some marked literal for unificator λ).
7. Repeat steps: 5 and 6 until the empty clause is got, or the Compressing Operation is not applied on the last non-marked literal.

The rule of OL-resolution with marked literals is separated in two parts: ***in-resolution*** and ***pre-resolution***. The steps: 1 - 5 are represented in-resolution. The steps: 6 - 7 are represented pre-resolution. Mid-resolvents are the products of in-resolution and without their memorizing, the completeness of the method can be lost. This modification of Ordered Linear resolution rule is served as the base for development of the system for automatic theorem proving ATP.

3. ATP System for Automatic Theorem Proving

ATP is a system for automatic theorem proving [2], which is implemented on PC - computer by Pascal programming language. The rule of Ordered Linear Resolution with marked literals presents the system base. The system is developed at Technical Faculty "Mihajlo Pupin" in Zrenjanin. ATP is projected for scientific - researching, teaching and practical purpose. The system ATP disposes three search strategies: breadth-first, depth-first and their combination. Numerously experiments with ATP system show that depth-first strategy is the most efficient. In depth-first search, a new node is generated at the next level, from the one current, and the search is continuing deeper and deeper in this way until it is forced to backtracking.

The main characteristics of ATP system:

- This system presents a complete logical deductive base: *the clauses-set is unsatisfied (contradictory) iff the empty clause is generated by finite use of the resolution rule*. So, the proof of conclusion **A** is completed ($\mathbf{F} \vdash \mathbf{A}$) when the empty clause is generated by the resolution from clauses-set $\mathbf{F} \cup \{\neg \mathbf{A}\}$.
- Besides the theoretical completeness of the system, it has the satisfying practical efficiency limited by the space-time computer resources.
- The first-order logic is the form of representation in ATP system (each formula is transformed into the clause form). This deductive base has no restriction in Horn clause (expansions concerning Horn clauses) and it allows the logical treatment of negation (escaping negation treatment as a definite failure).

Therefore, the system of automated reasoning ATP is put into the base for development of the descriptive language for logic programming LOGPRO. This logical complete deductive base is used for building a new descriptive logical programming language ([3, 4, 5]).

4. The Concept of LOGPRO – LOGic PROgramming Language Based on ATP System

Many logic programming languages have been implemented, but PROLOG is the most popular language and it is useful for solving many problems. PROLOG as a logic-oriented language ([6, 14, 15]) contains a resolution-based theorem-prover (PROLOG-system). The theorem-prover in PROLOG appears with the depth-first search approach. It uses the special resolution rule: ***SLDNF*** (Linear resolution with Selection function for Definite clauses and Negation as Failure).

4.1. Formalization on PROLOG

The first-order predicate logic is the form of representation in PROLOG. PROLOG-program is a set of sentences. Every sentence is finished by full stop. Program in PROLOG consists of axioms (rules, facts) and a theorem to be proved (goal). The

axioms are restricted in ***Horn clauses*** form. ***Horn clauses*** [9], are clauses with at most one positive literal.

The ***rules*** have the form:

G :- T1, T2, ... , Tn .

where **G** is positive literal and **Tj (j=1,2,...,n)** are literals (positive or negative). The symbol for conjunction is: **,** . The element **G** is presented head of the rule. The elements **Tj (j=1,2,...,n)** are presented body of the rule. The separator **:-** corresponds to implication (⇐). The symbol for negation is: **not**.

The ***facts*** have the form: **G.**

where **G** is positive literal.

The ***goals*** (questions) have the form: **?- T1, T2, ... , Tn .**

where **Ti (i=1,2,...,n)** are literals.

Practically, programming in PROLOG is restrictive in a subset of first-order logic. Horn clauses are represented the first defect of PROLOG. The concept of negation as definite failure is represented the second defect of PROLOG.

4.2. Formalization on LOGPRO

An other approach to logic programming is implementation of a different deductive concept. The determinate system ATP for automated theorem proving is especially put into the base of prolog-like language LOGPRO, as the surrogate for the concept of negation as definite failure. This logical complete deductive base is used for building a new descriptive logic programming language.

The first-order logic is the form of representation in ATP system, too. But, this system has not restriction in Horn clauses. The program on logic language LOGPRO based on the ATP system is a set of sentences (clauses). There are three kinds of sentences: rules, facts and goals. Every sentence is finished by full stop.

The ***rules*** have the form:

G1, G2, ... , Gm :- T1, T2, ... , Tn .

where **Gi (i=1,2,...,m)** and **Tj (j=1,2,...,n)** are literals (positive or negative). The symbol for conjunction is: **,** . The elements **Gi (i=1,2,...,m)** present head of the rule. The elements **Tj (j=1,2,...,n)** present body of the rule. The separator **:-** corresponds to implication (⇐). The symbol for negation is: ~ .

The ***facts*** have the form: **G.**

where **G** is literal (positive or negative).

The ***goals*** (questions) have the form: **?- T1, T2, ... , Tn .**

where **Ti (i=1,2,...,n)** are literals (positive or negative).

The rules and facts (axioms) are presented by auxiliary clauses. The goal (central clause) is negating the theorem to be proved. Symbol **?-** in goal is the substitution for negation. The execution procedure is ATP system based on OL-resolution with marked literals. This formulation enables eliminating the defects of PROLOG-system.

The logic programming languages PROLOG and LOGPRO are compared. PROLOG rules and facts do not allow the explicit statement of negative information. But, the declarative syntax of the logic programming language LOGPRO allows the expression of negative information in rules and facts. Also, it is possible to construct the rule with more than one element in the rule‘s head.

Example 1.

The problem of trying to formulate sentence:

"Alice likes whatever Queen dislikes, and dislikes whatever Queen likes." into PROLOG form, [16]. The representations

```
likes(alice,X1) :- not likes(queen,X1).
not likes(alice,X1) :- likes(queen,X1).
```

are illegal in PROLOG because the second rule has a negation in head (it isn't Horn clause). It is possible to solve the problem by trick - using a modified predicate likes, and expressing the statement as:

```
likes(alice,X1,true) :- likes(queen,X1,false).
likes(alice,X1,false) :- likes(queen,X1,true).
```

The expansion concerning Horn clauses on the logic programming language based on ATP system has the possibilities to express the statement as:

```
likes(alice,X1) :-  ~ likes(queen,X1).
~ likes(alice,X1) :-  likes(queen,X1).
```

PROLOG-system has the negation defect, [7]. This defect is corrected in ATP system. It can be illustrated by the following example.

Example 2.

Program in PROLOG:

```
vegetarian(tom).
vegetarian(ivan).
smoker(tom).
likes(ana,X1) :- not (smoker(X1)), vegetarian(X1).
```

PROLOG-system gives unconnected answers on the following questions:

```
?- likes(ana,X1).
no
?- likes(ana,ivan).
yes
```

If the last clause is now:

```
likes(ana,X1) :-  vegetarian(X1), not (smoker(X1)).
```

PROLOG-system gives wrong answers on the following questions:

```
?- likes(ana,X1).
X1=ivan
?- likes(ana,ivan).
yes
```

These answers are incorrect because we have not data about Ivan and smoking. We don't know if Ivan is a smoker or not. The correct answer will be: "I don't know".

The program in LOGPRO :

```
vegetarian(tom).
vegetarian(ivan).
smoker(tom).
likes(ana,X1) :- ~ smoker(X1), vegetarian(X1).
```

ATP-system gives answers on the following questions:

```
?- likes(ana,X1).
Success=0
```

```
The proof isn't completed
?- likes(ana,ivan).
Success=0
The proof isn't completed
```

When the last clause is:

```
likes(ana,X1) :- vegetarian(X1), ~smoker(X1).
```

ATP system also gives the correct answers: **"Success=0, the proof isn't completed"**.

In fact, the system generates resolvents, but can not complete the proof with depth-first strategy. The treatment of negation as definite failure in this system is escaped. The concept of LOGPRO allows eliminating of endless branches, recursion using and works with structures and lists, as well as PROLOG. It is presented in some concrete examples, [3].

5. Applications of ATP System and LOGPRO

5.1. Time-Table And Scheduling System DEDUC

This system ATP is incorporated in the system for automatic creating of the combinatorial disposition DEDUC, where it has presented the satisfying practical efficiency.

DEDUC is a software package that integrates scientific results of Constraint Logic Programming and practical needs in generating combinatorial schedules. The work on the system started in 1991. After ten years the powerful system was developed.

System is based on synchronized work of two processes: data storing and optimization of gaps in the timetable. Theorem prover controls both processes to secure that initial requirements are not violated

System facilities and performances:

- Automatic and interactive generating of schedules for the initial data and conditions;
- Setting up and maintenance of the initial data and requirements by using the user friendly interface;
- Setting up various initial requirements like splitting classes into groups, connecting groups and classes, enabling multiple lessons; setting restrictions in using laboratories and rooms; setting teachers' requirements and the other pedagogic obligations and desirable demands;
- Generating schedules respecting school shifts;
- Screen browsing and printing the general, per-class and per-teacher schedules;
- Generating and archiving different versions of the schedules generated for the same initial data and conditions;
- Maintenance of the archived versions, data and conditions.

The practical efficiency and system limits can be observed for complicated examples as well as for the simple ones. Essentially, getting the acceptable version of the schedule depends on structure of the initial requirements and their connections with data, although the number of requirements has no influence.

Practical experiences in the real world assure that DEDUC system generates timetables with quality level higher than quality level of hand-made timetables. The time needed for generating a timetable varies from few minutes to several hours, and depends on amount of data, structure of conditions as well as on computer performances. More informations at `http://deduce.tripod.com`

5.2. A Technique For The Implicational Problem Resolving For Generalized Data Dependencies

A mechanism for generalized representation of various data dependency types, such as functional (*fd*), multivalued (*mvd*), join (*jd*), implied (*imd*) and inclusion (*id*) dependencies, has been developed in the relational data model theory. The initial supposition was that all the data dependencies (i.e. "rules" which hold amond data) can be represented, in unified manner, by one, or more symbolic data templates, satisfying certain criteria, according to defined interpretation rules for such symbolic templates. On the basis of that supposition, the term of generalized data dependency has been introduced, so as to represent the other dependency types in the same way. One of the important questions, arising when new data dependency type is introduced, is how it can be stated if a data dependency is a logical consequence of a given set od data dependencies. This problem in relational databases is known as the *implicational problem*.

At the beginning, the terms of: tableau, as a symbolic data template, generalized dependency (*gd*) and its interpretation are defined without explanations, because they are considered as already known. In ([12], [13]) is presented a possible approach to resolving the implicational problem for *gds*, it is estrablised at the same time a way of testing the implicational problem for all the specific data dependency types which can be formalized by means of *gds*. The proposed approach considers a usage of the ATP System.

To resolve the implicational problem for a given sets of *gds* Γ and arbitrary *gd* γ means to establish if $\Gamma \models \gamma$ holds. It is not pratcitally possible to test he implicational problem $\Gamma \models \gamma$ by exact applying of definition of generalized dependencies by systematic generating of all the relations form *SAT (R)* and checking the implication $r \models \Gamma \Rightarrow r \models \gamma$, because *SAT(R)* is, in most cases, the set of high cardinality and it can be even infinite. Therefore, the other methods have to be applied so as to resolve the problem. According to the nature of *gds*, it is concluded that for the automations of the test $\Gamma \models \gamma$, the resolution procedure can be applied. Therefore, the set of Γ and the dependency γ will be represented by appropriate predicate formulas. According to the resolution theorem and theorem of generalized dependencies, the test of the condition $\Gamma \models \gamma$, where $\Gamma = \{\gamma_1, \ldots, \gamma_n\}$, is performed by disproving procedure, on the basis of the set of initial assumptions $\mathbf{F}(\Gamma)$ and the negation of the conclusion $\neg F$. In that case, the theorem that should be proved by ATP System is of the form: $\mathbf{F}(\Gamma) \rightarrow F$.

To prove the theorem, the clauses should be built from the set of assumptions $\mathbf{F}(\Gamma)$ and negation $\neg F$. They represent the input for ATP. Beside that, two additional input parameters are: (i) maximal searching deep and (ii) maximal clause length. With respect to the resolution theorem, there are three possible outcomes from ATP. (a) "*positive*"; an empty clause has been reached, which means that the assertion *F* holds. According to theorem of generalized dependencies, $\Gamma \models \gamma$ holds, too; (b) "*negative*":

the empty clause has not been reached and there are no more possibilities for new clause generating. It means that the conclusion F cannot be derived from $\mathbf{F}(\Gamma)$. According to theorem of generalized dependencies, we conclude $\Gamma \models \gamma$ does not hold; (c) "*uncertain"*: the empty clause has not been obtained, whereas maximal searching deep and maximal clause length have been reached, or memory resources have been exhausted.

5.3. Intelligent Tutoring system based on ATP System

The concept of LOGPRO can be an excellent base for an intelligent tutoring system iTUTOR ([7], [5]). Such deductive logic programming language can perform tasks that standard PROLOG system could not [10].

It is now possible to define predicate:

```
know(studentname, concept, degree).
```

where studentname is name of a student, concept is name of a concept that student should know and degree indicates grade of concept cognition.

Negation in the head of the rule can be implemented as:

```
~know(studentname, concept, degree):-
     know(studentname, concept1, degree1),
     know(studentname, concept2, degree2),
     ...
     know(studentname, conceptn, degreen),
     degree1<c1,
     degree2<c2,
     ...
     degreen<cn.
```

which means that student does not know concept in certain degree if he does not know minor concepts concept1, concept2, ..., concept in degree greater than or equal with c1, c2, ..., cn where c1, c2, ..., cn are predefined constants. Furthermore, one can calculate degree of concept cognition by adding a line at the end of previous rule:

```
degree is (degree1+ degree2+...+ degreen)/n.
```

Rule:

```
~nextlesson(studentname, lessonname):-
     know(studentname, concept1, degree1),
     know(studentname, concept2, degree2),
     ...
     know(studentname, conceptn, degreen),
     degree1>c1,
     degree2>c2,
     ...
     degreen>cn.
```

indicates lessons that particular student should not learn because he knows all belonging concepts in degree greater than predefined constant.

Similar rule :

```
~nextlesson(studentname, lessonname):-
     ~know(studentname, concept1, degree1),
```

```
know(studentname, concept2, degree2),
...
know(studentname, conceptn, degreen),
degree1>c1,
degree2>c2,
...
degreen>cn.
```

where some predicates in the body of the rule are negated, indicates lessons that student should not learn because he does not know all belonging concepts in degree greater than predefined constant. Application of such logic programming language LOGPRO can overcome some problems during the process of student modelling in an intelligent tutoring system.

6. Conclusion

Completeness and universality of the resolution method, as the base of ATP system, enables it to be applied as the deductive base of prolog-like language. The relationship between programming language based on ATP system and programming language PROLOG are emphasized. The logic programming language based on ATP system enables eliminating the defects of PROLOG-system (the expansion concerning Horn clauses, escaping negation treatment as definite failure, eliminating of endless branches), keeping the main properties of prolog-language. In this paper are also described some applications of ATP system and LOGPRO such as: time-table and scheduling (DEDUC), a technique for the implicational problem resolving for generalized data dependencies and intelligent tutoring system based on atp system (iTUTOR). Also, ATP system is used for the development BASELOG system, which will be described in separate papers.

References

1. Barr A., Cohen P. R., Feigenbaum E. A., *The Handbook of Artificial Intelligence,* Vol.I, II, III, Heuris Tech Press, W. Kaufmann, Inc., California 1982.
2. Berković I., *Variable Searching Strategies in the Educationally Oriented System for Automatic Theorem Proving,* M. Sc. Thesis, Technical Faculty "Mihajlo Pupin", Zrenjanin, 1994. (in Serbian).
3. Berković I., *The Deductive Bases for the Development of the Descriptive Languages for the Logical Programming,* Ph. D. Thesis, Technical Faculty "Mihajlo Pupin", Zrenjanin, 1997. (in Serbian).
4. Berković I., Hotomski P., *The concept of logic programming language based on resolution theorem prover,* Proceedings of VIII Conference on Logic and Computer Science LIRA, Sept 1-4 1997, pp. 111-120.
5. Berković I., Hotomski P., Brtka V., *The Concept of Logic Programming Language Based on the Resolution Theorem Prover and its Appliance to Intelligent Tutoring Systems;* IEEE 7th International Conference on Intelligent Engineering Systems; March 4 - 6.; Assiut - Luxor; Egypt; IEEE Proceedings; (2003); 169 - 172; ISSN: 977.246.048.3/1562
6. Bratko I., *PROLOG Programming for Artificial Intelligence,* Addison-Wesley Publ. Comp. 1986.

7. Brtka V., *Tutoring educational software,* M. Sc. Thesis, Technical Faculty "Mihajlo Pupin", Zrenjanin, 2001, (in Serbian).
8. Gevarter W. B., *Intelligent Machines,* Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1985.
9. Hotomski P., Pevac I., *Mathematical and Programming Problems of Artificial Intelligence in the Field of Automatic Theorem Proving,* Naučna knjiga, Belgrade, 1991. (in Serbian).
10. Hotomski P., *Systems of Artificial Intelligence,* University of Novi Sad, Technical Faculty "Mihajlo Pupin" Zrenjanin, 2004. (in Serbian) .
11. Kowalski P., Kuchner D., *Linear Resolution with Selection Function,* Artificial Intelligence, Vol. 2, 1971., pg. 227-260.
12. Luković I., Hotomski P., Radulović B., Berković I., *A Proof of Generalized Data Dependency Satisfaction by The Automatic Reasoning Method,* II Symposium on Comp, Sc. and Informatics YU INFO, Brezovica, 02-05.-4.1996. (in Serbian).
13. Luković I., Hotomski P., Radulović B., Berković I., *A Technique for the Implicational Problem Resolving for Generalized Data Dependencies,* VIII Conf. on Logic and Computer Science LIRA '97, Novi Sad, 01-04.09.1997, pp. 111-119.
14. Malpas J., *PROLOG: A Relational Language and Its Applications,* Prentice-Hall International Inc., 1987.
15. Pereira F. C. N., Shieber S. M., *PROLOG and Natural-Language Analysis,* CLSI, Leland Stanford Junior University, 1987.
16. Subrahmanyam P. A., *The "Software Engineering" of Expert Systems: Is Prolog Appropriate?* IEEE Transactions on Software Engineering, Vol. SE-11, No. 11, november 1985., pp. 1391-1400.
17. Wos L., *Automated Reasoning,* The American Mathematical Monthly, Vol. 92, No. 2, february 1985., pp. 85-93
18. Wos L., Overbeek R., Lusk E., Boyle J., *Automated Reasoning: Introduction and Applications,* Prentice-Hall, 1992.

Proceedings of the International Multiconference on Computer Science and Information Technology pp. 1081–1091
ISSN 1896-7094
© 2007 PIPS

Adaptive Language Approach to Software Systems Evolution*

Ján Kollár, Jaroslav Porubän, Peter Václavík,
Jana Bandáková, and Michal Forgáč

Technical University of Košice, Košice, Slovakia

{Jan.Kollar,Jaroslav.Poruban,Peter.Vaclavik}@tuke.sk,
{Jana.Bandakova,Michal.Forgac}@tuke.sk
WWW home page: http://kpi.fei.tuke.sk

Abstract. From the viewpoint of adaptability, we classify software systems as being nonreflexive, introspective and adaptive. Introducing a simple example of LL(1) languages for expressions, we present its nonreflexive and adaptive implementation using Haskell functional language. Multiple metalevel concept is an essential demand for a systematic language approach, to build up adaptable software systems dynamically, i.e. to evolve them. A feedback reflection loop from data to code through metalevel data is the basic implementation requirement and the proposition for semi-automatic evolution of software systems. In this sense, practical experiment introduced in this paper is related to the base level of language, but it illustrates the ability for extensions primarily in horizontal but also in vertical direction of an adaptive system.

Keywords Adaptive systems evolution, programming languages, program transformation, aspect oriented languages, domain specific languages, reflection.

1 Introduction

There is an increasing demand for systems that can be easily configured for a specific environment or they even adjust themselves dynamically to a changing environment at runtime. Adaptive behaviour is the proposition for the runtime adjustment, or evolution, provided that it is performed automatically.

Adaptiveness [7] is mostly related to the area of software engineering – object oriented programming, aspect-oriented programming, intentional programming, template programming, etc., where it is exploited for changing the semantics of programs.

On the other hand, the properties of systems are expressed using different kinds of languages, such as programming languages, specification languages, or modeling languages, in that solutions of problems are constructively formulated,

* This work was supported by VEGA project No. 1/4073/07 "Aspect-oriented Evolution of Complex Software Systems"

no matter of a language abstraction level. At the bottom level of a system, the machine code written in a machine language is executed. At the top level, human thoughts arise and are formulated using a natural language.

Our current research concentrates on how a language (not a program) can vary its semantics, reflecting not just compile time but also runtime events. According to our opinion, static and dynamic adaptation of the language to new software aspects and runtime events should exclude current expensive methods of compiler construction and extensions. The language should be minimal and strongly associated with the properties of a software system.

We have recognized the concepts of *metaprogramming* and *reflection* are fundamental. *Metaprogramming* is about writing programs that represent and manipulate other programs or themselves, i.e. metaprograms are programs about programs [1]. *Reflection* is an entity's integral ability to represent, operate on, and otherwise deal with itself in the same way that it represents, operates on, and deal with its primary subject matter [3]. The main idea of applying reflection as a general principle for flexible systems in software engineering is to split a system into two parts: metalevel and a base level. A metalevel provides information about selected system and makes the software *self-aware*. A base level includes the application logic.

In this paper, we present the principle of an adaptive context-free language focusing on the most complex phase – the syntax directed translation from lexical symbols to postfix code. For the purpose of simplicity and clarity, we decided to use Haskell without monads. The immediate impulse for this work was our discussion about the restricted degree of adaptiveness using Java's metaclasses and Java runtime environment.

In Section 2 we introduce our classification of software systems from the viewpoint of the degree of reflexive behaviour, and we analyze three selected cases. In Section 3 we present LL(1) language for expressions and its nonreflexive implementation. The crucial points of its adaptive version are presented in Section 4. The conception of multilevel adaptive language system is discussed briefly in Section 5.

2 Systems Behaviour Classification

In this section we classify software systems from the viewpoint of their degree of adaptive behaviour.

2.1 Nonreflexive Execution

Machine code – the constant set ${}^0\mathbf{C}$ of instructions at base level 0 does not vary during execution, and then the execution changes data ${}^0\mathbf{D}^{(k)}$ (a set of data records on the stack or in the heap) to a new data set ${}^0\mathbf{D}^{(k+1)}$. An execution step is the transformation of configuration (1).

$$ {}^0\mathbf{C} \quad \xrightarrow{\times} \quad {}^0\mathbf{D}^{(k)} \quad \Longrightarrow \quad {}^0\mathbf{C} \quad \xrightarrow{\times} \quad {}^0\mathbf{D}^{(k+1)} \tag{1} $$

In (1), the relation $(\xrightarrow{\times})$ denotes that multiple instructions from ${}^0\mathbf{C}$ can access multiple data records ${}^0\mathbf{D}$.

The execution is nonreflexive, if there is no feedback loop from data to code, and no possibility is given to code to observe or even to change itself.

2.2 Introspective Execution

In an introspective execution, code ${}^0\mathbf{C}$ constructs and changes data ${}^0\mathbf{D}^{(k)}$, as in a nonreflexive execution. In addition to this, each subset of the set ${}^0\mathbf{D}^{(k)}$ refers to (which we designate by $\nwarrow^-$) exactly one element of static data ${}^1\mathbf{S}$, and this data refers ($\nwarrow^+$) to a subset of code ${}^0\mathbf{C}$ accordingly (2). Static data set ${}^1\mathbf{S}$ at level 1 is metalevel static data to the level 0.

$$\begin{array}{ccccc} {}^1\mathbf{C} & \xrightarrow{\times} & {}^1\mathbf{S} & & \\ & {}^+\swarrow & \nwarrow^- & \Longrightarrow & \\ {}^0\mathbf{C} & \xrightarrow{\times} & {}^0\mathbf{D}^{(k)} & & \end{array} \qquad \begin{array}{ccccc} {}^1\mathbf{C} & \xrightarrow{\times} & {}^1\mathbf{S} & & \\ & {}^+\swarrow & \nwarrow^- & & \\ {}^0\mathbf{C} & \xrightarrow{\times} & {}^0\mathbf{D}^{(k+1)} & & \end{array} \tag{2}$$

Since metalevel data (set of records) ${}^1\mathbf{S}$ is static, metacode ${}^1\mathbf{C}$ may produce it just once, and then the execution of ${}^1\mathbf{C}$ is finished. Clearly, such metacode cannot be runtime process, and execution of ${}^0\mathbf{C}$ is nonadaptive. However, it is introspective, because of existence of feedback loop from code ${}^0\mathbf{C}$ to code ${}^0\mathbf{C}$ via data set ${}^0\mathbf{D}$ and some metadata element from ${}^1\mathbf{S}$. For example, introspective (but not adaptive) behaviour can be obtained using Java's metaclasses that exploit static metadata ${}^1\mathbf{S}$.

2.3 Adaptive Execution

Adaptive execution step is defined in (3). In this case, metalevel data ${}^1\mathbf{D}^{(m)}$ can change in runtime to data ${}^1\mathbf{D}^{(m+1)}$, by execution of metalevel code ${}^1\mathbf{C}$. This code itself is nonreflexive, since there is no feedback loop via metadata at metametalevel 2. On the other hand, a new ${}^1\mathbf{D}^{(m+1)}$ may yield a new ${}^0\mathbf{C}^{(k+1)}$, continuing its execution at level 0.

$$\begin{array}{ccccc} {}^1\mathbf{C} & \xrightarrow{\times} & {}^1\mathbf{D}^{(m)} & & \\ & {}^+\swarrow & \nwarrow^- & \Longrightarrow & \\ {}^0\mathbf{C}^{(k)} & \xrightarrow{\times} & {}^0\mathbf{D}^{(k)} & & \end{array} \qquad \begin{array}{ccccc} {}^1\mathbf{C} & \xrightarrow{\times} & {}^1\mathbf{D}^{(m+1)} & & \\ & {}^+\swarrow & \nwarrow^- & & \\ {}^0\mathbf{C}^{(k+1)} & \xrightarrow{\times} & {}^0\mathbf{D}^{(k+1)} & & \end{array} \tag{3}$$

In this way, code at level 0 may be not just introspective, but also adaptive, and this fact is essential for an adaptive execution. For example, adaptive behaviour can be obtained using Smalltalk's metaobjects that exploit dynamic metadata ${}^1\mathbf{D}^{(m)}$.

3 Nonreflexive Language Implementation

In this section, we introduce the implementation of simple LL(1) language for expressions in a nonreflexive manner.

Grammar of the language is written in extended BNF (EBNF), see (4).

$$\begin{array}{l} E \to A\ \{\ (\text{"}+\text{"} \mid \text{"}-\text{"})\ A\ \} \\ A \to B\ [\ (\text{"}*\text{"} \mid \text{"}/\text{"})\ A\ ] \\ B \to \mathit{const} \mid \text{"(" } E \text{ ")"} \end{array} \tag{4}$$

where $[\varphi] = (\varphi \mid \varepsilon)$, ε is empty symbol, φ is a syntactic expression, and $\{\varphi\} = \varepsilon \mid \varphi \mid \varphi\ \varphi \mid \ \ldots$ is the transitive closure.

Lexical analyzer and translator to postfix code are common passes to both compiler and interpreter of the language.

Lexical analyzer `lexical` translates lexical units from string to symbol form, for example " + " to **AddL**, " − " to **SubL**, integer constants *const* to **ValL** v, etc. The correspondence of string and symbol form for all lexical units is visible by comparison of (4) and (5).

The translator `translate` translates symbol form of lexical units to postfix code using syntax directed translation, following the rules (5).

$$\begin{array}{ll} \mathcal{E}\ [\![\ A\ \{\ \mathbf{AddL}\ A_k\ \}\ ]\!] & = \mathcal{A}\ [\![\ A\ ]\!]\ \{\ \mathcal{A}\ [\![\ A_k\ ]\!]\ \mathbf{Add}\ \} \\ \mathcal{E}\ [\![\ A\ \{\ \mathbf{SubL}\ A_k\ \}\ ]\!] & = \mathcal{A}\ [\![\ A\ ]\!]\ \{\ \mathcal{A}\ [\![\ A_k\ ]\!]\ \mathbf{Sub}\ \} \\ \mathcal{A}\ [\![\ B\ [\ \mathbf{MulL}\ A\ ]\ ]\!] & = \mathcal{B}\ [\![\ B\ ]\!]\ [\ \mathcal{A}\ [\![\ A\ ]\!]\ \mathbf{Mul}\ ] \\ \mathcal{A}\ [\![\ B\ [\ \mathbf{DivL}\ A\ ]\ ]\!] & = \mathcal{B}\ [\![\ B\ ]\!]\ [\ \mathcal{A}\ [\![\ A\ ]\!]\ \mathbf{Div}\ ] \\ \mathcal{B}\ [\![\ \mathbf{ValL}\ v\ ]\!] & = \mathbf{Push}\ v \\ \mathcal{B}\ [\![\ \mathbf{LparL}\ E\ \mathbf{RparL}\ ]\!] & = \mathcal{E}\ [\![\ E\ ]\!] \end{array} \tag{5}$$

The translation starts by $\mathcal{E}\ [\![\ E\ ]\!]$, since E is starting symbol.

Compiler consists of lexical analyser, translator, machine code generator and loader. The execution is performed by emulated target machine.

Code generation and loading are composed into single pass. The transformation of postfix code to machine code by $\mathcal{C}$ is defined in (6).

$$\begin{array}{lll} \mathcal{C}\ [\![\ \mathbf{Add}\ ]\!] = 1 & \mathcal{C}\ [\![\ \mathbf{Sub}\ ]\!] = 2 & \mathcal{C}\ [\![\ \mathbf{Mul}\ ]\!] = 3 \\ \mathcal{C}\ [\![\ \mathbf{Div}\ ]\!] = 4 & \mathcal{C}\ [\![\ \mathbf{Push}\ x\ ]\!] = 5\ x & \end{array} \tag{6}$$

where x is integer value, pushed on the stack by instruction **Push** (of code 5). Machine code is generated and loaded to memory (represented by a list) by function `genload`. In this way, `genload` performs code generation as well as loading actions, that are invoked by the application (7)

$$\texttt{genload}\ \mathit{pcode} \tag{7}$$

where *pcode* is postfix code produced by translator `translate`. The value of the application (7) is machine code.

Exit instruction is added (i.e. woven) to machine code in load time to enable to stop the execution. In this way, the semantics of original LL(1) language is statically (although not significantly) changed.

Machine architecture comprises program counter *pc*, the number of stacked values *sp* (used instead of stack pointer), the accumulator *a*, the memory *mem*, and the stack *stack*. An execution step is defined by the transformation of machine configuration (8).

$$(pc,\ sp,\ a,\ mem,\ stack) \Rightarrow (pc',\ sp',\ a',\ mem,\ stack') \tag{8}$$

Machine code is executed (emulated) by `exec` $(\texttt{0}, \texttt{0}, \texttt{0}, \texttt{mcode}, [])$, where `mcode` is target code loaded in memory *mem*.

Interpreter consists of lexical analyser, translator, and function `eval`, which evaluates postfix code *pcode* directly, according to (9).

$$\texttt{eval}\ pcode = v \tag{9}$$

in which *pcode* is postfix code produced by the translator and v is the result of interpretation.

Although compiler and machine emulator have great potential for adaptive implementation, we will focus on adaptive language using interpreter, for the limited scope of this paper. A nonreflexive version of interpreter `interpreter` is defined by composition, as follows:

$$\texttt{interpreter} = \texttt{eval} \,.\, \texttt{translate} \,.\, \texttt{lexical}$$

4 Adaptive Language Implementation

First, let us introduce the task of adaptation informally.

> *Depending on the result of interpretation, the language defined by* (4) *and* (5) *should be changed, and the next interpretation follows different semantics, i.e. potentially different result of the same source expression.*

We have selected this task taking into account that it affects the most complex phase – the translation of context-free language to postfix language.

More specifically, let *res* be a result of interpretation. If $res < 10$, then we require operations $(+)$ and $(-)$ be right-associative. If $res \in \langle 10, 20)$ then we require $(*)$ and $(/)$ be left-associative. And finally, if $res \geq 20$, then mutual interchange of priority of $\{+, -\}$ and $\{*, /\}$ is required.

All mentioned requirements are summarized in Table 1, which contains also zero variant, corresponding to original priority and associativity of operations defined by (4), as follows: operations $(+)$ and $(-)$ are left-associative, and they are of lower priority than operations $(*)$ and $(/)$, that are right-associative.

According to our specification, if a non-zero variant is selected, the language will never be adapted to its zero variant.

```
translate :: Int -> [LexUnit] -> [Instruction]
translate k ls = (snd . pE) (ls,[])
 where
  rules = [("E", gS v1), ("A", gS v2), ("B", pB) ]
          where (v1,v2) = variants !! k

  ap nt = snd (head [(n,f) | (n,f) <- rules , n==nt])

  pE = ap "E"

  gS :: (String,Char,LexUnit,Instruction,LexUnit,Instruction,String)->
        (LexUnits,Code) -> (LexUnits,Code)
  gS (s1,t,lo1,o1,lo2,o2,s2) ([],cs)  = ([],cs)
  gS (s1,t,lo1,o1,lo2,o2,s2) (ls,cs)
    | t == 'c' = cls ((ap s1 ) (ls,cs)) []
    | t == 'a' = alt ((ap s1 ) (ls,cs)) []
     where cls ([], cs) no = ([], cs++no)
           cls ((l:ls), cs) no
               | l == lo1 = cls ((ap s2 ) (ls,cs++no) ) [o1]
               | l == lo2 = cls ((ap s2 ) (ls,cs++no) ) [o2]
               | otherwise = ((l:ls),cs++no)
           alt ([], cs) os  = ([], cs++os)
           alt ((l:ls), cs) os
                | l == lo1 = alt ((ap s2 ) (ls,cs)) (os++[o1])
                | l == lo2 = alt ((ap s2 ) (ls,cs)) (os++[o2])
                | otherwise = ((l:ls),cs++os)

  pB :: (LexUnits,Code) -> (LexUnits,Code)
  pB ([],cs)                  = ([],cs)
  pB (((ValL x):ls),cs)       = (ls,cs++[Push x])
  pB ((l:ls),cs) | l == LparL = skipR ((ap "E" ) (ls,cs))
                                  where
                                    skipR ((l:ls'),cs') = (ls',cs')
```

Fig. 1. Adaptive translator to postfix code

Table 1. The task of adaptation

Variant	On condition	Requirement
0	none	none
1	$res < 10$	$\{+,-\} \rightarrow R$
2	$res \in \langle 10, 20)$	$\{*,/\} \rightarrow L$
3	$res \geq 20$	$\{+,-\} \leftrightarrow \{*,/\}$

First, we generalize the translator, using the following methodology: Comparing the translation schemes of rules for E and A, we define more general implementation for two rules in the form of function `gS`, introduced in Fig. 1.

The adaptiveness is reached by parameter `(s1,t,lo1,o1,lo2,o2,s2)` of `gS`, by function `rules`, and by function `ap`.

Function `rules` represents translation rules in a graph form, i.e. as a data and it is sensitive to variant. The translation rules are then applied indirectly – using function `ap`. Variants that affect the translation rules, performing translator adaptation, are shown in Fig. 2.

Finally, we define a simple metacode, including also adaptive interpreter `interpreter`, see Fig. 3. Using auxiliary function `allVariants`, we can verify correctness of adaptive interpretation for all variants, for example:

```
> allVariants "10-3-1"
= [6,8,6,6]
> allVariants "20-3-1"
= [16,18,16,16]
> allVariants "30-3-1"
= [26,28,26,26]
```

Function `adaptInt` performs the change of the language semantics according to the selected variant `k`, dependent on previous result of evaluation.

Function `adaptInt` takes a source expression and produces the triple: the first item is the value of the source expression, the second item is selected variant number depending on this value. The third item—new value is obtained by interpretation of the same source expression translated to potentially different postfix code by adapted translator and subsequently evaluated.

For example, for input expression $(10 - 3 - 1)$ (of value 6), variant 1 is selected and expression will be re-evaluated as being in the form $(10 - (3 - 1))$, producing value 8.

This is so, because variant 1 selects pair `(v3,v2)` from variants, see Fig. 2, and adapts scheme $\mathcal{E}$ in (5) to the scheme

$$\begin{aligned} \mathcal{E}\ [\![\ A\ [\ \mathbf{AddL}\ E\ ]\ ]\!] &= \mathcal{A}\ [\![\ A\ ]\!]\ [\ \mathcal{E}\ [\![\ E\ ]\!]\ \ \mathbf{Add}\ ] \\ \mathcal{E}\ [\![\ A\ [\ \mathbf{SubL}\ E\ ]\ ]\!] &= \mathcal{A}\ [\![\ A\ ]\!]\ [\ \mathcal{E}\ [\![\ E\ ]\!]\ \ \mathbf{Sub}\ ] \end{aligned}$$

For input expression $(14 - 3 - 1)$ (of value 10), variant 2 is selected and the expression will be re-evaluated as $((14 - 3) - 1)$, producing the same value 10.

```
variants = [(v1,v2), (v3,v2), (v1,v4), (v5,v6)]
             where
               v1 = ("A",'c',AddL,Add,SubL,Sub,"A")
               v2 = ("B",'a',MulL,Mul,DivL,Div,"A")
               v3 = ("A",'a',AddL,Add,SubL,Sub,"E")
               v4 = ("B",'c',MulL,Mul,DivL,Div,"B")
               v5 = ("A",'a',MulL,Mul,DivL,Div,"E")
               v6 = ("B",'c',AddL,Add,SubL,Sub,"B")
```

Fig. 2. Definition of variants

```
selVariant v | v <  10           = 1
             | v >= 10 && v < 20 = 2
             | v >= 20           = 3

interpreter k = eval . translate k . lexical

adaptInt s = (res, variant, interpreter variant s)
             where
               res      = interpreter 0 s
               variant  = (selVariant res)

allVariants s = [interpreter 0 s,
                 interpreter 1 s,
                 interpreter 2 s,
                 interpreter 3 s]
```

Fig. 3. Metalevel code

Although variant 2 adapts scheme $\mathcal{A}$ to the scheme

$$\begin{array}{l} \mathcal{A}\ [\![\ B\ \{\ \mathbf{MulL}\ B_k\ \}\ ]\!] = \mathcal{B}\ [\![\ B\ ]\!]\ \{\ \mathcal{B}\ [\![\ B_k\ ]\!]\ \mathbf{Mul}\ \} \\ \mathcal{A}\ [\![\ B\ \{\ \mathbf{DivL}\ B_k\ \}\ ]\!] = \mathcal{B}\ [\![\ B\ ]\!]\ \{\ \mathcal{B}\ [\![\ B_k\ ]\!]\ \mathbf{Div}\ \} \end{array}$$

this semantical change does not affect the expression, in which just subtraction is applied.

The evaluation in both cases described above is as follows:

```
> adaptInt "10-3-1"
= (6,1,8)
> adaptInt "14-3-1"
= (10,2,10)
```

But notice, `adaptInt "64/8/2"` would evaluate first time to 16, but for the second time to 4.

Our metacode can be extended for solving more powerful tasks, and different metadata variants may result to semantically different adaptation effects. It can be also noticed, that adaptive language implementation is stronger than an introspective one. For example, an introspective task is such as counting the number of addition operations used in an expression.

5 Discussion and Related Works

The work presented in this paper comes out from our past research in the application of our process functional paradigm [5,6] to the aspect-oriented languages [4,8,10], until we have recognized that statically defined semantics of the language of pointcut designators insufficiently supports the adaptiveness. The detailed analysis of this fact is over the scope of this paper, but having performed this we have decided to return back to Lieberherr's [7] concept of adaptive systems, and even to their essential principles, such as metaprogramming and reflection.

In this paper, we have used a purely functional approach using Haskell [9], but without monads [11], to simplify the notation as much as possible. Close relation of our adaptive language implementation and adaptive execution (3) is still visible: we abstract multiple translators by single `translate k` function, but this is equivalent to the association of a new version of translator via a new metadata.

A two-dimensional separation of concerns for compiler construction [12] tends us to think about two dimensional domain specific language evolved in multiple metalevels.

This approach has sense, if each higher metalevel generalizes lower metalevel very consisely, i.e. $|\ ^{(\ell+1)}\mathbf{D}| \ll |\ ^{\ell}\mathbf{D}|$ holds, and the computational time does not increase significantly, i.e. $^{(\ell+1)}\tau \ll\ ^{\ell}\tau$ holds.

In contrast to language evolution by inferring a language from samples of programs [2], our approach is based on inferring a metalanguage from samples of metaprograms.

6 Conclusion

Presented adaptive LL(1) language can be extended in many directions, exploiting feedback loops from any phase to any phase of compiler, via metadata. Using an object-oriented language, adaptive behaviour (3) would be directly implemented, instead of current abstraction of an expression e to $(\lambda k.e)$, where k is a parameter designating a version. Presented generalization of two translation rules is just ad-hoc solution, and it is necessary to extend it to all constructs of EBNF or BNF.

The main contribution of this work, from the viewpoint of our future research, is as follows.

Provided that a level or metalevel is adaptive, it contains feedback loops from data to code via metalevel or metametalevel, respectively. Even if any level or metalevel is adaptive, it still must be manually initiated (programmed, specified, modelled). This is an essential principle of control systems. The task of adaptiveness is to reduce this manual work, or to shift it to the higher metalevels. There is no need for a universal language, just for a multi-metalevel domain specific language, which is able to express current and future properties of a system accurately.

References

1. Krzystof Czarnecki and Ulrich E. Eisenecker: *Generative Programming: Methods, Tools, and Applications,* Addison Wesley (2005), 832 pp.
2. Matej Črepinšek and Marjan Mernik: *Inferring Context-Free Grammars for Domain-Specific Languages, Conf. on Language Descriptions, Tools and Applications,* LDTA 2005, April 3, 2005, Edinburgh, Scotland, UK, pp. 64–81.
3. Peter Ebraert and Tom Tourwe: *A Reflective Approach to Dynamic Software Evolution,* In the proceedings of the Workshop on Reflection, AOP and Meta–Data for Software Evolution (RAM–SE'04), June 2004, pp. 37–43.
4. Gregor Kiczales, Erik Hilsdale, Jim Hugunin, Mik Kersten, Jeffrey Palm and William G. Griswold: *An Overview of AspectJ,* ECOOP'01, 2001, LNCS, vol. 2072, pp. 327–355.
5. Ján Kollár: *Object Modelling using Process Functional Paradigm,* Proc. ISM'2000, Rožnov pod Radhoštěm, Czech Republic, May 2–4, 2000, pp. 203–208.
6. Ján Kollár: *Unified Approach to Environments in a Process Functional Programming Language,* Computing and Informatics, 22, 5, 2003, pp. 439–456.
7. Karl Lieberherr: *Adaptive Object-Oriented Software: The Demeter Method with Propagation Patterns,* (1996), Northeastern University May 23, 1995, 33 pp.
8. Hidehiko Masuhara and Gregor Kiczales: *Modeling crosscutting in aspect-oriented mechanisms,* In ECOOP 2003 – Object-Oriented Programming European Conference, Springer-Verlag, 2003, pp. 2–28.
9. Simon L. Peyton Jones and John Hughes (editors): *Report on the Programming Language Haskell 98—A Non-strict, Purely Functional Language,* February 1999, 163 pp.
10. Friedrich Steimann: *The paradoxical success of aspect-oriented programming,* OOPSLA 2006, 2006, pp. 481–497.

11. Philip Wadler: *The essence of functional programming,* In 19th Annual Symposium on Principles of Programming Languages, Santa Fe, New Mexico, January 1992, pp. 1–14.
12. Xiaoqing Wu, Suman Roychoudhury, Barrett R. Bryant, Jeffrey G. Gray, and Marjan Mernik: *A Two-Dimensional Separation of Concerns for Compiler Construction,* Proceedings of the 2005 ACM symposium on Applied computing, 2005, pp. 1365–1369.

Proceedings of the International Multiconference on Computer Science and Information Technology pp. 1093–1101
ISSN 1896-7094
© 2007 PIPS

Separate compilation of grammars with Tatoo

Julien Cervelle, Rémi Forax, and Gilles Roussel

Université Paris-Est
Laboratoire d'informatique de l'Institut Gaspard-Monge UMR CNRS 8049
5 bd Descartes, 77454 Marne la Vallée Cedex 2, France

Abstract. This paper presents an extension of the Tatoo compiler compiler that supports separate compilation of formal grammars. It allows the developer to define reusable libraries of grammars such as those of arithmetic expressions or of classical control operators. The aim of this feature is to simplify the development of domain specific languages especially for non specialists in grammar writing.

1 Introduction

This paper presents an extension of the Tatoo compiler compiler [2] that provides the developers of formal grammars with separate compilation. It allows the developer to use predefined libraries of general purpose grammar, possibly with its semantic support. Surprisingly this feature is not available in popular compiler compilers such as Bison [1], ANTLR [7] or JavaCC [5].

We consider that such a mechanism would simplify the elaboration of formal grammars. This is especially necessary for non-specialists of grammar writing who want, for instance, to develop small domain specific languages (DSL) without being bothered by the cumbersome design details of long-time and oftentimes already written grammars. Analogous module mechanisms are available in LISA [6] or Rats! [4]. However, they work at source level and need re-compilation of all the parts of the grammar.

By separate compilation, we mean that the developer should be able to split the specification of the grammar into several sub-parts and to compile each independently into parsing tables. Afterwards, they can link these grammar fragments together dynamically with minimum overhead. With such a mechanism it is then possible to create pre-compiled libraries, such as arithmetic expressions or classical control structures, and to reuse them in different contexts at no extra cost. Moreover, this mechanism allows semantics to be embedded along with pre-compiled grammars, and possibly several different semantics for various purposes (building abstract syntax trees, writing intermediate code or java bytecode, etc.). Also, separate compilation provides modularity of error reporting, simplifying the debugging. There can be no LR conflicts neither coming from precompiled libraries nor due to the linking phases.

The base principle of the separate compilation provided by Tatoo is to construct special parser tables for incomplete grammars and to be able, at runtime, to call another parser when necessary. The dynamic linking only requires

specifying the precise parser (grammar) to be plugged where the grammar is incomplete. At runtime, the basic idea is to switch from one parser to another when a parse error is encountered if the parser is at a branch point. Note that with this approach, the current running parser is always of utmost importance: the branching is done only if an error occurs. When the running parser is unable to recognize the input sequence, either a sub-parser is called or the error is considered as the end-of-input in order to exit from the sub-parser.

The rest of this paper is organized as follows. Section 2 provides details about the separate compilation and dynamic linking in Tatoo. Section 3 presents a complete example illustrating these features. Section 4 presents related works and a conclusion.

2 Detailed mechanisms

In this section the details of the mechanisms that provide the separate compilation and the dynamic linking are presented.

2.1 Specification

In order to provide separate compilation, the developer must be able to specify incomplete grammars. This feature is supported by three of Tatoo's mechanisms in the grammar specification.

First, to specify an incomplete grammar, the developer omits some parts of the grammar specifying branch non-terminals. For each of these branch non-terminals, at runtime, another grammar (possibly also incomplete) is associated. In order to differentiate intentional splitting of the grammar from developer's errors that are statically checked (such as a non-terminal misspelling or an unintentional mistakes), the branch non-terminals are explicitly designated. This is done through the special keyword `branches` in the grammar specification.

Second, to enable different input points in the grammar, it is possible in Tatoo to specify several axioms [2]. In the context of the construction of libraries, this mechanism permits, for instance, the factorization of similar parts of different grammars into a single one, allowing the semantics associated with it to be shared. Multiple axioms may be specified in Tatoo using the `starts` keyword.

Finally, the notion of version available in Tatoo [2] allows the developer to make its library grammars evolve without breaking the links for the users of preceding versions.

2.2 Compilation

The compilation process of incomplete grammars consists of two steps. First, branch points are considered from the compilation point of view, exactly as classical terminals of the grammar (even if from a grammar theory point of view, the branch points behave more like non-terminals) and the construction of SLR, LR(1) or LALR(1) tables are produced by the Tatoo engine. Then, the base idea

is to consider that parser branching or exiting will be performed when an error occurs. In the context of Tatoo where the parser provides the set of terminals it anticipates to the lexer, this always corresponds to lexer errors[1]. The semantics of this behavior is to consider that the running parser is of utmost importance until the lexer is unable to recognize the current input sequence. These errors may correspond to a branch point or to the end-of-input of a sub-parser or to real parsing errors. Thus, for all of the states where there is a transition through a terminal corresponding to a branch point or the end-of-input terminal, Tatoo statically replaces the error action with an access to a branching table.

The branching table associates to each state the action to perform if an error occurs. There are three kinds of actions in the branch table: a classical reduction, an enter into a new parser or exit of the current parser. More precisely, for each state containing a non-error action for the end-of-input terminal, the branch table associates to this state either an exit action - if the action in the original table was an accept - or a reduce action - if the action in the original table was a reduce. For each state which has a non-error action for a branch point terminal, the branch table associates to this state either a reduce action - if the action in the original table was a reduce - or an enter action - if the action in the original table was a shift. All of the actions associated to branch points are removed from the original table since terminals associated to branch points will never be produced by the lexer.

The following table summarizes this behavior:

Lookahead	Action	Branch action
end-of-input	`accept`	`exit`
end-of-input	`reduce`	`reduce`
branch point	`shift`	`enter`
branch point	`reduce`	`reduce`

During the construction of the branch table some conflicts may occur since different actions can be associated to the end-of-input and multiple branch points in a given state. Among classical *shift-reduce* and *reduce-reduce* conflicts, *shift-shift* conflicts, impossible in classical table constructions, could occur. For instance, if there is a state where two branch points are possible. Indeed, since, in our approach, the actions associated to end-of-input and the branch points all go into the same table, it is sometimes impossible to choose which action to perform on errors: enter, reduce or exit. These conflicts are detected statically when the branch table is constructed for the incomplete grammar. They may be resolved in Tatoo using Tatoo's priorities associated by the developer with the end-of-input and the branch points.

2.3 Dynamic linking

Now, that we have provided separate compilation of grammar tables, the tables of different grammars have to be linked together to obtain a global parser.

[1] Our approach is also usable, though a bit less efficiently, with parser errors.

The user has to specify the association between the branch points and the sub-grammars. The association provides a start axiom and the version to be used by the sub-parser. This declarative specification is used to generate a Java implementation that registers proper listeners for enter and exit actions. It also creates an instance of each lexer and parser that will be necessary during the parsing process so that no new memory is needed during the parsing, except possibly parser stack extension.

2.4 Runtime

At runtime, the top level lexer and parser are started like classical Tatoo parsers. If a token is recognized by the lexer, since Tatoo runtime only activates rules corresponding to the terminals leading to non-error actions, the parser never fails and the classical table is used to retrieve the action, shift, reduce or accept that needs to be performed. When a lexer error occurs the branch table of the current grammar is queried.

If a reduce action is encountered, it is performed, a new state is reached in the branch table and a new action has to be perform.

If an enter action is found, the registered listener looks at the lexer and parsers instances to be activated, pushes a new stack frame to allow recursive calls to the same grammar and call the new parsing process. Moreover, the input characters previously read by the lexer before the error occurred, must be "pushed back" .

If it is an exit action, the registered listener closes the parser and the lexer, returns to the enclosing parser and lexer and restarts them in a state that corresponds to the shift on the branch point.

Otherwise, a parsing error is thrown and the error recovery policy is activated.

The Figure 1 displays a diagram of the connections between lexers, parsers and other mechanisms.

However, in special cases, the enter and the exit actions may result in a runtime error. The enter action may result in an error if no listener has been registered by the dynamic linking phase. These kinds of errors can be detected by the linking phase, checking that every branch point of a grammar is associated to an axiom of "another" grammar. The exit action may also result in an error if the parser is the top-most in the stack.

2.5 Dealing with the semantics

Contrary to most parser generators [7, 5, 3, 9, 1] Tatoo's grammar specification does not embed any foreign languages (like C, Java etc.) to express the semantics [2]. Tatoo's specification requires a name for each production and a type for each terminal and non-terminal. Based on this information Tatoo generates a Java interface that allows the developers to implement the semantics. Thus, the semantics of grammar libraries is easily reusable.

A developer can choose to provide a single class for the whole parser by implementing all of the semantic interfaces or to reuse library classes implementing

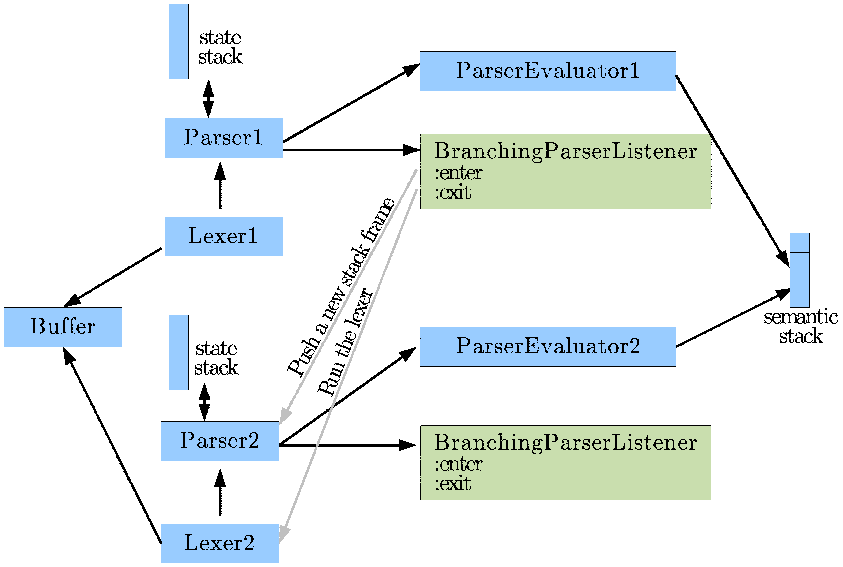


Fig. 1. Transfer between lexers and parsers at runtime

the semantics of each remaining grammar part. The sharing of the semantic stack allows the semantics to work altogether: when the embedded parser exits, the result of its evaluation is pushed on the common semantic stack. The enclosing parser accesses the value popping it from the stack.

3 Example

In this section, we present a complete example demonstrating how the branching feature of Tatoo can be used to develop "grammar libraries" and how it helps people who are not accustomed to writing grammars to incorporate them into simpler grammars.

Supposing that one wants to design a simple DSL for an application, such as simple routines to automate tasks on an application. Usually, the syntax for the routine carries the form of any programming language but with special entry points and special end points and constructions.

For instance, in a spreadsheet program, the entry point is a formula following an "`=`" symbol, and the end points are the special naming of cells (like `A3:A4`, `A$3`, etc.) and function call.

3.1 Defining the library

To this end, one could use an already written and compiled grammar for all mathematical expressions and control structures.

The following grammar is an example of the grammar written in Tatoo's ebnf format:

```
priorities:
  plus_minus= 1 left
  star_slash= 2 left
tokens:
  lpar= '\('
  rpar= '\)'
  plus= '\+'  [plus_minus]
  minus= '-'  [plus_minus]
  star= '\*'  [star_slash]
  slash= '\/' [star_slash]
blanks:
  space= "( |\t|(\r?\n))+"
branches:
  idBranchPoint
starts:
  expr
productions:
  expr = expr 'plus' expr    [plus_minus]
       | expr 'minus' expr   [plus_minus]
       | expr 'star' expr    [star_slash]
       | expr 'slash' expr   [star_slash]
       | 'lpar' expr 'rpar'
       | 'idBranchPoint'
       ;
```

In this specification one should notice in the `branches` section, the use of the branch point `idBranchPoint`. This allows the developer to choose their own atomic expressions.

This specification is then taken as an input of the Tatoo compiler. It generates the tables related to this grammar.

3.2 Using the library

Now that the library is available, the user only needs to write the grammar specific to the spreadsheet application. This specification has to contain an axiom `start` for the global parser and a special axiom `endpoint` to define the syntax for the cell description and functions.

Using the Tatoo ebnf syntax[2], the spreadsheet grammar may be specified as follows:

```
tokens:
  assign= '='
  comma= ','
```

[2] The notation `'exprBranchPoint'/'comma'*` means zero or more `exprBranchPoint` separated by `comma`.

```
  colon= ':'
  lpar= '\('
  rpar= '\)'
  dollar= '\$'
  number= '[0-9]+'
  identifier= '[a-zA-Z]+'
blanks:
 space= "( |\t|(\r?\n))+"
branches:
  exprBranchPoint
starts:
  start
  endPoint
productions:
  start = 'assign' 'exprBranchPoint'
        ;
  endPoint = 'dollar' 'identifier' identifierSuffix
           | 'identifier' identifierOrFunction
           ;
  identifierOrFunction = 'lpar' 'exprBranchPoint'/'comma'* 'rpar'
                       | identifierSuffix
                       ;
  identifierSuffix = 'dollar'? 'number' range?
                   ;
  range = 'colon' 'dollar'? 'identifier' 'dollar'? 'number'
        ;
```

Note that to work properly, neither identifiers nor functions should appear as a terminal in the grammar library. Should this be the case, an input sequence such as `A3:B4` would not be recognized properly. Indeed, in this case `A3` would be recognized as an identifier of the library and the error would occur on the ':'. For sake of efficiency, we do not want to backtrack on errors and ':' is not a correct branch point. However, as libraries are usually provided by grammar-aware individuals, they are typically well-written and DSL writers should not be aware of this difficulty.

3.3 Dynamic linking

The last part of the specification has to do with wiring up the grammars together. This is done with a specific Tatoo language. The specification details the files that contain the different grammar parts[3] and the associations between the branch points and the grammar axioms together with their version.

[3] Compiled parsers are packed-up into a specific `Jar` file with extension `.tlib` that contains their tables together with necessary meta-information.

For instance for the previous example this file looks like[4]:

```
grammars:
  expr= "grammars/expr/expr.tlib"
  spreadsheet= "grammars/spreadsheet/spreadsheet.tlib"

links:
  spreadsheet.exprBranchPoint = expr.start
  expr.idBranchPoint = spreadsheet.endPoint
```

This file is translated into a Java code. This code creates a lexer and a parser for each grammar. All the lexers share the same input buffer and each parser has its own state stack but the stack for the semantics is shared by all of them. Each parser is configured with a branching parser listen. Each listener contains an association between a branch point and a pair lexer/parser to be started on a given axiom.

4 Related works and conclusion

As far as we know there is no parser generator that supports the separate compilation of grammars.

Several tools such as ANTLR2 [7], Rats! [4] or LISA and its extensions [6, 9] provide built-in features that simplify software engineering and modular development. However, these tools require the construction of a global specification before the generation of tables. They can be viewed as pre-processors in the compiler compiler chain.

The ELL Parser Library [8] is a tool that supports the dynamic extension of parsers. It allows the developer to extend the grammar at runtime and uses a backtracking mechanism to solve the ambiguities in the LL table extended dynamically. It proposes a mechanism to suspend the parser at runtime which is comparable to Tatoo's one.

Tatoo's approach is not in contradiction with the previous approaches. However, to preserve online parsing with low memory consumption, we chose not to introduction backtracking in Tatoo's implementation. Moreover, Tatoo's modularity works at a lower level. It avoids the construction of a global grammar before the generation of a parser and it sees the parser as a set of small cooperating parsers. A very simple semantics has been adopted to follow or to exit branch points in the grammar. The current running parser is always of the utmost importance and branching points, or leaving of the parser are only considered when the lexer detects an error. Priorities may be used to solve potential conflicts.

Further work under consideration involves checking if other mechanisms that support modularity such as inheritance or aspect programming can be implemented using separate compilation. This will probably require to adding a

[4] Since our grammar examples do not contain version, no version information appears in this file.

branching mechanism to all non-terminals which was not our original design choice. We could also use a Tatoo branching mechanism to implement dynamic extensions of the grammar since Tatoo already supports the creation of tables at runtime.

Moreover, it is still not clear whether or not this splitting approach extends to the class of global grammars accepted as input for Tatoo. Further investigation needs to be undertaken in this direction.

References

1. Anthony Aaby. Compiler construction using flex and bison. http://cs.wwc.edu/~aabyan/464/Book/, 1996.
2. Julien Cervelle, Rémi Forax, and Gilles Roussel. Tatoo: An innovative Parser Generator. In *Proc. of the 4th Int. Conf. PPPJ'06*, ACM International Conference Proceedings, pages 13–20, Mannheim, Germany, August 2006.
3. Etienne M. Gagnon and Laurie J. Hendren. SableCC, an object-oriented compiler framework. In *Technology of Object-Oriented Languages and Systems*, pages 140–154. IEEE Computer Society, 1998.
4. Robert Grimm. Better extensibility through modular syntax. In *Proc. of PLDI'06*, pages 38–51, 2006.
5. Viswanathan Kodaganallur. Incorporating language processing into java applications: A JavaCC tutorial. *IEEE Software*, 21(4):70–77, August 2004.
6. Marjan Mernik, Nikolaj Korbar, and Viljem Zumer. LISA: A tool for automtic language implementation. *SIGPLAN Notices*, 30(4):71–79, 1995.
7. Terence J. Parr and Russell W. Quong. ANTLR: A predicated-LL(k) parser generator. *Software Practice and Experience*, 25(7):789–810, 1995.
8. Christian Plesner Hansen. *An Efficient, Dynamically Extensible ELL Parser Library.* PhD thesis, University of Aarhus, Denmark, May 2004.
9. Damijan Rebernak, Marjan Mernik, Pedro Rangel Henriques, and Maria João Varanda Pereira. AspectLISA: An aspect-oriented compiler construction system based on attribute grammars. *Electr. Notes Theor. Comput. Sci.*, 164(2):37–53, 2006.

Proceedings of the International Multiconference on Computer Science and Information Technology pp. 1103–1112
ISSN 1896-7094
© 2007 PIPS

Flexible Object Representation using Properties

Koen Vanderkimpen, Marko van Dooren, and Eric Steegmans

K. U. Leuven
Department of Computer Science
Leuven, Belgium
{Koen.Vanderkimpen,Marko.vanDooren,Eric.Steegmans}
@cs.kuleuven.be

Abstract. Many object-oriented programming languages use fields to represent object state. But subclasses cannot alter the representation of an object sufficiently. Because of that, a number of subclassing problems are difficult or impossible to deal with if the superclasses were not designed to anticipate them, especially under multiple inheritance. Moreover, in languages such as Java and C#, encapsulation necessitates a lot of boilerplate code.
In this paper, we improve the C# concept of properties, making them more flexible for use with inheritance and reducing their boilerplate code. Using our properties makes it easier for programmers to model programs in a more consistent manner. Furthermore, our Properties allow redefining an object's attributes in ways that equal the possibilities for redefinition of virtual methods in many programming languages, which makes them better suited to deal with unanticipated reuse. Specifically, using our construct, it becomes possible to join several superclass attributes into only one at the subclass level, conjointly decreasing memory consumption.

1 Introduction

In object-oriented programming languages, the representation or state of objects is stored in fields. Classes inherit all features, including those fields, from their superclasses. This has an adverse effect on a number of subclassing problems that are already difficult or impossible to deal with if the superclasses were not designed to anticipate them. One problem is consistency: when two superclass fields represent the same concept in a subclass, it requires effort to ensure they remain equal at all times. Another problem is redundancy: when a superclass field is no longer used by a subclass, it requires effort to ensure it is not used by accident.

Such problems are even worse when using multiple inheritance. For example, when a class inherits from two similar superclasses, we can select and refine features of both, but when each class introduces a field for one and the same feature, we get an overlap in state, and thus, again, a redundant field.

Beyond redundant state, unanticipated subclassing can expose several other problems. For example, a field that is not encapsulated in the superclass makes

it impossible to impose any constraints on it [12], or an encapsulated field, accessed by other methods than by its getter and setter, increases the chance for unexpected side effects when these methods, or the getters and setters, are overridden.

Effectively dealing with these kinds of problems in current programming languages is difficult, especially if the classes were not designed with this kind of subclassing in mind. For example, redundant state and most other problems can be anticipated and/or solved by clever programming methods and design patterns [4, 8], or by completely reengineering a (self-contained) program [14, 15]. These solutions, however, are not elegant because they blur the connection between the modeled state and the code used to model it. They also require a lot of effort: both from the creators of the superclasses, who have to anticipate reuse, as from the designers of the subclasses, who need to carefully override many methods.

In addition, this effort usually involves introducing a lot of boilerplate code. In Java, for example, when we want proper encapsulation of a property, but still want to make it available to clients, we first need to write a getter and a setter for it. Thus the current solutions are not satisfying: redundant fields should be non-existant fields, boilerplate code should be non-existant code, and useless effort should be non-existant effort.

This paper tackles the inflexibility with which stored features are inherited by improving the language concept properties. Our properties combine the benefits of C# properties [7] with those of Eiffel's attributes [11]. They allow a more consistent relation between code and the state it models, which allows programmers to better reflect the modeled state in the code, increasing consistency. Furthermore, using our properties makes it possible to use simple inheritance instead of more elaborate design patterns when we need to refine certain kinds of class behaviour when it was not anticipated. Their use disallows access to an object's stored fields other than through the associated properties, and they can be redefined in ways that equal the possibilities for redefinition of regular methods in Eiffel. Specifically, using our construct, it becomes possible to join several superclass attributes into only one at the subclass level, effectively decreasing memory consumption.

In the next section, the basic improvements for isolated properties are introduced. In section 3, we explain the joining of properties in both multiple and single inheritance. Section 4 deals with derived properties and their effect on joining and inheritance. In section 5, compilation of properties is handled. Section 6 evaluates the property concept and section 7 relates it to existing work. In section 8, we present future work, and we conclude in section 9.

2 Properties: Basic Semantics

In this section, we introduce our property concept, with syntax similar to that of a property in C#. In our examples, we use a language based on Java, enriched with multiple class inheritance, as in [2].

Consider the class of `Person`s in Figure 1, which plays a central role in a framework for role playing games. A `Person` has two properties: a name and a number of hitpoints.

```
public class Person {
   public property String name;
   public property int hitPoints = 10;
}
```

Fig. 1. Properties in the class of Persons

In the example, the second line declares a private `String` field `name` and its associated public getter and setter. Just as in C#, it is possible to allow getting and setting the property as if it is a field, but using our properties, it is not necessary to declare the name and type of a field twice (as is sometimes necessary in C#: once as a field and once as a property). The main differences between a property and a normal field are automatic encapsulation and the possibility to override the getter and/or the setter to accomodate custom behaviour. In the third line, hitpoints are initialized to ten, which will implicitly invoke the default setter.

Our methods of customizing a property's behaviour are at least equally powerful and more self-contained as the possibilities offered by C#: everyting non-standard about a property can be declared inside a property block. There are however some syntactic differences. Since there no longer exists an explicitly declared private field to store the property, an implicit variable `value`, which defaults to the property's type, is used to denote it when changes, such as a different type, are needed. Another implicit variable called `argument`, which for isolated properties always has the property's own type, is used to denote the argument for the setter[1].

To illustrate this, a birthdate, represented internally by a `long` and externally by a `Date` object, has been added to the class `Person` in Figure 2. This is a robust way to circumvent the encapsulation problems associated with the Date class in Java [3].

```
public class Person {
   ...
   public property Date birthDate {
      long value = 0;
      get { return new Date(value); }
      set { value = argument.getTime(); }
   }
}
```

Fig. 2. A customized property

Here already, a beneficial semantic difference to the way the private field is stored in this paper's properties emerges: when we would override a property

[1] Note that the name `value` is differently used in C#, where it denotes the setter's argument

from a superclass and use the `value` keyword to redeclare it, the now redundant superclass field will be replaced instead of merely becoming hidden, saving memory space. This is possible because the encapsulation of the value inside the property block ensures there cannot occur unexpected side effects resulting from its use outside the block.

3 Joining Properties

When subclassing, two or more properties coming from the superclasses can be merged together, freeing up memory slots in the subclass objects. This is used when a relationship between two previously independent variables from the superclasses occurs in the subclass.

A common situation that benefits from multiple inheritance, enhanced with our properties, is the following: consider two frameworks. The first is our framework for computer games, without anything built on top of it. The second framework includes graphical user interfaces (GUIs), and the use of the Observer pattern [8] for communication between the GUI and the program logic. We want to use our model from the first framework with the GUI of the second one, but the latter introduces its own model classes. Using properties, we can simply subclass both the model classes to get the behaviour we need. For example, consider Figure 3. The Observer-based[2] framework class `Character` also introduces a name and birthdate for a person. Obviously, we do not want to store those attributes twice. Furthermore, we want the birth date to behave like it does in the class `Person`.

Joining properties can be done by placing a `join` keyword after the property declaration in the subclass, followed by a list of the superclass properties to be joined. If the join would cause inheritance of two or more conflicting versions of the getter, setter or values, they have to be overridden or one of the versions must be selected, like the `birthDate` property in Figure 3, which also illustrates renaming. One exception to this rule exists: when all superclass properties are default properties (no block) and have the same type, no further specification is needed. This is the case for the `name` property in the example in Figure 3.

Joining properties coexists with dynamic binding: the single property in the subclass is used when objects of that class get polymorphic calls to their superclass interface(s). For this to work correctly, the Liskov substitution principle [10] has to be respected.

More precisely, this means that the argument type for the setter in the new subclass should be a super type for all the joined setter argument types, since for method arguments, we have contravariance [5]. The best choice is the least upper bound, which always exists, but might degenerate to `Object`. Conversely, the greatest lower bound of all joined property types will be the new type for the joined property, because we have covariance [5] of the return types of the getters. The latter may not always exist, and such a join would thus result in a compile-time error.

[2] For simplicity, the actual code to observe the state has been omitted.

```
public class Character {
   public property String name;
   public property Date dateOfBirth;

   public void addObserver(...) {...};
}

public class CharacterPerson extends Person, Character {
   public property String name join Person.name, Character.name;
   public property Date birthDate join Person.birthDate,
       Character.dateOfBirth {
      value,get,set : Person.birthDate;
   }
}
```

Fig. 3. The classes of Characters and CharacterPersons

This selection mechanism may result in different types for the getter and setter of a property. In that case, the programmer has two options. The first is overriding the setter to accept arguments of the more general type, transforming and storing them in a private value of the more specific type. The second is to override the getter to transform a private value of the more general type into the most specific type before returning it. Beyond overriding one of the methods, it will also often be necessary to select an implementation for the other one, when the mechanism does not choose one with a desirable type.

The private value of a property is only visible inside the declaring property block and in overriding property blocks. The getter's visibility is linked to that of the property and the setter's default visibility also equals that of the property, but it can be made more restricted. This is for example needed when we want a readonly property. When a property is not private, however, the setter cannot be made less visible than protected, since it has to remain overridable.

Joining Properties in Single Inheritance

It is also possible to join multiple properties from a single superclass; the same mechanics (such as the selection mechanism) apply. The Merger of properties coming from one superclass is useful when a relationship between previously independent fields from that class occurs in the subclass, usually because of stronger class invariants. Consider a typical two-class example: a superclass `Rectangle` and a subclass `Square`. A rectangle has a `width` and a `height`, which for a square are equal and are named the `side`. This can be accomplished easily by joining the `width` and `height` of a `Rectangle` with a property `side` in the subclass `Square`, as seen in Fig. 4(a).

This has several consequences. First, it means that there is now an implicit constraint on the representation, saying that for squares, the width and height are equal (this is like making the class invariant stronger, as allowed by the

principles of substitutability [10]). Second, width and height are both renamed to side.

The ability to join properties in both single and multiple inheritance can avoid a catastrophic blow-up of unused variables in certain class hierarchies, without the need for specific hacks, such as storing all variables in a hashmap or arraylist. Consider a more extended hierarchy of quadrangles, as given in Table 4(b). Each successive subclass of `Quadrangle` requires less variables to completely define its form. Using normal inheritance without joining, the class of `Square`s would have four extra unused variables, wasting memory space for each square in the system.

```
public class Rectangle {
   public property int width;
   public property int height;
}

public class Square
      extends Rectangle {
   public property int side join
      width, height;
}
```

(a) The classes Rectangle and Square

Class	Fields Required
Quadrangle	3 lengths, 2 angles
Trapeze	2 lengths, 2 angles
Parallellogram	2 lengths, 1 angle
Rhombus	1 length, 1 angle
Rectangle	2 lengths
Square	1 length

(b) The need for fields

Fig. 4. A class Hierarchy of Quadrangles

4 Derived Properties

Often, many of an object's properties are derived from others: they are not stored separately. A square's surface and perimeter, for example, can be derived from its side. In C# [7], it is possible to use derived properties, simply by neglecting to introduce a new instance variable and programming the getter and setter to use other variables and/or methods. Using that feature, all getters without arguments in a class can be made properties.

Our properties can also be used as derived properties. When the getter and setter are overridden in the property-block without use of the `value` keyword, the compiler will infer that the property is derived and no private value need be stored. Often, the setter is omitted, since derived properties are usually readonly.

When subclassing, it is possible to change the derived status of a property to stored and vice versa. Going from stored to derived is the same as declaring a new derived property: we write a property block without using the `value` keyword.

Consider the `Person` subclass `MyPerson` in Figure 5. We introduce an integer property `vitality` and we make a `Person`'s hitpoints equal to this vitality multiplied[3] by ten.

[3] we ignore overflow in the getter

```
public class MyPerson extends Person {
   public property int vitality;
   public property int hitPoints {
      get { return vitality*10; }
      set { vitality = argument/10; }
   }
}
```

Fig. 5. The class of MyPersons

For the reverse, going from derived to stored, it is possible to write a new property-block that uses the `value` keyword, but we can also simply redeclare the property without block, in which case the default stored implementation will override the previous one.

Derived properties can be joined just like stored ones, and both types can be mixed together. Depending on which versions of getter and setter are selected, the resulting property will use zero to one memory slots. When the selection mechanism would choose a getter or setter from a stored property being put together with a setter or getter from a derived one, we have to override or manually select both.

5 Transformation and Compilation

We have written a tool for the automatic transformation of our version of property-enhanced Java to regular Java code. It makes use of the Chameleon [16] framework for metamodels of programming languages. It parses the code and transforms the AST to a model, then transforms that model to a regular Java-model, and finally outputs the code for the transformed model.

The tranformation makes use of the State pattern [8]. For each class in the source model, two classes appear in the output model: one for the class and all its functionality without variables, and one for the representation only. The former classes follow the normal inheritance hierarchy, but the latter all inherit from the interface `Representation`. This separation between behaviour and state makes it possible to simulate the removal of stored variables and thus the joining and overriding of properties.

Such a simple transformation to Java obviously fails to support the memory-efficiency benefits of properties, since, for each object in the source model, we now have two objects in the transformed model. Therefore, to realize the full potential of properties, we need some form of direct compilation to byte-code. A deeper investigation of how properties could be compiled would draw us too far in this paper, but we can at least conclude it would necessitate fields being treated as redefinable (virtual) methods and included in a virtual function table, instead of the fixed position in an object's memory block they have now. This would cause a space/time trade-off, since stored state takes longer to access in this way, but there are no longer any unused variables included in that state.

Nevertheless, when programming in object-oriented languages in a decent way (certainly in Java), fields are encapsulated in getters and setters, which also requires a method lookup. Bearing this in mind, the difference in lookup-time is neglectable. Furthermore, much research has been done in the area of eliminating virtual function calls in optimizing compilers [1, 9, 17].

6 Evaluation

The use of properties and the ability to join them have several advantages. Above all, we can represent objects more consistently than in other object-oriented languages. If a square has one defining feature while a rectangle has two, we are able to drop one and make `Square` a subtype of `Rectangle`, as it is in the real world. If we inherit from multiple classes and two or more of them have implemented a similar property, we can join those properties and all of their corresponding methods easily. This increases the consistency between code and the real world.

Another advantage is simplicity: programmers don't have to worry about rewriting methods to make sure a redundant variable is no longer used in a subclass. The alternative, to keep several existing variables consistent behind the scenes, is at least equally cumbersome to accomplish. All getters and setters are merged by joining properties, and the other, more complex methods, are written in function of those, so they don't have to change.

The compilation of properties necessitates a virtualization of stored features, potentially decreasing runtime efficiency. This, however, is balanced by a better memory efficiency and countered by constantly improving compiler optimization techniques. Furthermore, the same reduction in runtime efficiency occured before, when we manually encapsulated those features, whereas now, they are automatically encapsulated. Thus, practically, there is no real decrease. In any case, when it comes to being able to write code more correctly, faster and more easily, we feel that runtime efficiency is of secondary importance. With today's increasingly performant machinery, well-designed and reusable code is a lot more important than being able to execute it faster.

7 Related Work

In C# [7], it is possible to define a property block, but the getter and setter still need to be explicitly implemented, as well as the private field that stores the property. Moreover, in C#, properties cannot be joined and stored properties cannot be eliminated.

In Eiffel [11], features (methods and attributes) inherited from multiple superclasses can be joined by renaming and undefining them in all except one of the superclasses. For this to work, they need to be renamed to the same name as the feature in the class in which they are not undefined. But it is not possible to rename two features of a single superclass to the same name; that would introduce an ambiguity. Deferring one of them is not an option, since the rename

clause has to preceed the undefine clause, and thus, when they are renamed to the same name, they would both be deferred.

Furthermore, it is possible in Eiffel to merge two or more stored properties, also called attributes, when they are from superclasses with a common base class, but not in other cases. In the language, we can only turn derived properties into stored ones, not vice versa.

In Cecil [6], fields are similar to our properties. An instance variable, or field, can be declared, and this will automatically be hidden. A getter and setter method are also automatically created for such a field. Because the representation is always wrapped by methods, it becomes possible in Cecil to override methods by instance variables and vice versa. Joining fields explicitly without having to override many methods, however, is not possible.

In the dynamically-typed language Self [13], variables and procedures are united into a single construct. Because it is prototype-based, redefinitions not resulting in redundant state, similar to those for properties, are possible. As in Cecil, explicit joining is not supported.

8 Future Work

We would like to introduce constraints on properties as a language concept, as is possible for methods in Eiffel. This will allow us to better explore the consequences of using and joining properties on class contracts. If we were able to associate invariants with properties, it would be possible to see a join between properties as simply an extra class invariant, declaring equality between existing properties.

9 Conclusion

We have shown how the current storage of fields in statically typed object-oriented languages is inadequately able to cope with certain changes when subclassing, which results in redundant fields or an obscured connection between the modeled state and the code. We have also explained how this may unnecessarily complicate a programmer's work.

To solve these problems, we have improved the concept of properties in those languages. Properties are a kind of advanced attributes, which encapsulate fields and automatically introduce getters and setters in a class. Properties can be made stored or derived, and this can be changed in subclasses. Derived properties do not take up space in memory. Properties can also be joined in subclasses, stating that in the subclass, several superclass attributes always have the same value. Joined properties use only one slot of memory. The use of properties helps to program an object's representation more logically and consistently, and increases the memory efficiency of programs. Furthermore, it reduces boilerplate code.

References

1. Gerald Aigner and Urs Hölzle. Eliminating virtual function calls in C++ programs. In *ECOOP '96: Proceedings of the 10th European Conference on Object-Oriented Programming*, pages 142–166, London, UK, 1996. Springer-Verlag.
2. Lorenzo Bettini, Michele Loreti, and Betti Venneri. On multiple inheritance in Java. In *Technology of Object-Oriented Languages, Systems and Architectures, Proc. of TOOLS Eastern Europe 2002*, pages 1–15. Kluwer Academic Publishers, 2003.
3. Joshua Bloch. *Effective Java: Programming Language Guide*. Java series. Addison-Wesley, Boston, 2001.
4. Grady Booch. *Object-oriented analysis and design with applications (2nd ed.)*. Benjamin-Cummings Publishing Co., Inc., Redwood City, CA, USA, 1994.
5. Giuseppe Castagna. Covariance and contravariance: conflict without a cause. *ACM Trans. Program. Lang. Syst.*, 17(3):431–447, 1995.
6. Craig Chambers. The Cecil language: Specification and rationale. Technical Report TR-93-03-05, 1993.
7. Microsoft Corporation. *Microsoft C# Language Specifications*. Microsoft Press, Redmond, WA, USA, 2001.
8. Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides. *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison Wesley, Reading, Massachusetts, 1994.
9. Kazuaki Ishizaki, Motohiro Kawahito, Toshiaki Yasue, Hideaki Komatsu, and Toshio Nakatani. A study of devirtualization techniques for a Java just-in-time compiler. In *OOPSLA '00: Proceedings of the 15th ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications*, pages 294–310, New York, NY, USA, 2000. ACM Press.
10. Barbara H. Liskov and Jeannette M. Wing. A behavioral notion of subtyping. *ACM Trans. Program. Lang. Syst.*, 16(6):1811–1841, 1994.
11. Bertrand Meyer. *Object-oriented software construction (2nd ed.)*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1997.
12. D. L. Parnas, P. C. Clements, and D. M. Weiss. Enhancing reusability with information hiding. In *Software reusability: vol. 1, concepts and models*, pages 141–157, New York, NY, USA, 1989. ACM Press.
13. Randall B. Smith and David Ungar. Self: The power of simplicity. In *LISP and Symbolic Computation*, pages 187–205, Netherlands, 2005. Springer.
14. Gregor Snelting and Frank Tip. Reengineering class hierarchies using concept analysis. In *SIGSOFT '98/FSE-6: Proceedings of the 6th ACM SIGSOFT international symposium on Foundations of software engineering*, pages 99–110, New York, NY, USA, 1998. ACM Press.
15. Frank Tip and Peter F. Sweeney. Class hierarchy specialization. *OOPSLA SIGPLAN Not.*, 32(10):271–285, 1997.
16. Marko van Dooren. *Abstractions for improving, creating, and reusing object-oriented programming languages*. Phd, Department of Computer Science, K. U. Leuven, Leuven, Belgium, June 2007.
17. Yoav Zibin and Joseph Gil. Two-dimensional bi-directional object layout. In *ECOOP '03 - Proceedings of the 17th European Conference on Object-Oriented Programming*, pages 329–350, Berlin, Germany, 2003. Springer-Verlag.

Proceedings of the International Multiconference on
Computer Science and Information Technology, pp. 1113 – 1122
ISSN 1896-7094
© 2007 PIPS

A Generator of SQL Schema Specifications

Slavica Aleksić[1], Ivan Luković[1], Pavle Mogin[2], Miro Govedarica[1]

[1] University of Novi Sad, Faculty of Technical Sciences,
21000 Novi Sad, Trg Dositeja Obradovića 6, Serbia
{slavica, ivan, miro}@uns.ns.ac.yu
[2] Victoria University of Wellington,
Wellington, P.O. Box 600, New Zealand
pmogin@mcs.vuw.ac.nz

Abstract. IIS*Case is an integrated CASE tool that supports the automation and intelligent support of complex and highly formalized design and programming tasks in the development of an information system. IIS*Case generates relational database schemas in 3 rd normal form with all relevant data constraints. SQL Generator is an IIS*Case tool that generates the implementation specification of a database schema according to ANSI SQL:2003 standard. The generator may also produce a database schema specification for Microsoft SQL Server or Oracle DBMSs. The paper describes SQL Generator's traits, considers aspects of its application, and shows its use in the implementation of a complex database constraint using procedural mechanisms of a particular relational DBMS. SQL Generator is implemented in Java and Oracle JDeveloper environment.

1 Introduction

Integrated Information Systems*Case (IIS*Case) V.6.2 is a tool that provides the automation and an intelligent support for performing complex and highly formalized design and programming tasks in the development of an information system. It is designed to provide complete support for: (i) developing database (db) schemas that are complex with regard to the number of concepts used, and (ii) software applications of an information system.

The form type is one of the main IIS*Case concepts. It is semantically rich enough to enable expressing the elements of the static and dynamic structures in an application domain. By means of form types, a designer creates simultaneously a model of the structure and behavior of various business documents, and a conceptual db schema. Starting from the created conceptual schema, IIS*Case automatically generates a relational db schema in 3rd normal form (3NF) with all relevant data constraints. Detailed information about IIS*Case and its main concepts may be found in several authors' references, as well as in [5, 8, 13, 14].

IIS*Case is designed to provide a fast generation of db schemas and application prototypes. Although the concept of the form type is highly formalized, it is very close to the perception power of an average user. Therefore, IIS*Case may support an intensive and efficient communication among designers and users of an application domain throughout the software development process. Consequently, we believe that it is a tool suitable for the application in the agile software development. A case study illustrating main features of IIS*Case and the methodological approach to its usage is given in [5].

SQL Generator is a tool of IIS*Case that generates SQL specifications of relational db schemas. One of the main reasons for the development of such a tool was to make db designer's and developer's job easier, and particularly to free them from manual coding and testing of SQL scripts. The goal was to provide an efficient transformation of design specifications into error free SQL specifications.

There are a number of CASE tools that provide generation of SQL scripts. Some of them are described in [2, 12, 15]. One of the advantages of our SQL Generator is that it provides the implementation of some special cases of db constraints. For example, in contrast to Oracle Server Generator [12], and Sybase Power Designer PDM [15], IIS*Case SQL Generator provides the implementation of not only the default, but also the partial and the full referential integrity constraints, according to [4]. For all those types of referential integrity constraints, SQL Generator also allows selecting the following actions: No Action, Cascade, Set Default and Set Null, both for deleting and updating in the referenced table. Besides, SQL Generator provides the implementation of the inverse referential integrity constraints [8, 9], which are not rare in the real world and to the best of our knowledge, neither of the other CASE tools provides the same functionality. SQL Generator also produces a trigger that prohibits updates of a relation scheme primary key, if such a rule is specified in the design.

SQL Generator provides creating SQL scripts according to the syntax of: (i) ANSI SQL:2003 standard [4], (ii) DBMS Microsoft (MS) SQL Server 2000/2005 with Microsoft T-SQL [6, 7], and (iii) DBMS Oracle 9i/10g with Oracle PL/SQL [10, 11].

In this paper we present basic features of SQL Generator that are already implemented, and aspects of its application. We also present methods for implementation of a selected db constraint, using mechanisms provided by a relational DBMS. A complete description of SQL Generator may be found in [1].

2 Generating the SQL Specifications of a DB Schema

IIS*Case generates 3NF relational db schemas with all the relation scheme keys, null value constrains, unique constrains, referential and inverse referential integrity constraints. These schemas are stored in the IIS*Case repository. The specification of the IIS*Case repository is given in [13]. The input into SQL Generator is a schema stored in the repository.

Using SQL Generator, a user may produce SQL scripts for the creation of tables, views, indexes, sequences, procedures, functions and triggers, even without knowing SQL syntax and mechanisms for the implementation of constrains of a selected DBMS. SQL Generator may produce scripts for implementing a new db schema, or modify an already existing one in the following three ways: (i) by creating SQL scripts in files only for a later execution, (ii) by creating and immediately executing SQL scripts under a selected db server with an established connection, and (iii) by creating and immediately executing SQL scripts on a selected data source with an established connection via an ODBC driver. In all three cases, generated SQL scripts are stored in one or more files.

Figures 1-5 present screenshots of a form that is used to define values of SQL Generator input parameters. The field *DBMS* (Fig. 1) enables the selection of the type and version of a target db server. The radio button *DDL Files only* (Fig. 1) provides the creation of SQL scripts in files only. The scripts may be created in one, or more files

(see the check box *One File Only* in Fig. 1). If a user selects the former option, separate files are created for tables, constraints, triggers and indexes. The main command file is also generated. It contains calls to all the other script files. The radio button *Database Source* (Fig. 1) enables the selection of either Oracle or MS SQL db server, establishing a connection, and immediate execution of SQL scripts. In this case, SQL Generator creates a script file, invokes the appropriate SQL tool, and passes necessary parameter values for the script execution. The radio button *ODBC Source* (Fig. 1) enables the creation and the immediate execution of SQL scripts in a selected ODBC data source. An appropriate ODBC driver for the target db server must be installed and configured. SQL Generator supports the user authentication when it works via an established connection. The field *DB Schema Name* (Fig. 1) enables defining a db name that is then included in an appropriate CREATE DATABASE command.

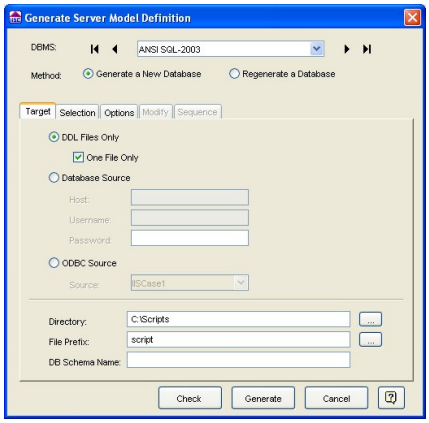


Figure 1. SQL Generator - Target panel

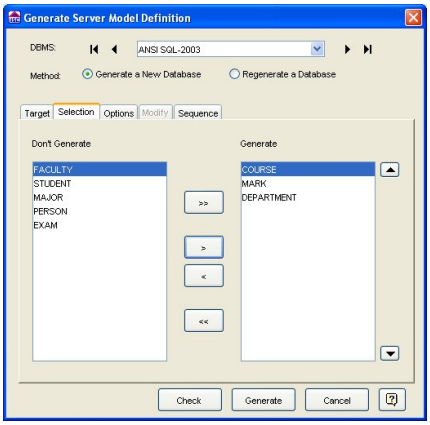


Figure 2. SQL Generator – Selection panel

By means of the *Selection* panel (Fig. 2), a user picks relation schemes. SQL Generator will produce the appropriate SQL commands for the selected relation schemes only, and place them in script files. The list of selected relation schemes should not be empty. Otherwise, a user will get a warning and the focus will be returned to the *Selection* panel automatically.

By means of the *Options* panel (Fig. 3), a user defines which types of db objects are to be generated. By checking the appropriate check-box items (Fig. 3), he or she may decide to generate: (i) indexes for primary, alternate and foreign keys, (ii) SQL CONSTRAINT clauses, (iii) triggers, and (iv) comments.

If the *Generate SQL CONSTRAINT Clauses* check-box is checked, SQL Generator produces the following CONSTRAINT clauses: PRIMARY KEY, UNIQUE, CHECK and FOREIGN KEY, for each key, unique, tuple, or foreign key constraint in db schema that may be implemented in a declarative way. If *Generate Triggers* check-box is checked, SQL Generator will produce all the db triggers, procedures and functions, necessary to implement db constraints that cannot be expressed in a declarative way. For inverse referential integrity constrains [8, 9], SQL Generator offers two ways of implementation: (i) by means of SQL views and the appropriate stored procedures, or (ii) by means of stored procedures only. If *Include Comments*

check-box is checked, SQL Generator will create comments in SQL code, comprising creation date and time, and the selected type and version of the DBMS.

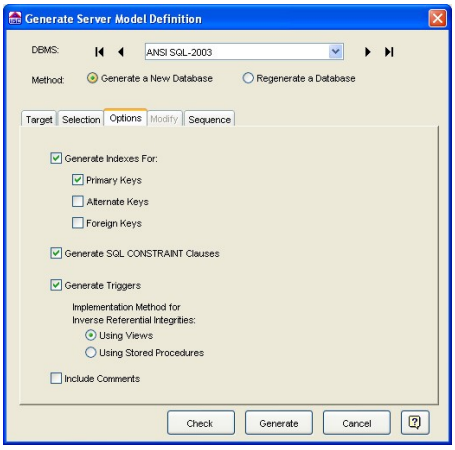


Figure 3. SQL Generator - Option panel

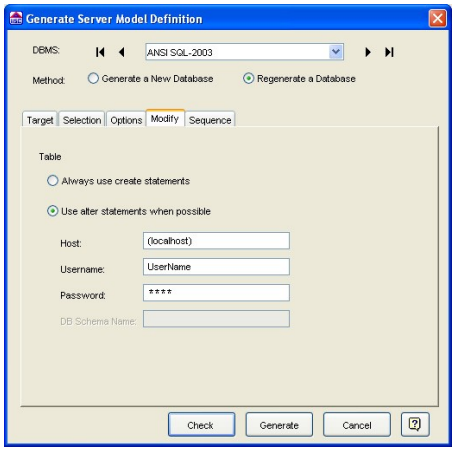


Figure 4. SQL Generator - Modify panel

A user can select one of the following script generating methods. If the radio button *Generate a New Database* is selected (Fig. 1-5), SQL Generator will produce scripts with CREATE statements, for the implementation of a new db schema under a DBMS. Otherwise, if *Regenerate a Database* is selected, it will regenerate, i.e. modify an already implemented db schema under a DBMS. In the later case, the *Modify* panel (Fig. 4) is used. SQL Generator uses the values for *Host* , *Username* , *Password* and *DB Schema Name* (Fig. 4) to establish a connection to the target database, compares the information from IIS*Case repository with the information obtained from the data dictionary of the target database, and generates scripts containing the appropriate CREATE, ALTER and DROP statements. If a user selects the radio button *Always use create statements* , each modification is accomplished by dropping old and then creating objects. Otherwise, if *Use alter statements when possible* is selected, SQL Generator will produce ALTER statements, whenever it is possible.

SQL Generator also supports the creation of sequence generators. A sequence generator specification is defined by the *Sequence* panel (Fig. 5). The selection of sequence generator properties, and the way of its implementation depends on the characteristics of the DBMS selected.

Not all possible combinations of the selected generator options are always valid. By pressing the *Check* button (Fig. 1-5), a user initiates a check of the selected options. If some inconsistencies arise, a user gets the appropriate warnings. Pressing the button *Generate* initiates the generation of SQL scripts and their saving in one or more files. The content of each generated file can be viewed, or modified through the form presented in Fig. 6. By pressing the button *Execute* , a user can also start the execution of a script file on a selected DBMS manually.

Generating SQL scripts may produce various kinds of warnings as a result of potentially incorrect designer's decisions. For example, the following will produce a warning: choosing a *Set Null* action for a constraint comprising a not null attribute, or

giving names that will cause the name of a trigger longer than 30 characters in an Oracle DBMS. A separate panel *Messages* is used to view the warnings.

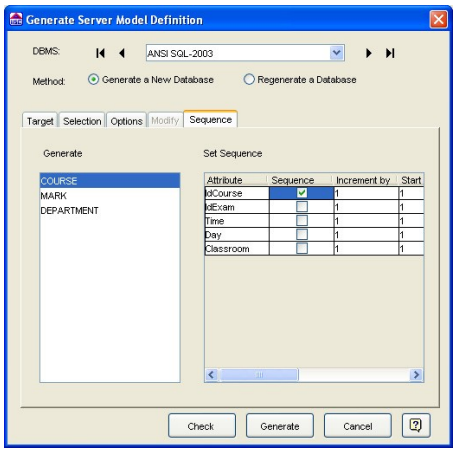


Figure 5. SQL Generator - Sequence panel

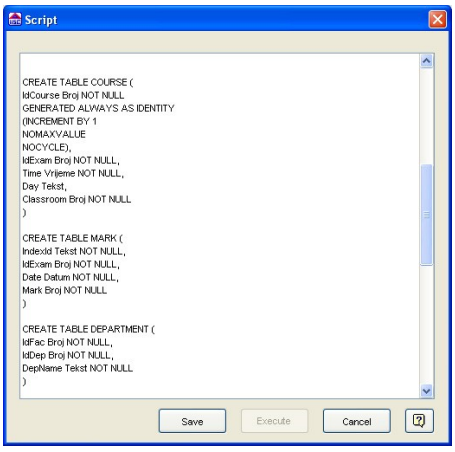


Figure 6. The form for reviewing script

3 Constraint Types Supported

SQL Generator implements constraints of the following types: domain constraints, key constraints, unique constraints, tuple constraints, native and extended referential integrity constraints (default, partial, full), referential integrity constraints inferred from nontrivial inclusion dependencies (default, partial, full), native inverse referential integrity constraints, and inverse referential integrity constraints inferred from nontrivial inclusion dependencies. [1, 3, 4, 8, 9]

According to [4], a designer qualifies each referential integrity constraint in IIS*Case as a default, partial or full and this affects the way of its validation. He or she also selects an action for preserving consistency in the case of an attempt to violate the constraint during inserts, updates, or deletes. The possible actions are: No Action, Cascade, Set Default and Set Null. Before implementing a constraint, SQL Generator analyzes designer's selections. If a selected combination is not applicable, SQL Generator produces a warning.

Constraints are implemented by the declarative DBMS mechanisms, whenever it is possible. However, the expressivity of declarative mechanisms of commercial DBMSs is usually limited in comparison to [4]. Therefore, SQL Generator implements a number of constraints through the procedural mechanisms.

4 An Example of the Procedural Implementation of a Constraint

Common algorithms for controlling a constraint validation are given in [1, 3, 8]. The process of the procedural implementation of a constraint can be unified. It consists of the following steps: (i) specifying a parameterized pattern of the algorithm for a

specific DBMS, (ii) replacing the pattern parameters with real values, and (iii) generating an SQL script comprising necessary triggers, procedures and functions. [1]

In this Section, we present an example of a trigger for the control the deletion of a set of tuples from a relation $r(N_j)$, where N_j is a relation scheme participating in a native, partial referential integrity constraint $N_i[X] \subseteq N_j[Y]$. If a user selects ANSI SQL as a target DBMS, the trigger is not needed, since the partial referential integrity constraint is implemented declaratively, by means of the constraint clause FOREIGN KEY and its subclause MATCH PARTIAL [4]. Suppose a user selects MS SQL Server as a target DBMS. Since MS SQL Server currently does not support the MATCH clause, a trigger is needed. A parameterized generic pattern of such a trigger is shown in Fig. 7. A corresponding pattern for the Oracle Server is presented in [1].

```
CREATE TRIGGER TRG_<N j >_< ConstraintName> _DEL ON <N j > FOR DELETE
AS
 DECLARE    < DeclarationFor _Y>, < DeclarationFor _X>,   < DeclarationFor
_PK_u>
 DECLARE Cursor_<N j >  CURSOR FOR SELECT < AttributeSetFrom _Y> FROM
Deleted
 OPEN Cursor_<N j >
 FETCH NEXT FROM Cursor_<N j > INTO  < VariablesFor _Y>
 WHILE @@FETCH_STATUS=0
 BEGIN
    DECLARE Cursor_<N i > CURSOR FOR
    SELECT< AttributeSetFrom _X>, < AttributeSetFrom _PK_u>  FROM <N i >
    WHERE  < SelectionCriteria _ Cursor_N i >
    OPEN Cursor_<N i >
    FETCH NEXT FROM Cursor_<N i > INTO  < VariablesFor _X>, < Variables
_PK_u>
    WHILE @@FETCH_STATUS=0
    BEGIN
      IF ExistPRI_<Ni> (<VariablesFor_X>)=0
           < Perform_Activity >
       FETCH NEXT FROM Cursor_<N i > INTO  < VariablesFor _X>, < Variables
_PK_u>
    END
    CLOSE Cursor_<N i >
    DEALLOCATE Cursor_<N i >
    FETCH NEXT FROM Cursor_<N j > INTO  < VariablesFor _Y>
 END
 CLOSE Cursor_<N j >
 DEALLOCATE Cursor_<N j >
```

Figure 7. A parameterized generic pattern of the trigger for the control of tuple deletions

The purpose of the trigger is to check if there is a tuple u in r (N_i) that references only a tuple v in r (N_j), which is marked for the deletion. If it is so, a specified action is initiated. Otherwise, v is deleted from r (N_j), regardless of the specified constraint action.

Since the trigger syntax of MS SQL Server does not include the FOR EACH ROW clause, cascaded cursors are used in the parameterized pattern in Fig. 7. In the process of generating a trigger from the pattern, parameter <N_j> is replaced by the relation scheme name N_j and <N_i> is replaced by the name of N_i. Each constraint has its own

name that is embedded into the trigger name by replacing the parameter <ConstraintName>. <DeclarationFor_Y> and <DeclarationFor_X> represent lists of variable declarations of the form @<Attribute_From_Y> *data type*, and @<Attribute_From_X> *data type*, for each attribute in *Y* and *X*, respectively. <DeclarationFor_PK_u> is a list of variable declarations of the form @<Attribute_From_PKey> *data type*, each one for a primary key attribute of N_i. *Deleted* in the statement DECLARE Cursor_<N_j>... is a table with all tuples deleted from $r(N_j)$.

Parameter <VariablesFor_Y> in Fig. 7 is replaced by the list of variables defined by <DeclarationFor_Y>, where each variable is of the form @<Attribute_From_Y>. The variables take values from $v[Y]$, where v is a tuple for deletion. <AttributeSet-From_Y> and <AttributeSetFrom_X> represent the lists of all attributes from *Y* and *X*, respectively. <AttributeSet_PK_u> represent the list of primary key attributes of N_i. <SelectionCriteria_Cursor_N_i> is specified as a sequence of comparison expressions connected by the logical operator AND:

(<Attribute_From _X> IS NULL OR
<Attribute_From _X>=@<Attribute_From _Y>),

where each <Attribute_From _X> or <Attribute_From _Y> belongs to <AttributeSetFrom_X> or <AttributeSetFrom_Y>, respectively.

Parameter <VariablesFor_X> is replaced by the list of variables defined by <DeclarationFor_X>, where each variable is of the form @<Attribute_From_X>. In the same way, <Variables_PK_u> is replaced by the list of variables defined by <DeclarationFor_PK_u>. These variables take their values from a tuple *u*. Depending on the constraint action selected by the user, parameter <Perform_Activity> is replaced by one of the following procedures: NoAction_<N_j>, SetNullPRI_<N_j>, SetDefaultPRI_<N_j> and CascadeDelPRI_<N_j>. Current primary key values of <Varibales_PK_u> are passed to all of the procedures, except to the first one.

A parameterized pattern of the function ExistPRI_<N_i> is shown in Fig. 8. For each primary key value of a tuple *u*, <SelectionCriteria> is specified as a sequence of comparison expressions connected by the logical operator AND:

(@<Attribute_From _X> IS NULL OR
v.<Attribute_From _Y> = @<Attribute_From _X>).

```
CREATE FUNCTION ExistPRI_<N i > (< DeclarationFor _X>)
RETURNS int
AS
BEGIN
  DECLARE @Count int,  @Ret  int
  SELECT @Count = COUNT(*) FROM <N j > v WHERE <SelectionCriteria>
  IF @Count != 0 SELECT @ret=1
  ELSE SELECT @ret=0
 RETURN @ret
END
```

Figure 8. A parameterized pattern of function ExistPRI_<N_i>

Procedure NoAction_<N_j> is presented in Fig. 9. It is used to implement the constraint action No Action. Procedure SetNullPRI_<N_j> is presented in Fig. 10. It is used to implement the constraint action Set Null. <Attribute_value> is a sequence of comma separated expressions, one for each attribute from *X*, specified as follows:

u.<Attribute_From _X> = NULL.

<SelectionCriteria> is a sequence of expressions connected by AND, as follows:

u.<Attribute_From _PK> = @<Attribute_From _PK>.

```
CREATE PROCEDURE NoAction_<N j >
AS
  RAISERROR('Tuple cannot be deleted from the specified relation ', 16, 1)
  ROLLBACK TRAN
```

Figure 9. A parameterized pattern of the No Action

```
CREATE PROCEDURE SetNullPRI_<N j >  (< DeclarationFor _PK_u>)
AS
  UPDATE u SET <Attribute_value> FROM <N i > u WHERE <SelectionCriteria>
```

Figure 10. A parameterized pattern of the Set Null action

Procedure SetDefaultPRI_<N_j> used to implement Set Default action is presented in Fig. 11. <ValueAssignmentFrom_X> is replaced as follows:

@<Attribute_From _X>= *u*.<Attribute_From _X>.

```
CREATE PROCEDURE SetDefaultPRI_<N j > (< DeclarationFor _PK_u>)
AS
  DECLARE < DeclarationFor _X>, @AttributesForUpd VARCHAR(255)
  SET @ AttributesForUpd  = ' '
  SELECT <ValueAssignmentFrom_X> FROM <N i > u WHERE <SelectionCriteria>
  IF (<UpdateCondition>)
  BEGIN
   IF (@<Attribute_From _X> IS NOT NULL)
   BEGIN
    IF @AttributesForUpd != ''
      SET  @AttributesForUpd=@AttributesForUpd+',
      u.<attribute_from_X> = default'
    ELSE
      SET  @AttributesForUpd = 'u.<attribute_from_X> = default'
   END
   EXEC ('UPDATE u SET ' + @AttributesForUpd + 'FROM <N i >  u WHERE
       u.<Attribute_From_PK>=' +@<Attribute_From_PK>)
   SELECT <ValueAssignmentFrom_X> FROM <N i > u WHERE
      <SelectionCriteria>
   IF dbo.ExistPRI_<N i >(< VariablesFor _X>)=0
   BEGIN
     RAISERROR('Tuple cannot be deleted from the specified relation ', 16, 1)
     ROLLBACK TRAN
   END
  END
```

Figure 11. A parameterized pattern of the Set Default action

<SelectionCriteria> is a sequence of expressions connected by AND, one for each attribute from <Attribute_From_PK>, specified as follows:

(*u*.<Attribute_From _PK> = @<Attribute_From _PK>).

Since only the attributes having non null values are set to the default values, the first IF statement in Fig. 11 checks if there is at least one having a non null value. <UpdateCondition> is a sequence of the expressions connected by OR, one for each attribute in *X*, specified as follows:

@<Attribute_From _X> IS NOT NULL.

The bolded code in Fig. 11 is repeatedly generated, once for each attribute in *X*. Therefore, for each attribute in *X* having a non null value, a string

'*u*.<Attribute_From_X> = default'

is concatenated to the current value of the variable @AttributesForUpd.

The WHERE clause of the UPDATE command in the string used in EXEC command is correctly defined in Fig. 11, if the primary key of N_i consists of the only one attribute. Otherwise, the clause is transformed to include an expression of the form

u.<Attribute_From_PK> = @<Attribute_From_PK>,

for each primary key attribute, and all such expressions are connected by the AND operator.

Procedure CascadeDelPRI_<N_j> used to implement Cascade action is presented in Fig. 12. <SelectionCriteria> is an expression of the form <SelectionCriteria1> AND <SelectionCriteria2>. <SelectionCriteria1> is a sequence of expressions connected by AND, one for each attribute in <Attribute_From_PK> specified as:

(u.<Attribute_From_PK> = @<Attribute_From_PK>).

<SelectionCriteria2> is a sequence of expressions connected by AND, one for each attribute in *X*, specified as:

(u.<Attribute_From_X> IS NOT NULL).

```
CREATE PROCEDURE CascadeDelPRI_<N j > (< DeclarationFor _PK_u>)
AS
   DELETE FROM <N i > u  WHERE <SelectionCriteria>
```

Figure 12. Parameterized pattern of procedure for cascade delete

In this example, we have presented one of the cases met in the real life. We have deliberately selected here the partial referential integrity constraint, since its implementation is the most complex. Also, we have deliberately selected MS SQL Server, because it makes the implementation even more complex, since its trigger syntax does not support FOR EACH ROW clause.

5 Conclusion

The paper describes SQL Generator that is a component of IIS*Case. IIS*Case is a complex software tool that supports automatic generation of 3NF db schemas and software applications. In the framework of IIS*Case, SQL Generator provides users with such an intelligent support that they can generate implementation specifications of db schemas even without knowing the SQL syntax and procedural DBMS mechanisms for the implementation of constraints.

An advantage of SQL Generator over other similar products is that users have a wider selection of possible actions to preserve db consistency. Besides the generation of common db constraints, like key, unique, not null, and native referential integrity, SQL Generator also enables the implementation of the default, partial and full referential integrity constraints, and the selection of an appropriate action from the set {No Action, Cascade, Set Default, Set Null}. Also, SQL Generator provides the implementation of the inverse referential integrity constraints. SQL Generator validates selections of input parameter values, analyzes designer's solutions, and issues warnings if it detects any inconsistency.

Further research and development are focused on extending the functionality of SQL Generator. We plan to:

- implement the generation of the extended referential integrity [9],
- provide the compatibility checking of data types,
- add modules for the design and implementation of physical data structures for particular DBMSs,
- implement in IIS*Case visual editors for specifying user defined functions and tuple (check) constraints [8], and
- enable generating SQL scripts for a wider selection of DBMSs.

Acknowledgment. The publishing was supported by Provincial Departments of Education and Culture, and Science and Tech. Development, of Autonomous Province of Vojvodina.

References

1. Aleksić S., *An SQL Generator of Database Schema Implementation Specification in a CASE Toll IIS*Case,* M. Sc. (Mr.) Dissertation, University of Novi Sad, Faculty of Technical Sciences, Novi Sad, Serbia, 2006.
2. ARTech. *DeKlaritTM (The Model-Driven Tool for Microsoft Visual Studio 2005),* Chicago, U.S.A. Available at: `http://www.deklarit.com` [June, 2007].
3. Govedarica M., *Design the Set of Implementational Database Schema Constraints,* M.Sc. (Mr.) Dissertation, University of Novi Sad, Faculty of Technical Sciences, Novi Sad, 1998.
4. ISO/IEC 9075-{1, 2, 11}:2003 (ANSI SQL:2003), American National Standards Institute;
5. Luković I, Mogin P, Pavićević J, Ristic S, *An Approach to Developing Complex Database Schemas Using Form Types,* Software: Practice and Experience, John Wiley & Sons Inc, Hoboken, USA, ISSN: 0038-0644, Published Online, May 29, 2007, DOI: 10.1002/spe.820.
6. *Microsoft SQL Server 2000,* User Manuals.
7. *Microsoft SQL Server 2005,* User Manuals.
8. Mogin P, Luković I, Govedarica M, *Database Design Principles,* 2nd Edition, University of Novi Sad, Faculty of Technical Sciences, Novi Sad, Serbia, 2004, ISBN: 86-80249-81-5.
9. Mogin P, Luković I, Govedarica M, *Extended Referential Integrity,* Novi Sad Journal of Mathematics, Novi Sad, Serbia, ISSN: 1450-5444, Vol. 30, No. 3, 2000, pp. 111-122.
10. *Oracle DBMS 9i,* User Manuals.
11. *Oracle DBMS 10g,* User Manuals.
12. *Oracle Designer 9i,* On-line Documentation.
13. Pavićević J, *Development of A CASE Tool for Automated Design and Integration of Database Schemas,* M. Sc. (Mr.) Dissertation, University of Montenegro, Faculty of Science, Podgorica, Montenegro, 2005.
14. Pavićević J, Luković I, Mogin P, Govedarica M, *Information System Design and Prototyping Using Form Types,* INSTICC I International Conference on Software and Data Technologies, Setubal, Portugal, September 11-14, 2006, Proceedings, Vol. 2, pp. 157-160.
15. Sybase PowerDesigner 10, On-line Documentation.

Proceedings of the International Multiconference on
Computer Science and Information Technology, pp. 1123 – 1132
ISSN 1896-7094
© 2007 PIPS

A QTI Metamodel

Sonja Radenković, Nenad Krdžavac, Vladan Devedžić

FON - School of Business Administration,
Department of Information Systems and Technologies,
University of Belgrade, Serbia
sonjam@fon.bg.ac.yu, devedzic@etf.bg.ac.yu,
nenadkr@galeb.etf.bg.ac.yu

Abstract. Question & Test Interoperability specification (QTI) provides a good starting point for modeling and designing assessment systems. This paper presents a metamodel for developing a QTI-based assessment system using the Model Driven Architecture (MDA) standard. The main reason for applying the MDA standard in development of assessment systems is to make a clear difference between conceptual and concrete modeling in order to automate transfer and sharing of information and knowledge. The first prerequisite for this is a QTI metamodel.

Keywords: Assessment, Metamodeling, Model Driven Architecture

1. Introduction

In order to create a flexible, interoperable assessment system that can be easy to maintain and reuse, the Model Driven Architecture (MDA) standard [7] that comes from software engineering can be applied. By applying this standard, it is possible to utilize the possibilities of traditional assessment, and at the same time to combine additional demands of new types of assessment with limited resources. The core concept here is to change the system's specification rather than implementation using the Unified Modeling Language [9] as the standardized modeling language that most tools provide support for.

This paper proposes a metamodel for development of assessment systems based on the IMS Question & Test Interoperability specification [4] (or just QTI systems, for short) using the MDA standard.

2. IMS QTI Standard – A Short Overview

The QTI standard [4] specifies how to represent question (assessmentItem) and test (assessmentTest) data and the corresponding result reports. These items are the smallest exchangeable assessment object within this specification [5]. An item is more than a 'question' in that it contains the question and instructions of how to be presented, the response processing to be applied to the candidate response(s), and the

feedback that may be presented (including hints and solutions). There are different forms of items, such as simple- or multiple-choice questions or fill-in-the-blank tasks [6]. The presentation provides the structure for defining several possibilities for the same question, in order to be able to present the same question in different ways. Each answer within this question can also have different wordings. The response processing allows the author of a test to predefine how the answers to the test will be evaluated. The results of a test can be expressed using the "result reporting". This definition describes that the results of a test can be recorded so that other systems can make use of it. The feedback component of QTI consists of two types, modal and integrated [5]. There is an exchange of items, assessment, and results between authoring tools, item banks, learning systems and assessment delivery systems. For interchange between these systems, an XMI binding (see Section 5 for explanation) is provided [10].

3. Related Work

There were efforts to develop an assessment system [8] that will conform to the IMS QTI standard in order to support the reuse of both new and traditional assessment types. Some of them tried to deploy a specific methodology in order to improve reusability of assessment systems [2]. They developed an extensible educational model of assessment to provide a broader basis for interoperability specifications for the assessment process from construction to evaluation. The model was cast in the Unified Modeling Language (UML) [9], a standard language for specifying, visualizing, constructing and documenting concepts and artifacts.

This model, however, cannot provide interoperability of assessment systems at all. This model can be used to create Java classes and method bodies, for example, using Java UML profile [15], and then to transform this UML model into Java code, and to complete the Java program using a Java IDE [16]. On the other hand, it cannot be used to create heterogeneous assessment system that interoperates at the level of standard component interfaces.

4. Problem Statement

To create a flexible assessment system capable of presenting and analyzing the students' solutions, it is necessary to define the system requirements precisely. These requirements are:

- The system has to be designed according to standards of modern software engineering, such as the MDA standard.
- It has to provide a well-documented content format for storing items, in a way independent of the authoring tool used to create them;
- It has to support the deployment of item banks across a wide range of learning and assessment delivery systems;
- It has to support the deployment of items and item banks from diverse sources in a single learning or assessment-delivery system;

- It has to provide other systems with the ability to report test results in an intelligent manner; to this end, a good starting point can be the use of description logics (DL) reasoning techniques, such as concept classification and consistency [14].

By creating a system to accomplish the above requirements, one obtains:

- a flexible and quickly developed assessment environment;
- high-level of interoperability between various component systems;
- easy and intuitive testing of students' knowledge, done in an intelligent way;
- an easy-to-extend system that can be improved by including new subsystems.

5. A QTI Metamodel

In order to provide the interoperability in both homogeneous and heterogeneous QTI assessment systems, it is useful to apply the MDA standard in QTI assessment system development. The reasons are as follows:

- the interoperability in heterogeneous environments could be achieved via shared metadata of subsystems that create the assessment system.
- the overall strategy for sharing and understanding metadata consists of the automated development, publishing, management and interpretation of models [10].

In other words, the metamodel determines how expressive its models can be, because the models are defined using the concepts defined in the metamodel.

The first step in developing a QTI-based assessment system using the MDA standard is to create a metamodel that captures the main concepts of QTI, i.e. those for creating the models of Item Bank Tool, Test Construction Tool, as well as Learning System Tool. The novelty in creating QTI-based assessment systems here is the improvement of response processing. As mentioned above, the QTI standard specifies that the response processing allows the author of the test to predefine how the test answers will be evaluated. Specific rules can be defined, which allow the exact definition of when an answer to a question is wrong and when it is correct. Furthermore, scoring weights can be defined that allow for a more complex evaluation of a test. We are going a step further; we propose applying DL reasoning techniques in response processing of a QTI-based assessment system in order to check semantic consistency of answers to open-ended questions. Because of that, it is necessary to use a DL metamodel as a part of QTI-based assessment system metamodel.

The section below explains the QTI metamodel. The metamodel is designed using the Posedion UML tool (www.gentleware.com), and according to the IMS QTI standard [6]. The metamodel is a UML-based QTI metamodel. The metamodel is divided in four meta-packages, according to the QTI standard: *Item Bank Tool* , *Learning System Tool* , *Test Construction Tool* and *Description Logics Reasoning Tool* . Due to space limitations, only two of them are discussed in this paper.

5.1 Item Bank Tool Metapackage

According to the QTI standard [4], the following metaclasses may be defined: *ResponseDeclaration*, *TemplateDeclaration*, and *OutcomeDeclaration* (Figure 1). These metaclasses are derived from the *VariableDeclaration* metaclass [5]. The attribute called *Identifier* is the same for the above *declaration* metaclasses .

The core metaclass of the Item Bank Tool metapackage is *AssessmentItem* (Figure 1). The other metaclasses are connected to this metaclass. *Identification* of any assessmentItem model, at the `M1 level is defined using the *identifier* attribute in the metaclass.

According to [5], the *ItemBody* metaclass represents the text, graphics, media objects, and interactions that describe the item's content and information about how it is structured. The body is presented by combining it with stylesheet information, either explicitly or implicitly using the default style rules of the delivery or the authoring system [5], (Figure 2). The *ItemBody* metaclass is linked to the *FeedbackBlock* metaclass (Figure 2). The *FeedbackBlock* metaclass is very important in case of presenting any material to the students.

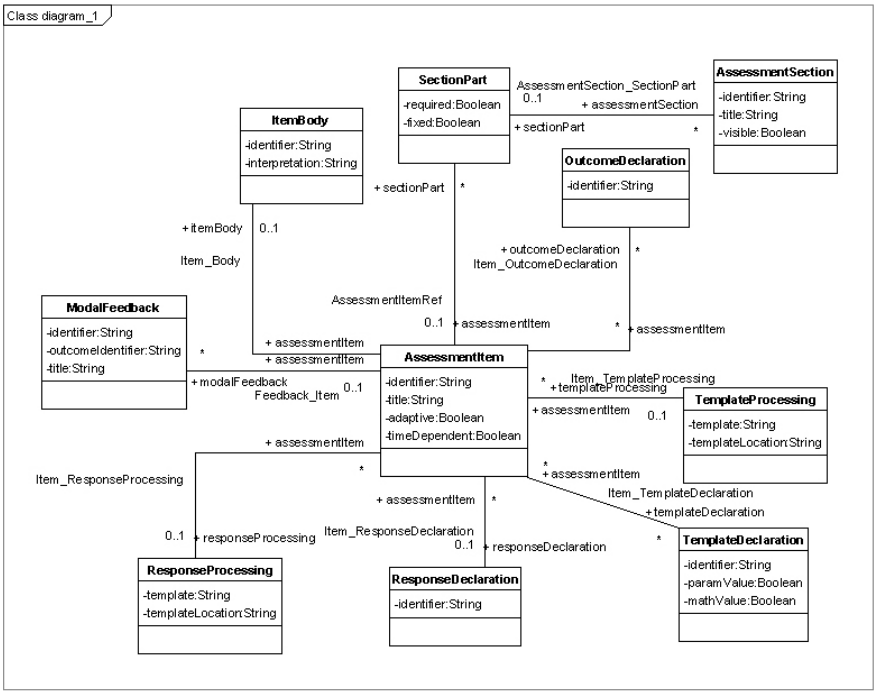


Fig. 1. Item Bank Tool's metaclasses

The metaclasses that are in charge of processing are *OutcomeProcessing*, *ResponseProcessing* and *TemplateProcessing* (Figure 3, Figure 4, Figure 5, and Figure 6). Response processing consists of a sequence of rules that are carried out, in a predefined order, by the response processor [5]. The *ResponseProcessing* metaclass has two attributes: *template* and *templateLocation* (Figure 4, Figure 5) [5].

ResponseProcessing depends on *ResponseConditions* and *ExitResponse* (Figure 4).

Response processing involves the application of a set of *responseRules* including the testing of *responseConditions* and the evaluation of expressions involving the item variables. The *ResponseProcessing* metaclass (Figure 4) represents this involvement. Response processing is the process by which the Delivery Engine assigns outcomes based on the candidate's responses [5]. The OutcomeProcessing metaclass is linked to *OutcomeRules* (Figure 3). The metaclasses *OutcomeCondition* and *ExitTest* extend *OutcomeRule*.

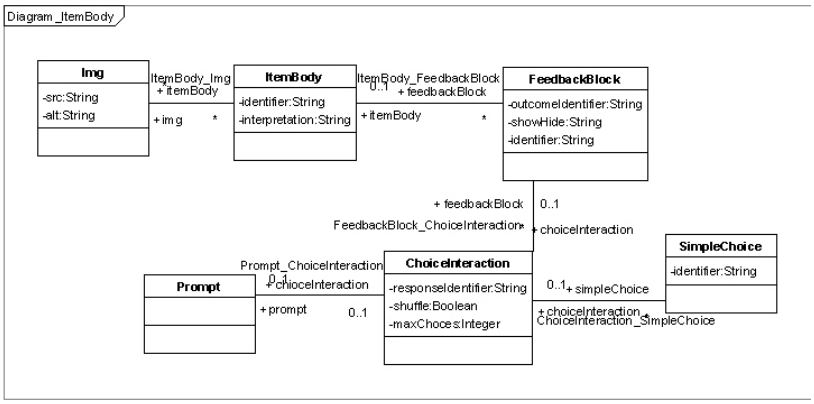


Fig. 2. The ItemBody metaclass in the QTI Metamodel

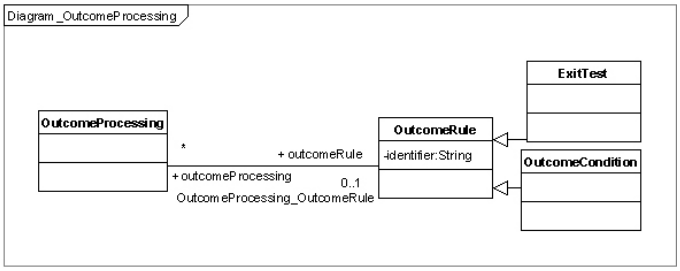


Fig. 3. The OutcomeProcessing metaclass in the QTI metamodel

The *feedback* that the QTI system provides to the student (candidate) conditionally includes the material presented. This is based on the result of *responseProcessing* . In other words, the feedback is controlled by the values of outcome variables [5] In the QTI metamodel, feedback is represented with the *FeedBackBlock* metaclass (Figure 2). There is, also, *integrated feedback* , which is embedded in the *itemBody.*

Template processing consists of one or more *templateRules* (Figure 5) that are followed by the cloning engine or delivery system in order to assign values to the template variables [5]. Template rules are described by the *TemplateRule* metaclass. The *SetTemplateValue* and *SetCorrectResponse* metaclasses extend the *TemplateRule* metaclass.

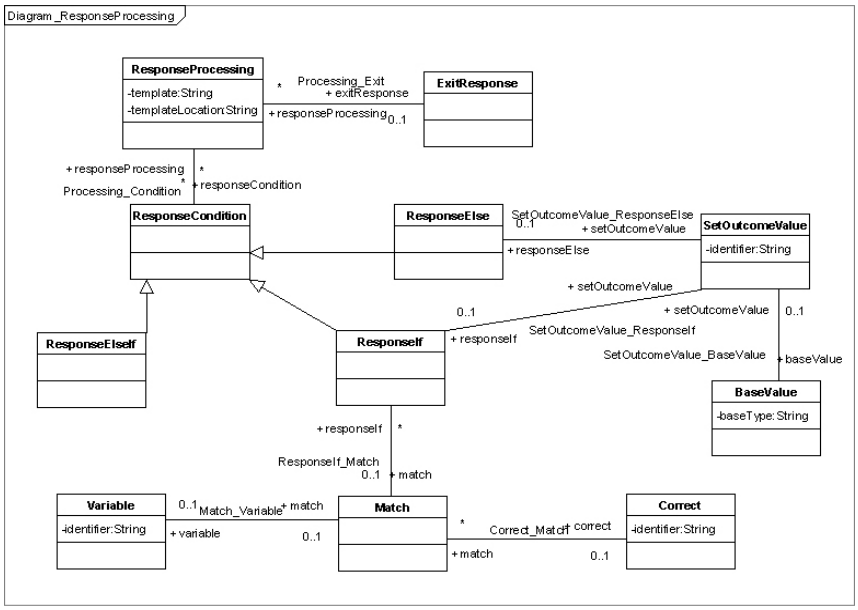


Fig. 4. The ResponseProcessing metaclass in the QTI metamodel

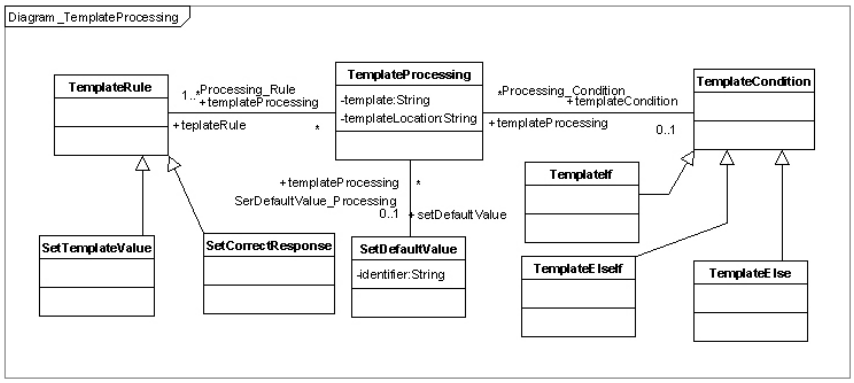


Fig. 5. The TemplateProcessing metaclass in the QTI metamodel

5.2 Test Construction Tool Metapackage

The central metaclass in the Test Construction Tool metapackage is *AssessmentTest*. This metaclass aggregates the *TestPart* metaclass and the associated *ScoreReport* and *TimeLimits* metaclasses (Figure 6).

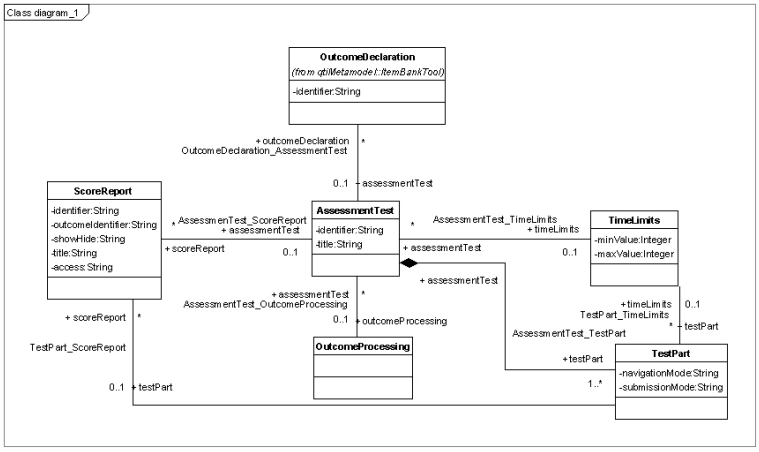


Fig. 6. Metaclasses in the Test Construction Tool Metapackage

6. QTI System Development Using the MDA Standard

The essence of QTI system development using the MDA standard is the transformation from Platform Independent Model (PIM) to Platform Specific Model (PSM) [7]. PIM specifies the system's functionality [10]. PSM is the specification of implementation details in a concrete technical environment. In the case of QTI system, the PIM is defined as a QTI metamodel and the models of different subsystems. The PSM can be generated using: Java Metadata Interfaces (JMI) [1], Ecore classes of the Eclipse Modeling Framework [12], and the QVT standard.

The transformation from PIM to PSM can be made in the Atlas Transformation Language (ATL) [3] that is used for model transformation. In case of a QTI system, it means that the assessment items will be transferred from Learning model to the corresponding Item Bank model (Figure 7), and then to the corresponding Test Construction Model in the process of evaluating a student's answer. The DL reasoning machine will check the consistency of the answer [14].

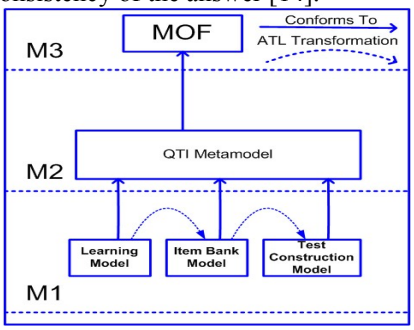


Fig. 7. The process of analysis the students' solutions in MDA

7. Discussion

In order to present the transformation from the QTI metamodel to the corresponding QTI model we used a simple choice example from the QTI standard, Figure 8. The part of the QTI metamodel that is relevant for this example is shown in Figure 9. This part of the QTI metamodel is transformed into the equivalent Ecore-based metamodel (EMF repository) using Eclipse plug-in. The part of QTI metamodel that refers to the simple choice item is shown in Figure 10. The result of this transformation is model given in Figure 11.



Fig. 8. The example of simple choice item in QTI metamodel [17]

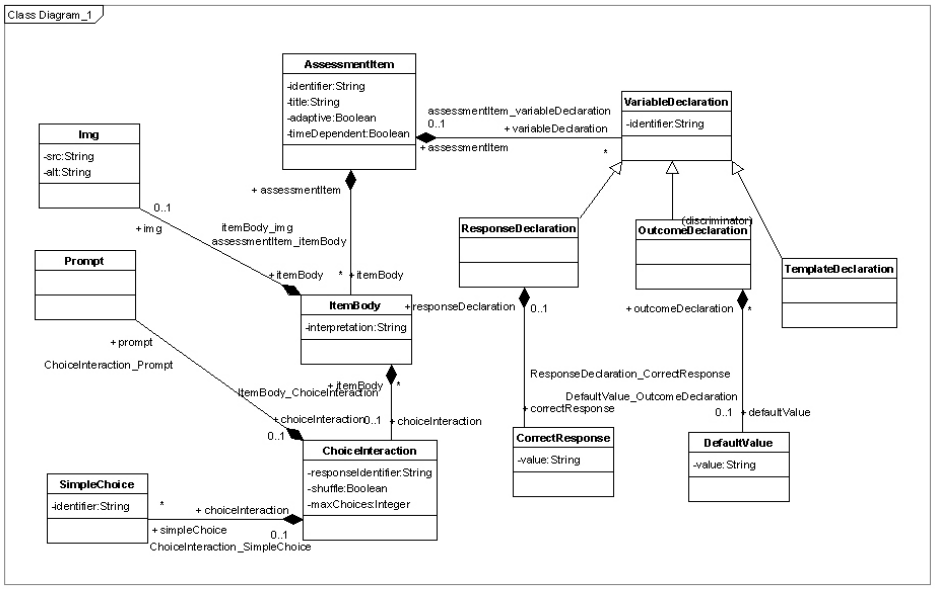


Fig. 9. The QTI metamodel for the simple choice item

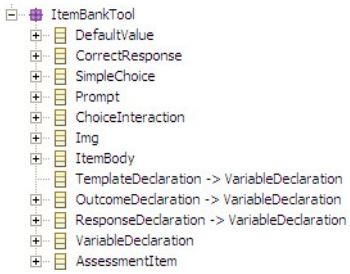


Fig. 10. The QTI metamodel for simple choice item

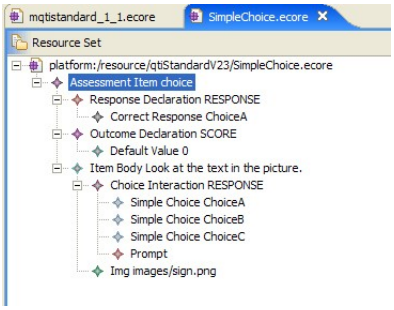


Fig. 11. The QTI model for simple choice item

Using the model above, it is possible to use Java in creating, updating and accessing instances of this model.

The QTI metamodel is created in order to help assessment system developers to develop the QTI systems using Model Driven Engineering (MDE). The metamodel itself was cast in terms of UML, which means that it is understandable and useable for experts in assessment domain with knowledge of UML. This metamodel is, also, machine-readable because it is transformed from an MOF-based QTI metamodel into an ECore-based QTI metamodel by using Eclipse plug-ins. This metamodel could be extended for assessment systems that are not QTI-based.

8. Conclusions and Future Work

A QTI metamodel can help all developers of assessment systems who want to apply the MDA standard in creating a flexible, robust and, interoperable assessment system. Once they have a QTI metamodel, they need to create transformations, for example in ATL, to transform their PIMs to PSMs. By making a clear difference between conceptual and concrete modeling, developers can successfully select mechanisms to automate transfer and sharing of information and knowledge. It allows development to move smoothly between the levels of abstraction, with much of the work being automated. Each time a new type of assessment system appears, it means that it is necessary to change, or to enrich the QTI metamodel rather than the implementation.

At the same time, it is possible to improve a specified subsystem individually, without changing the other subsystems.

Future work will be focused on the implementation of transformations from TestConstructionModel to OWL model. The target model will use a description logic reasoner for checking the satisfiability of students' answers.

References

1. R. Dirckze, (spec. leader): *Java Metadata Interface (JMI) API Specification ver. 1.0,* 2002. http://jcp.org/aboutJava/communityprocess/final/jsr040/
2. Desirée Joosten – ten Brinke*, Jan van Bruggen, Henry Hermans, Ignace Latour, Jan Burgers, Bas Giesbers & Rob Koper *Modeling Assessment for Re-use of Traditional and New Types of Assessment* [Online]. Available: http://dspace.ou.nl/bitstream/1820/355/4/modelling-assessment.pdf
3. F. Jouault, I. Kurtev, *Transforming Models with ATL,* In Proceedings of the Model Transformations in Practice Workshop at MoDELS, Montego Bay, Jamaica, 2005. Available: http://sosym.dcs.kcl.ac.uk/events/mtip/submissions/jouault_kurtev__transforming_models_with_atl.pdf
4. *IMS Question and Test Interoperability Overview,* Version 2.1 Public Draft Specification, IMS Global Learning Consortium, Inc. [Online]. Available: http://www.imsglobal.org/question/qti_v2p1pd/imsqti_oviewv2p1pd.htm
5. *IMS Question and Test Interoperability Information Model,* Version 2.1 Public Draft Specification, IMS Global Learning Consortium, Inc, February 2006, [Online]. Available: http://www.imsglobal.org/question/qti_v2p1pd/imsqti_infov2p1pd.html
6. *ELENA Creating a Smart Space for LearningTM,* Available: www.elena-project.org
7. *MDA Guide Version 1.0.1,* Available: www.omg.org/docs/omg/03-06-01.pdf
8. Petros Lalos, Symeon Retalis, Yiannis Psaromiligkos, *Creating personalised quizzes both to the learner and to the access device characteristics: the Case of CosyQTI,* 3rd International Workshop on Authoring of Adaptive and Adaptable Educational Hypermedia (A3EH), 2005, Amsterdam.
9. *UML 2.0 Infrastructure Overview,* Available: http://www.omg.org/issues/UMLSpec.pdf
10. *OMG XMI Specification, ver. 1.2,* OMG Document Formal/02-01-01, 2002. Available: http://www.omg.org/cgi-bin/doc?formal/2002-01-01.pdf
11. Biggs J. B. (1999). *Teaching for Quality Learning at University,* Buckingham: Society for Research in Higher Education & Open University Press.
12. *Discover the Eclipse Modeling Framework (EMF) and Its Dynamic Capabilities,* Available: http://www.devx.com/Java/Article/29093
13. Dochy F., Segers M., & Sluijsmans D. M. A. (1999). *The use of self-, peer-, and co-assessment in higher education: a review,* Studies in Higher Education, 24, 331-350.
14. F. Baader, U. Sattler: *An Overview of Tableau Algorithms for Description Logics,* Studia Logica, 69(1), (2001), 5–40.
15. *Catalog of UML Profile,* Available: http://www.omg.org/technology/documents/profile_catalog.htm
16. *Eclipse J2EE IDE,* Available: www.myeclipseide.org
17. *IMS Question and Test Interoperability Examples Version 2.1 Public Draft Specification,* IMS Global Learning Consortium, Inc, February 2006, [Online]. Available: http://www.imsglobal.org/question/qti_v2p1pd/examples/items/choice.xml

Proceedings of the International Multiconference on
Computer Science and Information Technology pp. 1133–1146
ISSN 1896-7094
© 2007 PIPS

An L-Attributed Grammar for Adjoint Code

Uwe Naumann[12] and Jan Riehme[2]

[1] LuFG Informatik 12, Department of Computer Science, RWTH Aachen University
52056 Aachen, Germany
[2] Department of Computer Science, University of Hertfordshire
Hatfield, AL10 9AB, United Kingdom

[naumann|riehme]@stce.rwth-aachen.de,
Website: http://www.stce.rwth-aachen.de/

Abstract. Gradients of high-dimensional functions can be computed efficiently and with machine accuracy by so-called adjoint codes. We present an L-attributed grammar for the single-pass generation of intraprocedural adjoint code for a simple imperative language (a subset of C). Our ideas can easily be applied to any programming language that is suitable for syntax-directed translation. Moreover the conceptual insights are useful in the context of multi-pass generation of adjoint code. Our focus is on correctness. The necessary domain-specific code optimizations are beyond the scope of this paper. We give references to corresponding work in this area.

1 Motivation

Numerical simulation plays a central role in computational science and engineering. Derivatives (gradients, Jacobians, Hessians or even higher derivatives) are required in order to make the highly desirable transition from pure simulation to optimization of the numerical model or its parameters. Refer to [2–5] for an impressive collection of such applications.

Consider an implementation of a multivariate nonlinear function $f : I\!R^n \to I\!R$ as a computer program where $y = f(\mathbf{x})$. Suppose that we are interested in the sensitivities of the objective y with respect to changes in the parameter vector $\mathbf{x}$, for example, in the context of an unconstrained optimization algorithm. Such derivatives (the gradient of y with respect to $\mathbf{x}$) can be approximated by centered (or forward, or backward) finite difference quotients

$$\frac{\partial y}{\partial x_i} \approx \frac{f(x_0,\ldots,x_i+h,\ldots,x_{n-1}) - f(x_0,\ldots,x_i-h,\ldots,x_{n-1},u)}{2h} \tag{1}$$

for an appropriate (small) value $h \in I\!R$. Choosing the right value for h for a given function evaluated in a given floating-point number system can be problematic. Cancellations can lead to very poor approximations of the derivatives. More importantly, the accumulation of the whole gradient requires $2n$ function evaluations which may be infeasible for high-dimensional problems. See [11] for an application in oceanography where n can be of the order of 10^{12} and higher.

Even for very simple representatives of Equation (1) (for example, $y = x_0 * \ldots * x_{n-1}$) the finite difference approximation of the gradient can take several hours for $n \geq 10^6$. In this paper we present an L-attributed grammar [10] for transforming the implementation of f into an adjoint code during a single pass compilation process. The adjoint code computes the same gradient in only a few seconds.

In Section 2 we outline the basic structure of adjoint codes. An L-attributed grammar for transforming programs written in imperative programming languages that are suitable for single-pass compilation ([1, 13]) is presented in Section 3. A simple proof-of-concept implementation based on `flex` and `bison` as well as a case study are discussed in Section 4. We conclude with an outlook to potential areas of application of the proposed technology in Section 5.

2 Adjoint Code

The problem of determining an appropriate value for h can be eliminated by considering a tangent-linear model $\dot{F}$ of F. Let therefore $\mathbf{x} = \mathbf{x}(t)$ with $t \in I\!R$ and set

$$\frac{\partial \mathbf{x}}{\partial t} = \dot{\mathbf{x}}.$$

By the chain rule we get

$$\frac{\partial \mathbf{y}}{\partial t} = \dot{\mathbf{y}} = \dot{F}(\mathbf{x}, \dot{\mathbf{x}}) = F' \cdot \dot{\mathbf{x}}. \tag{2}$$

Refer to Figure 1 (a) and (b) for a graphical illustration. It can be regarded as a transformation of the parser tree of $F(\mathbf{x}(t))$ (see (a)) into one for $\dot{F}(\mathbf{x}, \dot{\mathbf{x}})$. The technique is known as the forward mode of automatic differentiation (AD) [7]. The parse tree is linearized by attaching partial derivatives to the corresponding edges. The chain rule of differentiation is interpreted as the chained product of all edge labels along the path from t to $\mathbf{y}$. Assuming that we have an implementation of $\dot{F}$ we can compute the columns of F' by letting $\dot{\mathbf{x}}$ range over the Cartesian basis vectors in $I\!R^n$. The computational complexity of this approach is of the same order as that of finite differences.

To eliminate the dependence of the computational complexity on the potentially very large value of n we consider adjoint codes that can be generated by the reverse mode of AD. Let therefore $t = t(\mathbf{y})$ with $t \in I\!R$ and set

$$\frac{\partial t}{\partial \mathbf{y}} = \bar{\mathbf{y}}.$$

Exploiting the associativity of the chain rule we get

$$\bar{\mathbf{x}} = \bar{F}(\mathbf{x}, \bar{\mathbf{y}}) = \bar{\mathbf{y}} \cdot F'. \tag{3}$$

Refer to Figure 1 (c) and (d) for illustration. All barred vectors ($\bar{\mathbf{x}}$ and $\bar{\mathbf{y}}$) are row vectors. Assuming that we have an implementation of $\bar{F}$ we can compute the rows of F' by letting $\bar{\mathbf{y}}$ range over the Cartesian basis vectors in $I\!R^m$. Gradients of scalar

functions in particular can be obtained at a (hopefully) small constant multiple of the computational complexity of F. The realization of this theoretical result in practice is the subject of numerous ongoing research and development efforts world-wide. See `http://www.autodiff.org` for links and further information.

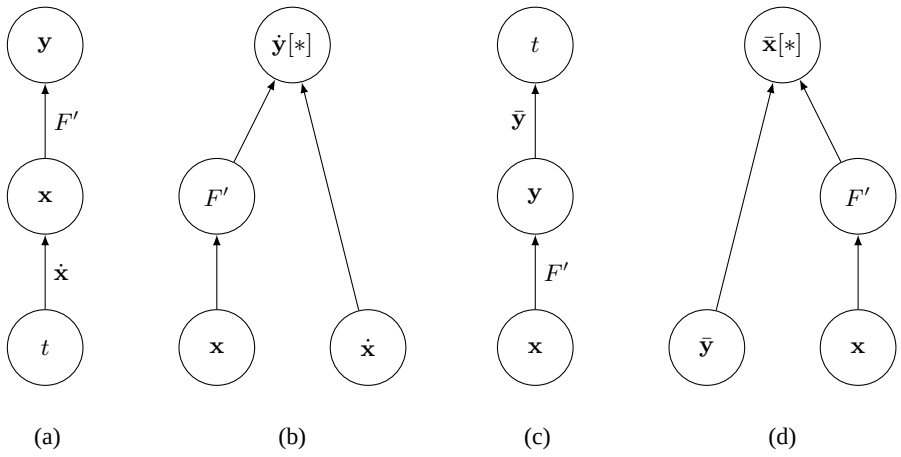


Fig. 1. Linearized $F(\mathbf{x}(t))$ (a), $\dot{F}$ (b) linearized $t(F(\mathbf{x}))$ (c), $\bar{F}$ (d)

Any execution of the program F is expected to decompose into a sequence of elemental assignments

$$v_j = \varphi_j(v_i)_{i \prec j} \tag{4}$$

for $j = 1, \ldots, p+m$ and $i \prec j$ if and only if v_i is an argument of φ_j. Equation (4) is also referred to as the *code list* of F at the given point that fixes the flow of control. We set $v_{i-n} = x_i$ for $i = 1, \ldots, n$ and $v_{p+j} = y_j$ for $j = 1, \ldots, m$. The v_k, $k = 1-n, \ldots, p+m$, are called *code list variables*.

The *elemental functions* φ_j are assumed to be continuously differentiable in a neighborhood of the current argument. The corresponding local partial derivatives are denoted by

$$c_{j,i} = \frac{\partial \varphi_j}{\partial v_i}.$$

Adjoints are propagated backwards with respect to the data flow in the code list. Hence, the values of the intermediate variables are not used in their original order of computation. In (incremental) reverse mode AD the local partial derivatives are computed during the adjoint evaluation.

$$v_{i-n} = x_i \qquad \text{for } i = 1, \ldots, n \tag{5}$$

$$v_j = \varphi_j(v_i)_{i \prec j} \qquad \text{for } j = 1, \ldots, p+m \tag{6}$$

$$y_k = v_{p+k} \qquad \text{for } k = 1, \ldots, m \tag{7}$$

$$\bar{v}_{p+k} = \bar{y}_k \qquad \text{for } k = 1, \ldots, m \tag{8}$$

$$\bar{v}_j = 0 \qquad \text{for } j = 1 - n, \ldots, p \tag{9}$$

$$c_{j,i} = \frac{\partial \varphi_j}{\partial v_i};\ \bar{v}_i = \bar{v}_i + c_{j,i} \cdot \bar{v}_j \qquad \text{for } i \prec j \text{ and } j = q, \ldots, 1 \tag{10}$$

$$\bar{x}_i = \bar{v}_{i-n} \qquad \text{for } i = 1, \ldots, n. \tag{11}$$

Adjoint assignments are generated for all assignments in the original code. We build assignment-level code lists as in Equations 5–7. The data-flow reversal requires arguments of nonlinear operations to be persistent. Conservatively we account for this by storing all overwritten values on a value stack (`push_v`). The resulting code is referred to as *augmented forward code*. Adjoints are propagated backwards according to Equations 8–11. The previously stored values are restored from the stack (`pop_v`). The resulting code is referred to as *backward code*. The compiler-generated intermediate variables v_j represent subexpressions of the right-hand side. Hence, they are read exactly once, thus eliminating the need for the initialization in Equation (9) as well as that for the incrementation in Equation (10). Adjoints of compiler-generated intermediate variables are simply overwritten. Only adjoints of program variables need to be initialized and incremented.

Example In our proof-of-concept implementation (see Section 4) the single assignment

```
x=x*y
```

is transformed into the augmented forward code

```
push_v(v0); v0=x;
push_v(v1); v1=y;
push_v(v2); v2=v0*v1;
push_v(x); x=v2;
```

followed by the backward code

```
pop_v(x); v2_=x_; x_=0;
pop_v(v2); v0_=v2_*v1; v1_=v2_*v0;
pop_v(v1); y_+=v1_;
pop_v(v0); x_+=v0_;
```

For notational simplicity Code list variables are enumerated starting from 0 rather than $1 - n = -1$. Adjoint variables are marked by a trailing underscore. The gradient of x as an output with respect to x as an input and y at a given point (x, y) is equal to (y, x). It can be computed numerically by a single run of the adjoint code. Note that, while being conservatively correct, this adjoint code is far from optimal. Optimization of adjoint code is a major issue in ongoing research and development in the field of AD. It is beyond the scope of this paper that aims to relate the fundamental concept of attribute grammars to correct automatically generated adjoint code.

The reversal of the data-flow implies the necessity to reverse the flow of control. A simple, conservatively correct approach is to enumerate all assignments and to remember their order of execution in the augmented forward code by pushing their respective

indexes onto a control stack (`push_c`). The backward code loops over the indexes in reverse (`pop_c`) thus executing the backward codes of all assignments in the correct order. Refer to Section 4 for an example.

3 L-Attributed Grammar for Adjoint Code

A syntax-directed approach to the automatic generation of tangent-linear code has been presented in [6]. In this section we extend these ideas to adjoint codes by the definition of an appropriate L-attributed grammar. Recall that a grammar is called *L-attributed* if the values of all inherited attributes are functions of non-terminals to the left in the given production rule (includes the parent on the left-hand side). A formal proof of the correctness of our approach is omitted due to space restrictions. All one has to show is that the augmented production rules (P0)–(P10) imply Equations (5)–(11).

The input code is a sequence of one or more statements ($\mathbf{code} :: s$). In addition to assignments (a) we introduce simple branch (b) and loop (l) constructs causing a potentially nontrivial intraprocedural flow of control. Five attributes are associated with each grammar symbol: The integer attribute j represents the assignment-level code list variable indexes. A second integer attribute n is used to synthesize the sizes of subtrees (parse trees) in right-hand sides of assignments. A third integer attribute k serves as an enumerator of the assignments in the input code. There are two text attributes to hold the forward (c^f) and backward (c^b) sections of the adjoint code. The text vector c^b has length α, where α denotes the number of assignment statements in the input code. The whole adjoint code is synthesized into $\mathbf{code}.c^f$ during a successful compilation. The complete augmented forward code $s.c^f$ is followed by the reverse loop over the adjoints of all executed assignments.

$(P0)$ $\mathbf{code} ::$

$$s.k = 0$$

$$s$$

$$\begin{aligned}
\mathbf{code}.c^f = {} & s.c^f \\
& + \text{"}\texttt{int i;}\text{"} \\
& + \text{"}\texttt{while(pop_c(i)) \{}\text{"} \\
& + \quad \text{"}\texttt{if(i == 1) \{}\text{"} \\
& + \qquad s.c_1^b \\
& + \quad \text{"}\texttt{\} else if(i == 2) \{}\text{"} \\
& + \qquad s.c_2^b \\
& \quad \vdots
\end{aligned}$$

$$
\begin{aligned}
&+ \quad \text{"}\}\,\texttt{else if(i ==}\text{"} + s.k + \text{"}) \{\text{"} \\
&+ \qquad s.c^b_{s.k} \\
&+ \quad \text{"}\}\text{"}
\end{aligned}
$$

$(P1)$ s ::

$$a.k = s.k + 1$$

a

$$s.k = a.k; \quad s.c^f = a.c^f; \quad s.c^b = a.c^b$$

The vector assignment $s.c^b = a.c^b$ operates at the elemental level, that is, $s.c^b_i = a.c^b_i$ for $i = 1, \ldots, \alpha$. We chose a split way of presenting the production rules together with their associated semantic actions that resembles the implementation in Section 4. For example, the attribute k of a is set prior to parsing the assignment itself. The forward and backward codes of the nonterminal symbol on the left-hand side of the production rule are synthesized at the time of reduction (in the context of a shift-reduce parser).

$(P1a)$ s ::

$$b.k = s.k$$

b

$$s.k = b.k; \quad s.c^f = b.c^f; \quad s.c^b = b.c^b$$

$(P1b)$ s ::

$$l.k = s.k$$

l

$$s.k = l.k; \quad s.c^f = l.c^f; \quad s.c^b = l.c^b$$

$(P2)$ s^l ::

$$a.k = s^l.k + 1$$

a

$$s^r.k = a.k$$

s^r

$$s^l.k = s^r.k; \quad s^l.c^f = a.c^f + s^r.c^f; \quad s^l.c^b = s^r.c^b + a.c^b$$

The vector sum $s^l.c^b = s^r.c^b + a.c^b$ is also elemental, that is, $s^l.c^b_i = s^r.c^b_i + a.c^b_i$ for $i = 1, \ldots, \alpha$.

$(P2a)$ s^l ::

$$b.k = s^l.k$$

b

$$s^r.k = b.k$$

s^r

$$s^l.k = s^r.k; \quad s^l.c^f = b.c^f + s^r.c^f; \quad s^l.c^b = s^r.c^b + b.c^b$$

$(P2b)$ $s^l ::$

$$l.k = s^l.k$$

l

$$s^r.k = l.k$$

s^r

$$s^l.k = s^r.k; \quad s^l.c^f = l.c^f + s^r.c^f; \quad s^l.c^b = s^r.c^b + l.c^b$$

$(P3)$ $a ::$

$$e.k = a.k; \quad e.j = 0$$

$V = e;$

$$\begin{aligned} a.c^f &= e.c^f + \text{"}\texttt{push_c(}\text{"} + a.k + \text{"}\texttt{);}\text{"} \\ &\quad + \text{"}\texttt{push_v(}\text{"} + V.c^f + \text{"}\texttt{);}\text{"} \\ &\quad + V.c^f + \text{"}\texttt{= v0;}\text{"} \\ a.c^b_{a.k} &= \text{"}\texttt{pop_v(}\text{"} + V.c^f + \text{"}\texttt{);}\text{"} \\ &\quad + \text{"}\texttt{v0_ =}\text{"} + V.c^f + \text{"}\texttt{_;}\text{"} \\ &\quad + V.c^f + \text{"}\texttt{_ = 0;}\text{"} \\ &\quad + e.c^b_{a.k} \end{aligned}$$

The root of the syntax tree of the expression of the right-hand side has fixed code list variable index 0. Variable references are stored in $V.c^f$ by the scanner or some preparser depending on the regularity of the syntax for variable references.

$(P4)$ $e :: V$

$$\begin{aligned} e.n &= 1 \\ a.c^f &= \text{"}\texttt{push_v(v}\text{"} + e.j + \text{"}\texttt{);}\text{"} \\ &\quad + \text{"}\texttt{v}\text{"} + e.j + \text{"}\texttt{=}\text{"} + V.c^f + \text{"}\texttt{;}\text{"} \\ a.c^b_{e.k} &= \text{"}\texttt{pop_v(v}\text{"} + e.j + \text{"}\texttt{);}\text{"} \\ &\quad + V.c^f + \text{"}\texttt{_+ = v}\text{"} + e.j + \text{"}\texttt{_;}\text{"} \end{aligned}$$

$(P5)$ $\quad e :: C$

$$
\begin{aligned}
e.n &= 1 \\
a.c^f &= \text{"}\texttt{push_v(v"} + e.j + \text{");"} \\
&\quad + \text{"}\texttt{v"} + e.j + \text{"="} + C.c^f + \text{";"} \\
a.c^b_{e.k} &= \text{"}\texttt{pop_v(v"} + e.j + \text{");"}
\end{aligned}
$$

$(P6)$ $\quad e^l ::$

$$
\begin{aligned}
& e^r.j = e^l.j + 1; \quad e^r.k = e^l.k \\
F(e^r) \quad & \\
& e^l.n = e^r.n + 1 \\
& e^l.c^f = e^r.c^f \\
& \quad + \text{"}\texttt{push_v(v"} + e^l.j + \text{");"} \\
& \quad + \text{"}\texttt{v"} + e^l.j + \text{"="} + F.c^f + \text{"(v"} + e^r.j + \text{");"} \\
& e^l.c^b_{e.k} = \text{"}\texttt{pop_v(v"} + e^l.j + \text{");"} \\
& \quad + \text{"}\texttt{v"} + e^r.j + \text{"_="} + F_{e^r.j} + \text{"}*\texttt{v"} + e^l.j + \text{"_;"} \\
& \quad + e^r.c^b_{e.k}
\end{aligned}
$$

F is an arbitrary unary function, such as $\sin$ or $\exp$. $F_{e^r.j}$ denotes the partial derivative of F with respect to the code list variable holding the value of the expression e^r.

$(P7)$ $\quad e^l ::$

$$
\begin{aligned}
& e^{r_1}.j = e^l.j + 1; \quad e^{r_i}.k = e^l.k \ \text{ for } i = 1, 2 \\
e^{r_1} \quad & \\
& e^{r_2}.j = e^{r_1}.j + e^{r_1}.n + 1 \\
Oe^{r_2} \quad & \\
& e^l.n = e^{r_1}.n + e^{r_2}.n + 1 \\
& e^l.c^f = e^{r_1}.c^f + e^{r_2}.c^f \\
& \quad + \text{"}\texttt{push_v(v"} + e^l.j + \text{");"} \\
& \quad + \text{"}\texttt{v"} + e^l.j + \text{"=}\ \texttt{v"} + e^{r_1}.j + O.c^f + \text{"}\texttt{v"} + e^{r_2}.j + \text{";"} \\
& e^l.c^b_{e.k} = \text{"}\texttt{pop_v(v"} + e^l.j + \text{");"} \\
& \quad + \text{"}\texttt{v"} + e^{r_2}.j + \text{"_="} + O_{e^{r_2}.j} + \text{"}*\texttt{v"} + e^l.j + \text{"_;"} \\
& \quad + \text{"}\texttt{v"} + e^{r_1}.j + \text{"_="} + O_{e^{r_1}.j} + \text{"}*\texttt{v"} + e^l.j + \text{"_;"} \\
& \quad + e^{r_2}.c^b_{e.k} + e^{r_1}.c^b_{e.k}
\end{aligned}
$$

O is an arbitrary binary operator, such as $+$ or $*$. $O_{e^{r_1}.j}$ denotes the partial derivative of O with respect to the code list variable holding the value of the expression e^{r_1}. (similarly e^{r_1})

$(P8)$ $\quad b :: IF(r)$

$$s.k = b.k$$
$$\{s\}$$
$$b.k = s.k$$
$$b.c^f = \text{"}\texttt{if}\text{"} + \text{"}(\text{"} + r.c^f + \text{"})\text{"} + \text{"}\{\text{"} s.c^f + \text{"}\}\text{"}$$
$$b.c^b = s.c^b$$

$(P9)$ $\quad l :: WHILE(r)$

$$s.k = l.k$$
$$\{s\}$$
$$l.k = s.k$$
$$l.c^f = \text{"}\texttt{while}\text{"} + \text{"}(\text{"} + r.c^f + \text{"})\text{"} + \text{"}\{\text{"} s.c^f + \text{"}\}\text{"}$$
$$l.c^b = s.c^b$$

$(P10)$ $\quad r :: V^{r_1} < V^{r_2}$

$$r.c^f = V^{r_1}.c^f + \text{"}<\text{"} + V^{r_2}.c^f$$

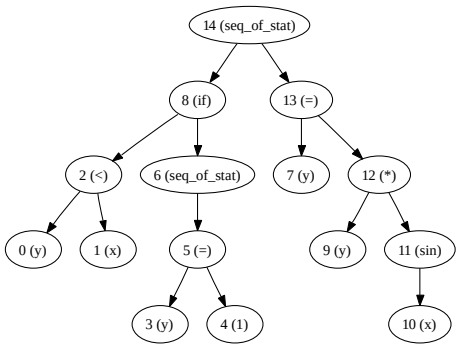


Fig. 2. Parse Tree (generated with `dot`; see `www.graphviz.org`)

Example We investigate the development of the values of all five attributes when parsing

```
if (y<x) {
  y=1;
}
y=y*sin(x);
```

The parse tree is depicted in Figure 2. Sequences of statement have been flattened into a single vertex with the corresponding statements as immediate successors (v_6 and v_{14}). The i-th vertex is referenced as v_i.

1. Synthesized subtree sizes in right-hand sides of assignments: $v_4.n = 1$, $v_9.n = 1$, $v_{10}.n = 1$, $v_{11}.n = v_{10}.n + 1 = 2$, $v_{12}.n = v_9.n + v_{11}.n + 1 = 4$.
2. Inherited code list variable indexes: $v_4.j = 0$ (P3), $v_{12}.j = 0$ (P3), $v_9.j = v_{12}.j + 1 = 1$ (P7), $v_{11}.j = v_{12}.j + v_9.n + 1 = 2$ (P7), $v_{10}.j = v_{11}.j + 1 = 3$ (P6).
3. Inherited assignment counter: $v_{14}.k = 0$ (P0) $v_8.k = v_{14}.k = 0$ (P2a), $v_6.k = v_8.k = 0$ (P8), $v_5.k = v_6.k + 1 = 1$ (P1), $v_4.k = v_5.k = 1$ (P3), $v_8.k = v_6.k = v_5.k = 1$ (P8, P2a), $v_{13}.k = v_8.k + 1 = 2$ (P2, P3), $v_{12}.k = v_{13}.k = 2$ (P3), $v_9.k = v_{10}.k = v_{11}.k = v_{12}.k = 2$ (P4–P7), $v_{14}.k = v_{13}.k = 2$ (P1, P2a).
4. Synthesized augmented forward code:
   - $v_0.c^f =$ "y" (Scanner)
   - $v_1.c^f =$ "x" (Scanner)
   - $v_2.c^f =$ "y < x" (P10)
   - $v_3.c^f =$ "y" (Scanner)
   - $v_4.c^f =$ "push_v(v0); v0 = 1;" (P5, Scanner)
   - $v_5.c^f = v_4.c^f +$ "push_c(1); push_v(y); y = v0;" (P3, Scanner)
   - $v_6.c^f = v_5.c^f$ (P1)
   - $v_8.c^f =$ "if(y < x){" $+ v_5.c^f +$ "}" (P8)
   - $v_7.c^f =$ "y" (Scanner)
   - $v_9.c^f =$ "push_v(v1); v1 = y;" (P4, Scanner)
   - $v_{10}.c^f =$ "push_v(v3); v3 = x;" (P4, Scanner)
   - $v_{11}.c^f = v_{10}.c^f +$ "push_v(v2); v2 = sin(v3);" (P6)
   - $v_{12}.c^f = v_9.c^f + v_{11}.c^f +$ "push_v(v0); v0 = v1 * v2;" (P7)
   - $v_{13}.c^f = v_{12}.c^f +$ "push_c(2); push_v(y); y = v0;" (P3, Scanner)
   - $v_{14}.c^f = v_8.c^f + v_{13}.c^f$ (P2a, P1)
5. Inherited backward code:
   - $v_4.c_1^b =$ "pop_v(v0);" (P5)
   - $v_5.c_1^b =$ "pop_v(y); v0_ = y_; y_ = 0;" $+ v_4.c_1^b$ (P3)
   - $v_8.c_1^b = v_6.c_1^b = v_5.c_1^b$ (P1, P8)
   - $v_9.c_2^b =$ "pop_v(v1); y_+ = v1_;" (P4)
   - $v_{10}.c_2^b =$ "pop_v(v3); x_+ = v3_;" (P4)
   - $v_{11}.c_2^b =$ "pop_v(v2); v3_ = cos(v3) * v2_;" $+ v_{10}.c_2^b$ (P6)
   - $v_{12}.c_2^b =$ "pop_v(v0); v2_+ = v1 * v0_; v1_+ = v2 * v0_" $+ v_9.c_2^b + v_{11}.c_2^b$ (P7)
   - $v_{13}.c_2^b =$ "pop_v(y); v0_ = y_; y_ = 0;" $+ v_{12}.c_2^b$ (P3)
   - $v_{14}.c_1^b = v_8.c_1^b (P2a); v_{14}.c_2^b = v_{13}.c_2^b$ (P2a, P1)

Finally, the augmented forward code and the backward codes of both assignments are synthesized according to production rule P0 to obtain the whole adjoint code:

$v_{14}.c^f$
```
int i;
while (pop_c(i)) { if (i==1) { v14.c1^b } else if (i==2) { v14.c2^b
} }
```

4 Implementation and Case Study

We have developed a proof-of-concept implementation based on the scanner and parser generators `flex` and `bison`. Alternatively we could have used an attribute grammar processing tool such as LISA [12] or JastAdd [9]. However, this research is part of the CompAD project – a larger R&D effort aiming at the integration of adjoint code generation capabilities into the NAGWare Fortran compiler [14]. The NAGWare compiler uses `yacc` / `bison` as a parser generator. Hence, our focus on `bison` in order to make our results directly exploitable in the context of CompAD.

The `bison` input has the following structure.

```
1  ...
2
3  %token V F IF WHILE
4  %left '*'
5
6  %%
7
8  code : s
9  s : a
10   | b
11   | l
12 s : a s { ... };
13   | b s { ... };
14   | l s { ... };
15 b : IF '(' c ')' '{' s '}' { ... };
16 l : WHILE '(' c ')' '{' s '}' { ... };
17 c : V '<' V { ... };
18 a : V '=' { ... } e ';' { ... };
19 e : e '*' e {
20     $$.j=clc++;
21     get_memory_f(&$$); get_memory_r(&$$,k);
22     sprintf($$.cf,"%s%spush_v(v%d); v%d=v%d*v%d;\n",$1.cf
           ,$3.cf,$$.j,$$.j,$1.j,$3.j);
23     sprintf($$.cb[k],"pop_v(v%d); v%d_=v%d_*v%d; v%d_=v%
           d_*v%d;\n%s%s",$$.j,$1.j,$$.j,$3.j,$3.j,$$.j,$1.j,
           $3.cb[k],$1.cb[k]);
24     free_memory_f(&$1); free_memory_r(&$1,k);
           free_memory_f(&$3); free_memory_r(&$3,k);
25   };
26   | F '(' e ')' { ... };
27   | V { ... };
28   | C { ... };
29
30 %%
```

IF, WHILE, V, F, as well as the remaining seven single-character tokens are delivered by the lexical analyzer. As an example we include the treatment of the product of two expressions. The inherited attribute j in grammar rule P7 is implemented as a global counter variable (line 20). While the order of the enumeration is changed, the necessary properties (uniqueness, correct dependences among the code list variables) are preserved. New dynamic memory is allocated for the augmented forward and adjoint codes corresponding to the nonterminal symbol e on the left-hand side of the production rule (line 21). The augmented forward code ($$.cf) is generated on line 22. The adjoint code ($$.cb[k]) is generated on line 23. Finally, the dynamic memory associated with the two nonterminal symbols on the right-hand side of the production rule is deallocated (line 24). The entire code is open source. It can be obtained by sending an email to the first author.

As a final case study consider the following simple C-code fragment.

```
if(x<y) {
  x=x*y;
  while (y<x) {
   x=sin(x*y);
  }
}
```

With x=-5.0 and y=-0.5 as inputs the while-loop is traversed once to compute x=-0.949. The corresponding gradient (0.079,1.577) is computed by a single run of the following automatically generated adjoint code.

```
1  if (x<y) {
2    push_c(1);
3    push_v(v0); v0=x;
4    push_v(v1); v1=y;
5    push_v(v2); v2=v0*v1;
6    push_v(x); x=v2;
7    while (y<x) {
8      push_c(2);
9      push_v(v0); v0=x;
10     push_v(v1); v1=y;
11     push_v(v2); v2=v0*v1;
12     push_v(v3); v3=sin(v2);
13     push_v(x); x=v3;
14   }
15 }
16 int i;
17 while (pop_c(i)) {
18   if (i==1) {
19     pop_v(x); v2_=x_; x_=0;
20     pop_v(v2); v0_=v2_*v1; v1_=v2_*v0;
21     pop_v(v1); y_+=v1_;
22     pop_v(v0); x_+=v0_;
```

```
23    }
24    else if (i==2) {
25      pop_v(x); v3_=x_; x_=0;
26      pop_v(v3); v2_=cos(v2)*v3_;
27      pop_v(v2); v0_=v2_*v1; v1_=v2_*v0;
28      pop_v(v1); y_+=v1_;
29      pop_v(v0); x_+=v0_;
30    }
31  }
```

The two assignment statements are enumerated by pushing unique indexes onto the control stack (lines 2 and 8; grammar rule P3). Assignment-level code lists are built and the values of all overwritten variables are saved on the value stack (lines 3 – 6 and 9–13; grammar rules P3–P7). The flow of control remains unchanged in the augmented forward code.

The adjoint code executes the adjoint code lists in reverse order by restoring the indexes of the corresponding original assignments (lines 16, 17, 18, and 24; grammar rule P0). All adjoint code list statements are preceded by pop accesses to the value stack to restore the old value of the variables of the left-hand side of the original code list statement (lines 19 – 22 and 25–29; grammar rules P3–P7). For example, the adjoint corresponding to the assignment on line 12 first restores the value of v3 followed by the evaluation of the product of cos(v2) (derivative of sin(v2)) with the adjoint of v3 to get the adjoint of v2 (line 26).

5 Conclusion

We are actively involved in the development of the adjoint code compilers OpenAD [15] and of the differentiation-enabled NAGWare Fortran compiler [14]. Conceptually, the semantic actions that are performed on the respective internal representations are similar to those proposed in this paper.

The syntax-directed compilation of adjoint codes for numerical programs written in suitable (subsets of) programming languages represents a low-development-cost alternative to full-size adjoint code compilers such as OpenAD or the differentiation-enabled NAGWare Fortran compiler. Due to the lack of static program analysis and the corresponding optimizations (see, for example, [8]) the output of a single-pass adjoint compiler should not be expected to have the same level of efficiency. However the conceptual insight provided by the formulation of adjoint code generation rules in form of an L-attributed grammar represents a good entry point into the subject. For example, we follow this approach in our course on “Adjoint Compilers" taught at the Department of Computer Science at RWTH Aachen University.

The proposed method can be modified and extended to decrease the computational complexity through a decrease of the memory requirement. The use of compiler-generated variables in the context of assignment-level code lists can be reduced drastically. Arithmetic expressions (required for the local partial derivatives) need to be synthesized instead of indexes of the corresponding code list variables. This work is to be continued

on the basis of graduate- and undergraduate-level projects. Currently we are investigating the syntax-directed compilation of adjoint Matlab code. Matlab is certainly not a single-pass language. However we expect to find a suitable large enough subset having this property.

References

1. A. Aho, R. Sethi, and J. Ullman. *Compilers. Principles, Techniques, and Tools*. Addison-Wesley, 1986.
2. M. Berz, C. Bischof, G. Corliss, and A. Griewank, editors. *Computational Differentiation: Techniques, Applications, and Tools*, Proceedings Series. SIAM, 1996.
3. M. Bücker, G. Corliss, P. Hovland, U. Naumann, and B. Norris, editors. *Automatic Differentiation: Applications, Theory, and Tools*, number 50 in Lecture Notes in Computational Science and Engineering, Berlin, 2005. Springer.
4. G. Corliss, C. Faure, A. Griewank, L. Hascoet, and U. Naumann, editors. *Automatic Differentiation of Algorithms – From Simulation to Optimization*, New York, 2002. Springer.
5. G. Corliss and A. Griewank, editors. *Automatic Differentiation: Theory, Implementation, and Application*, Proceedings Series. SIAM, 1991.
6. D. Gendler, U. Naumann, and E. Varnik. Syntax-Directed Tangent-Linear Code. In *Proceedings of the International IADIS Conference on Applied Computing 2007 (IADIS AC07)*, pages 425–430, February 2007.
7. A. Griewank. *Evaluating Derivatives. Principles and Techniques of Algorithmic Differentiation*. Number 19 in Frontiers in Applied Mathematics. SIAM, Philadelphia, 2000.
8. L. Hascoët, U. Naumann, and V. Pascual. To-be-recorded analysis in reverse mode automatic differentiation. *Future Generation Computer Systems*, 21:1401–1417, 2005.
9. G. Hedin and E. Magnusson. Jastadd: an aspect-oriented compiler construction system. *Sci. Comput. Program.*, 47(1):37–58, 2003.
10. D. Knuth. Semantics of context-free languages. *Mathematical Systems Theory*, 2(2):127–145, 1968.
11. J. Marotzke, R. Giering, Q. K. Zhang, D. Stammer, C. N. Hill, and T. Lee. Construction of the adjoint MIT ocean general circulation model and application to Atlantic heat transport sensitivity. *J. Geophys. Res.*, 104:29,529 – 29,548, 1999.
12. M. Mernik, V. Žumer, M. Lenič, and E. Avdičaušević. Implementation of multiple attribute grammar inheritance in the tool lisa. *SIGPLAN Not.*, 34(6):68–75, 1999.
13. S. Muchnick. *Advanced Compiler Design and Implementation*. Morgan Kaufmann Publishers, San Francisco, 1997.
14. U. Naumann and J. Riehme. A differentiation-enabled Fortran 95 compiler. *ACM Transactions on Mathematical Software*, 31(4):458–474, 2005.
15. J. Utke, U. Naumann, C. Wunsch, C. Hill, P. Heimbach, M. Fagan, N. Tallent, and M. Strout. OpenAD/F: A modular, open-source tool for automatic differentiation of Fortran codes. *ACM Transactions on Mathematical Software*, 2007. To appear.

Proceedings of the International Multiconference on Computer Science and Information Technology pp. 1147–1150
ISSN 1896-7094
© 2007 PIPS

On applying stochastic programming in mathematical theory of programming

Viliam Slodičák, Valerie Novitzká, Anita Verbová

Department of Computers and Informatics,
Faculty of Electrical Engineering and Informatics, Technical University of Košice
Letná 9, 042 00, Slovakia
viliam.slodicak@tuke.sk, valerie.novitzka@tuke.sk, anita.verbova@tuke.sk,
Website: http://kpi.fei.tuke.sk/

Abstract. Stochastic programs are mathematical programs where some of the data incorporated into the objective or constraints are uncertain. Probabilistic programming is one of methods in stochastic programming. It is a programming in which the probabilities of the values of the variables are of interest. Our idea is that solving of a problem we can construct in logical reasoning over some mathematical theories. In this approach we use category theory for construction of logical system. We discuss here application of one method of stochastic programming in our approach.

We can model many decision problems by using mathematical programs, which seek to maximize or minimize some objective which is a function of the decisions. The possible decisions are usually constrained by limits in resources, minimum requirements, etc. Decisions are represented by variables, which may be, for example, nonnegative or integer, i.e. they have some types. Objectives and constraints are functions of the variables, and problem data. Examples of problem data include unit costs, production rates, sales, or capacities. Stochastic programs are mathematical programs where some of the data incorporated into the objective or constraints are uncertain. Probabilistic programming is one of methods in stochastic programming. It is a programming in which the probabilities of the values of the variables are of interest. For example, if the probability distribution for inputs is known, it is possible to calculate the probability distributions of outputs. Predicative programming is a programming according to a first-order semantics; it is a way of writing programs so that each programming step is proven as it is made [3]. Usually the first step is to decide what quantities are of interest, and to introduce a variable for each such quantity. A specification is defined as a boolean expression whose variables represent the quantities of interest. In a specification, some variables may represent inputs, and some may represent outputs. A specification is implemented on a computer when, for any values of the input variables, the computer generates values of the output variables to satisfy the specification. In other words, we have an implementation when the specification is true for every computation. The specification is

deterministic if, for each input, the satisfactory output is unique. A program is a specification that has been implemented, so that computer can execute it.

Our idea is that running program can be described by mathematical logic. We use Curry-Howard corresponence for propositions as types and proofs as programs. For this purpose we use linear logic. It is a type of substructural logic that denies the structural rules of weakening and contraction. Girard's linear logic [2] (introduced in 1987) becames a natural mean for research and applications in computer science. It has offered great promise, as a formalism particularly well-suited to serve at the interface between logic and computer science. It is able to describe systems that are changed during they are used. Applications of linear logic cover the theory of concurrent processes, logic programming and functional programming. Using Curry-Howard correspondence, propositions of linear logic are interpreted as types [2].

Our approach is that to solve large scientific problems by mathematical machines we always start with the formulation of their theoretical foundations. We need to formalize these theoretical foundations as logical reasoning in some mathematical theories because the programs should really prove the correctness of their results.

Category theory is a part of mathematics. Its importance for theoretical computer science growth in last decade. Categorical methods are already well-established for the semantical foundation of type theory or data type specification frameworks [1]. Categories are structures which enable to work with objects of arbitrary complexity. Fundamentals of category theory are relations between objects. These relations are expressed by morphisms—they usually correspond to computations, proofs or another forms [5].

A program consists of data structures and algorithms. Data structures always have some types. As a first step in problem solving we express types and operations over types for given problem by many-typed signature which consists of finite set of basic types necessary for a given problem and of finite collection of function symbols. From basic types by using type constructors we can construct more complex Church's types, polymorphic types or dependent types. We construct type theory for a given problem as classifying category with contexts as objects and terms as category morphisms. We define it as a symmetric monoidal closed category and its construction can be found in [4]. This classifying category is a base category for fibration, which is a special functor. Fibration allows substitution and indexing. We construct the subcategory called the fibre over each type. A fibre contains linear logic over that type and type indexes the fibre. The objects of fibre are formulas and morphisms are sequents. Term t in classifying category induces the substitute functor t^*, which is a part of proof tree. Then all the total category expresses the linear logic over the given signature together with defined set of axioms.

In this contribution we present some aspects about probabilistic predicative programming and how to interpret this style in categorical terms. First we assume a countably infinite set of variables $Var = (v_1, v_2, \ldots)$ that range over basic types. Every variable in a term has assigned a unique basic type from a signature,

written $v : \tau$ as a variable declaration. A finite sequence $\Gamma = (v_1 : \tau_1, \ldots, v_n : \tau_n)$ of variable declarations is called a type context. A sequent of term calculus has a form $\Gamma \vdash M : \tau$ and is read as a term M of type τ with variables in Γ. Terms can be constructed using the basic rules: identity rule and rule for function symbol. In constructing the linear classifying category $LinCl(\Sigma)$ containing variable declarations of Church's types we have the advantage that we can use Church's types instead type contexts as category objects [5]. Our category $LinCl(\Sigma)$ contains linear types as objects and linear terms as morphisms [4]. In process of solving problems of stochastic programming we have to introduce types of variables by extending the inputs and outputs so that each variable has the type. The list of variables we consider as a type contexts which are in correspondence with product types. We construct classifying category for a given problem with objects as types, morphisms as terms (parts of computations), identity for each object and with composition of morphisms. We use linear logic formulas to express the problem: we have to formulate linear formulas that use resources of given problem and work with them. The formulas we connect by implication which simulates stepwise execution. To illustrate the combination of nondeterminism and probability, we look at well-known Monty Hall's problem from [3]. Monty Hall is a famous game where there are three doors. A prize is hidden behind one of the doors. The contestant chooses a door. Monty then opens one of the doors, but not the door with the prize behind it, and not the door the contestant has chosen. Monty asks the contestant whether the contestant would like to change their choice of door, or stay with their original choice. We formulate the logical formulas describing the parts of problem. We need inpust: random contestant's choice, the door with the prize and Monty's choice. First step is to state the door where the prize is hidden. But the contestan't doesn't know, where the prize is, so we have to use non-deterministic additive disjunction. Next step is description of contestant's choice and Monty Hall's reaction. If the contestant choices the door with the prize behind, Monty Hall can decide which of the remaining doors will be opened. His choice is depend on his own. He is able to decide e.g. according to side of coin. We introduce a variable representing random contestant's choice. It can be expressed as result of random value from numbers of doors. Second variable value of Monty Hall's choice and can be expressed as a random value from two values. The last step is the contestant's change of choice after opening the doors. All this steps we describe by formulas of linear logic and connect them with linear implication. According those formulas we are able to construct a program describing this game.

In our work we formulate some aspects about our approach of scientific problem solving. We introduce types into given problem and construct the type theory for the issue. Then we formulate how to construct linear logic over the types. This is the base for one method of stochastic programming - the probabilistic programming. Finally we show the famous Monty Hall's problem in mathematical theory of programming as an example of construction the solution.

Acknowledgement

This work was supported by VEGA Grant No.1/2181/05: Mathematical Theory of Programming and its Application in the Methods of Stochastic Programming.

References

1. Ehrig H. Applied and computational category theory. In *Bulletin of the EATCS no 89* (June 2006), European Association for Theoretical Computer Science, pp. 134–135.
2. Girard J., Taylor P., and Lafont Y. *Proofs and Types.* Cambridge University Press, 1990, pp. 1–175.
3. Hehner E. Probabilistic Predicative Programming. (2004), Department of ComputerScience, University of Toronto, Toronto, Canada. `http://citeseer.ist.psu.edu/626001.html`
4. Novitzká V., Mihályi D., Slodičák V. Categorical models of logical systems in the mathematical theory of programming. In *MaCS'06 6th Joint Conference on Mathematics and Computer Science, Book of Abstracts* (2006), University of Pécs, Hungary, pp. 13–14.
5. Novitzká V., Slodičák V., Verbová A. On Modeling Higher-Order Logic. In *Ivan Plander (ed.): Proceeding from International Scientific Conference Informatics 2007, Bratislava* (June 2007), pp. 156-162