Proceedings of the International Multiconference on Computer Science and Information Technology pp. 229–236
ISBN 978-83-60810-14-9
ISSN 1896-7094

Semantic Relevance Measure between Resources based on a Graph Structure

Sang Keun Rhee, Jihye Lee, Myon-Woong Park
Intelligence and Interaction Research Center
Korea Institute of Science and Technology
39-1 Hawolgok-dong, Seongbuk-gu
Seoul, Korea
Email: greyrhee@kist.re.kr

***Abstract*—Upon developing an information system, establishing an effective information retrieval method is the main problem along with the management of the information. With ontologies, the semantic structure of knowledge can be represented and the resources can be managed along with their context information. Observing that within this knowledge space, resources are not isolated but connected to each other by various types of relations, we believe that utilising those relations can aid information provisioning. In this paper, we explore the concept of the *semantic relevance* between resources and the semantic structure of a knowledge space, and present a relevance measuring algorithm between resources based on a graph structure, with two implementation examples.**

I. Introduction

IN ANY knowledge space, each information resource does not exist in isolation but is connected to other resource(s) in many different types of relations. The relation may be explicitly represented or can be inferred, and resources can be directly linked or their relations may be indirect via another resource(s). Also, even two resources seem to have no relation whatsoever in a certain environment, they may have a contextual relation within another knowledge space. Concerning that the vast amount of readily available information in not only the Web but also in a limited knowledge space such as a single organisation, the information retrieval has been a challenging problem, and we believe that finding the semantic relevance between resources can contribute to effective information provisioning. To find the semantic relevance, the knowledge space needs to be semantically structured and represented, and a method in finding and measuring the relevance is also required.

In our earlier work [1], we presented our semantic relevance measure that calculates the relevance value between objects in a graph structure. In this paper, we enhance our initial approach by further discussing the meaning of the semantic relevance and describing the knowledge management within an information system with a knowledge model and an ontology, and creating the *Relevance Graph*, an interpretation of a knowledge model and the base structure of the relevance calculation, is explained in more detail. In addition, the relevance calculation algorithm is improved by adapting the *Edge Scaling* method presented in [2], and two systems which utilises this semantic relevance measure are introduced to describe examples of how this method can be implemented.

This work is partially sponsored by the Korea Institute of Science and Technology (KIST) with the project TSI(Tangible Space Initiative).

II. Semantic Similarity and Semantic Relevance

There have been various different views and approaches to discover the similarity or relatedness among information objects. One of the most obvious relation that can be discovered between resources is the *semantic similarity*, which is the likeness between objects. For textual information such as documents, it means the similarity of their contents, and this similarity can be measured by lexical matching methods. One of the traditional methods is the TF-IDF(term frequency-inverse document frequency) [3] based comparison, and the LSA(latent semantic analysis) [4] provides an automatic way of organising documents into a semantic structure for information retrieval. The semantic similarity can also be found between terms as well as textual documents. Within a set of structured terms, as a tree-based hierarchy, several approaches have been introduced including an edge-based method [5] which measures the similarity from the number of edges between terms, a node-based methods [5], [6] that claim to be more accurate than the edge-based measure, and hybrid methods [7], [8] that overlay the edge-based approach with the node-based one.

From a different point of view, the similarity or relation between resources can be measured based on the links that connect the resources, and the *PageRank* [9] is possibly the most well-known link analysis algorithm. In [10], the links in the blogosphere are interpreted into an influence model and the *spreading activation technique* [11] is applied on the influence graph. Another approach is finding the relevance between objects based on tags by users, and a relevance measuring method based on tagging has been introduced in [12] for pictogram retrieval.

Now, let us describe our point of view on semantic relevance. We have discussed the semantic similarity between documents above, but sometimes it may be necessary to find more than the simple similarity of contents. Two documents can be related even though they do not contain similar contents or have links to each other, and such relation may be affect the resource ranking. For instance, suppose we have a document

describing an interface design methodology and another document explaining a data mining algorithm. In this case, these two documents are not considered to be similar, since their contents are about two different topics. However, if there is a research project developing a recommender system, where its user interface is designed following the methodology described in the first document and the recommendation algorithm is implemented based on the contents of the second document, then these two documents now have an indirect relation between each other within this project. Here, the fact that the methods described in both documents are utilised in the same project is the context information, and the two documents are related to each other in the given context although they are neither similar nor directly linked.

The semantic relevance can also be found between two objects which have completely different nature. For example, a person and a document are semantically related if the person is the author of the document. This can be further extended if the document has a topic and there exists a research project which is related to the topic, then in this case, a relation between the person and the project is discovered even if the person does not know the project exists.

Therefore, the *semantic relevance* we are exploring in this paper is based on any kinds of relations that can be represented in a knowledge space, and we are also considering indirect relations via context information. Now, to utilise such relevance in an information system, the degree of importance that each relation represents is considered as a numeric value, in other words, the closeness between two objects in the given knowledge space, and those values are then analysed to calculate the relevance value between any two objects. The relevance value calculation algorithm is described later in this paper.

III. Semantic Knowledge Management

A. Semantic Knowledge Space

Within an information system, there exists a set of information, and not only the information objects but also the relations among them are important for effective information provisioning. These information objects and their relations form a *knowledge space*, and its structure needs to be represented semantically to build a semantic knowledge space. The simplest form of a knowledge space consists of a set of information (or data) that is to be provided to users - in other words, *resources* - without any semantic annotations. Then, links between those resources can be included in the knowledge space to represent relations among those resources. An intuitive example of this would be a set of documents where each document contains links to other document(s). In this knowledge space, the connections between the resources are described, but this structure still does not have semantic meanings as the relations are represented only via *syntactic links*. To describe the semantics of the resources, the meaning of each relation needs to be assigned to those links so that they become *semantic links*. In addition to those semantically linked resources, context information can be added to the knowledge space to represent richer semantic meanings of those resources within a certain context.

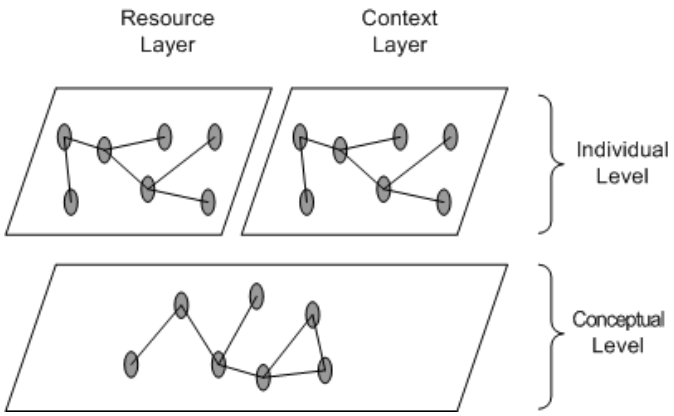

Fig. 1. Knowledge Model

Therefore, a *semantic knowledge space* is a semantic structure of information which contains resources, context information, and relations between them where each relation has its own meaning. We will now describe the structure of a semantic knowledge space in detail with a knowledge model and an ontological representation.

B. Knowledge Model and Ontology

In our point of view, a semantic knowledge space consists of two types of information, context information and resources. As discussed above, resources are the information objects that are to be provided to users, and context information is the additional environmental or domain information. For effective knowledge management, before describing each individual information objects and their relations, a structure of knowledge space needs to be designed in conceptual level first, then the individual information objects can be represented within the conceptual structure. To represent such semantic knowledge, ontology provides a suitable formal structure, and we will now discuss the ontological representation of a knowledge space along with a knowledge model view, with a simple example ontology of a research organisation. Note that the example ontology presented in this section is severely simplified mainly to support explaining the knowledge structure and relevance calculation. This example will be used throughout the rest of this paper.

The structure of our knowledge model is shown in Fig. 1 to describe the overview of a semantic knowledge space from our point of view. First, in the *Conceptual Level*, the structure of concepts, the relations between them, and additional attributes are defined. Suppose we want to manage and provide information on people and documents within a research organisation, then four concepts may be defined in this level, *Project*, *Topic*, *Person*, *Document*, and the relations between these concepts can also be specified. Therefore, the *Conceptual Level* can be seen as a base structure of an ontology specifying the concepts and their properties, without individual instances. Fig. 2 presents our example ontology.

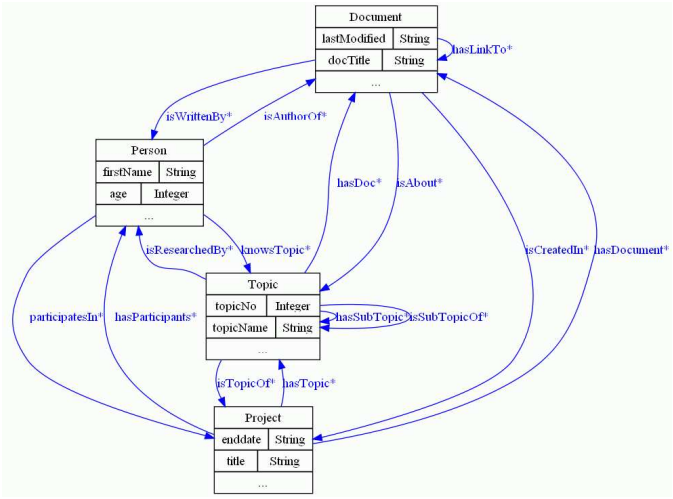

Fig. 2. An example ontology

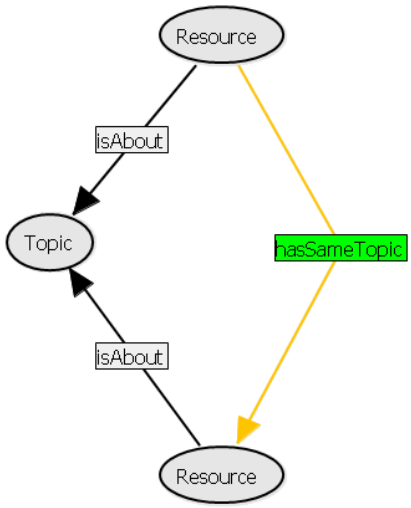

Fig. 3. An example rule

The *Individual Level* contains the resources and context information represented as instances of the ontology defined in *Concept Level*, and it is divided into two layers - the *Resource Layer* and the *Context Layer*. The *Resource Layer* contains the information to be provided to users, and in our example, a set of documents and a set of people can be regarded as the contents of this layer. The *Context Layer* contains the domain and context knowledge, and a hierarchy of topics and a list of projects are stored in this layer in our example. The contents of these layers are semantically linked to each other based on the relations defined in the *Conceptual Level*.

IV. Relevance Graph

Having the structured knowledge space, our next objective is measuring the semantic relevance between resources, represented in the *Resource Layer*. To discover their relevance, we interpret the knowledge space (i.e. ontology) into a graph structure where the relevance can be calculated as a numerical value. A *Relevance Graph* is an interpreted semantic knowledge model based on a knowledge structure, or ontology, representing the information objects and their relations. It is defined as a directed labelled graph $G = (V, E)$ where:

- V is a set of nodes, representing individuals;
- E is a set of edges, representing relations between individuals.

Note that our *Relevance Graph* does not have any restrictions in its structure. It is not limited to a tree structure, and different edges can represent different relation types. There may be multiple edges between two adjacent nodes, and the graph can contain cycles. Also, depending on each implementation and their purposes, a whole ontology can be interpreted into a graph, or only a part of an ontology can become the relevance graph. Or, even multiple ontologies can form a single relevance graph, although an ontology matching would be required in this case, and it is not discussed in this paper. The creation of the *Relevance Graph* includes node creation, edge creation, and edge labelling.

A. Node Creation

There can be two different ways of creating nodes in our *Relevance Graph*. First, all instances of an ontology can become nodes, or second, only the instances representing *resources* can be created as nodes. The former is simpler way, but the result graph become bigger so it requires more computational time in the relevance calculation phase. The latter requires defining additional rules to clarify the relations between resources via context information since those relations may disappear upon graph creation as individuals representing context information is not included in the graph. For example, upon creating the nodes based on the example ontology described in the previous section (Fig. 2), each instance representing a `Document` or a `Person` becomes each node, but the individuals of `Topic` or `Project` do not appear on the graph, hence indirect relations between resources via these two conceptual information can be lost. Therefore, additional rules need to be defined to keep the meaningful indirect relations, and one example rule may be representing ‘a document is about the same topic as another document.’ This example rule is shown in Fig. 3 and this can be represented in SWRL [13] as follows:

```
<ruleml:imp>
  <ruleml:_rlab ruleml:href="#sametopic"/>
  <ruleml:_body>
    <swrlx:individualPropertyAtom
      swrlx:property="isAbout">
      <ruleml:var>d1</ruleml:var>
      <ruleml:var>t1</ruleml:var>
    </swrlx:individualPropertyAtom>
    <swrlx:individualPropertyAtom
      swrlx:property="isAbout">
      <ruleml:var>d2</ruleml:var>
      <ruleml:var>t1</ruleml:var>
    </swrlx:individualPropertyAtom>
  </ruleml:_body>
  <ruleml:_head>
    <swrlx:individualPropertyAtom
      swrlx:property="hasSameTopic">
```

```
        <ruleml:var>d1</ruleml:var>
        <ruleml:var>d2</ruleml:var>
      </swrlx:individualPropertyAtom>
    </ruleml:_head>
  </ruleml:imp>
```

However, finding all meaningful indirect relations between resources and defining them as rules may not be a trivial task, hence in practice, the recommendable method is combining both ways of node creation depending on the characteristics of each application—creating all resources and 'some' context instances as nodes, and adding additional rules.

B. Edge Creation

The next step is creating edges between the nodes in the *Relevance Graph*. The edges represent the relations between nodes, and they can be created from the relations (or object properties in OWL) defined in the ontology. Not only explicit relations but also inferred ones between resources by the additional rules become edges. Depending on the knowledge structure and application, all such relations in the ontology can become edges or only the relations which represent meaningful relevance in the system can be selected. Note that any reflexive relations are ignored hence do not become edges, since they have no significance in measuring relevance between nodes:

$$\forall x \in V : \qquad e(x, x) \notin E$$

C. Edge Labelling

The edges in our *Relevance Graph* represent various relations defined in the ontology, and each relation has different importance in terms of representing the 'closeness' or 'relevance' between objects within a knowledge space. For instance, if we have two relations `hasSameTopic` and `hasSameAuthor` representing relations between documents, `hasSameTopic` can be regarded to be more important than `hasSameAuthor` because having the same topic means the contents of the two documents are very likely to have a close relationship whereas an author may write two different articles in two completely different area. Based on this assumption, each relation on the ontology can be weighted with a different value representing its importance within the system, and these values become the labels of the edges in the *Relevance Graph*. Note that, the value we are assigning here is the 'distance' value, which is defined as the inverse of the relevance value:

$$Distance = \frac{1}{Relevance} \tag{1}$$

Upon implementation, the distance value assignment can be included in the ontology (as in [14] and [2]) or in the graph creation module (as in [15]). Currently, we do not have an automated way of assigning the relevance value to each relation, so it needs to be done manually by the ontology developer and/or a domain expert. Practically, all distances can be instantiated with a single value, and the value for each relation can then be updated by applying the developers' domain knowledge and throughout testing, which was the initial implementation approach in [15] and was also followed in [2]. Also, there can be a (semi-)automatic way of updating the distance value via historical evaluation methods based on the system usage and explicit/implicit feedback, and this will be explored and experimented further in our future work.

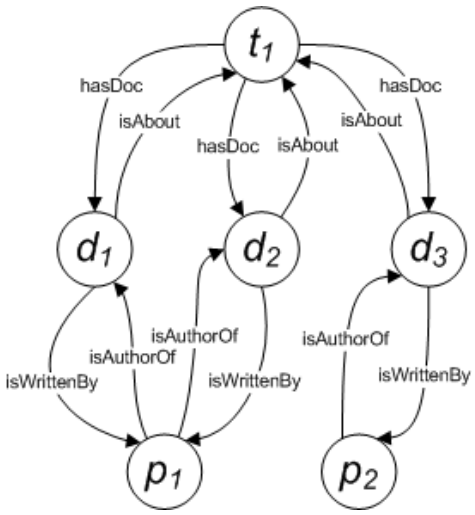

Fig. 4. An example illustration of relations among instances

V. Relevance Calculation

Having created the *Relevance Graph*, the relevance value between two nodes in the graph can now be calculated by a relevance calculation algorithm, which is based on our earlier work [1]. Before we proceed further, let us elucidate the assumptions of our algorithm and define the terms to clarify their meanings and avoid possible confusion. The two fundamental assumptions are as follows:

1) Each relation has different importance, and it can be represented as a numeric value.
2) Having more relations between two resources means that they are closer(more relevant).

The assumption 1) is already discussed in Section IV-C, and the assumption 2) can be explained with an example case presented in Fig. 4. In this figure, we have a graphical representation of relations among a topic(t_1), persons(p_1, p_2), and documents(d_1, d_2, d_3). Considering the relevance between documents d_1 and d_2, and between d_2 and d_3, it is safe to say that—it may not be true in all cases but has high probability—d_1 is likely to be more relevant to d_2 than d_3 is, as d_1 has the same topic(t_1) and the same author(p_1) as d_2 while d_3 has the same topic(t_1) but a different author(p_2). Based on this assumption, the merging algorithm is developed instead of taking a shortest(or longest) edge(or path) or taking the mean value (see Section V-B and V-D).

The terms used in the rest of this paper is defined as follows:

- *Relevance* represents the closeness between two entities. Each relation has its own relevance value.
- *Distance* represents how far two concepts are away from each other. The distance value is the inverse of the relevance value (see equation (1)).

- *Weight* represents the closeness between two entities in individual level. The weight value (0,1) is used to scale the distance (or relevance) value appropriately to each individual.

Now, the relevance values between the nodes in our *Relevance Graph* are measured by the following four steps.

A. Edge Scaling

The first step is individualising each edge's label (i.e. distance). It was not included in our original work [1] but developed while implementing the relevance calculation method in a *duty-trip support system* [2]. The edge label is set based on the importance of a certain relation in comparison to other relations in conceptual level(see Section IV-C), hence all instances connected by the same relation have the same distance value. However, depending on the application and its knowledge structure, it might be desired to represent the different degree of relevance to certain relations in individual level. For example, in Fig. 2, we defined the `isAbout` relation that connects `Document` and `Topic`, and the `isAbout` relation has a certain *distance* value. However, there can be two different documents related to the same topic but with different degree. Here, the degree of relevance can be defined in the ontology as following:

```
onto:Doc1
  a onto:Document ;
  onto:isAbout
    [ a onto:DocTopic ;
      onto:relatedTopic onto:Ontology ;
      onto:weight "0.90"^^xsd:float
    ] .

onto:Doc2
  a onto:Document ;
  onto:isAbout
    [ a onto:DocTopic ;
      onto:relatedTopic onto:Ontology ;
      onto:weight "0.25"^^xsd:float
    ] ;
  onto:isAbout
    [ a onto:DocTopic ;
      onto:relatedTopic onto:Java ;
      onto:weight "0.40"^^xsd:float
    ] .
```

The above example presents two documents `Doc1` and `Doc2` which are both related to a topic `Ontology` with different weight values. To apply this in our relevance graph, we multiply the initial *relevance value* by this *weight value* and obtain the scaled distance value.

$$newRelevance = relevance \times weight \tag{2}$$

Therefore, from the equation (1), for an edge $e(x,y) \in E$ where x and y are two adjacent nodes $x, y \in V$, its initial distance value $D_{e(x,y)}$, and the weight value w_{xy}, the scaled distance value $D'_{e(x,y)}$ is:

$$D'_{e(x,y)} = (\frac{1}{D_{e(x,y)}} \times w_{xy})^{-1} \tag{3}$$

We can regard this process as *individualisation* of the edge distance since the initial distance was determined in conceptual level and it is scaled by the weight value assigned in individual level. This step is optional part in relevance calculation since it can only be applied when the *weight* values are defined between instances in the knowledge structure (i.e. ontology).

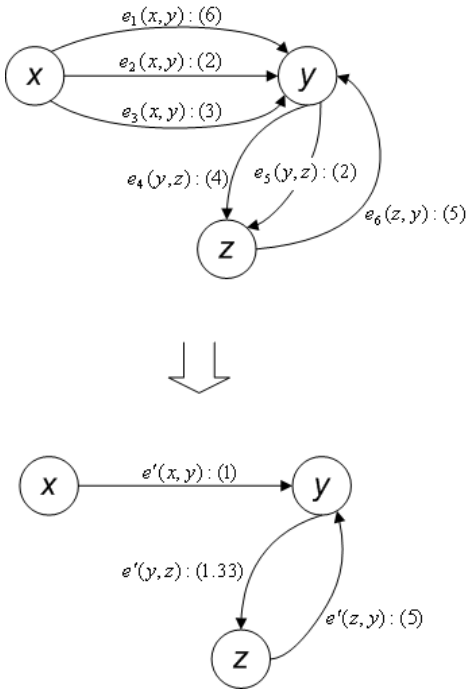

Fig. 5. Edge Merging

B. Edge Merging

In our *Relevance Graph*, there may be multiple edges with the same direction between two adjacent nodes, since two resources can be connected via more than one relation. Therefore, from our second assumption of the algorithm described at the beginning of this section, we merge those edges into a single one so that we can obtain a simpler graph where there is only a single edge with the same direction between two adjacent nodes. To calculate the distance value of the merged edge, we first calculate the relevance between two adjacent nodes. For two adjacent nodes $x, y \in V$, and edges $e_1(x,y), e_2(x,y), \ldots, e_n(x,y) \in E$ from node x to node y with the distance value $D_{e_i(x,y)}$ for edge $e_i(x,y)$ where $1 \leq i \leq n$, the semantic relevance value r_{xy} from node x to node y by direct relations(edges) is as follows:

$$r_{xy} = \sum_{i=1}^{n} \frac{1}{D_{e_i(x,y)}} \tag{4}$$

Therefore, from the equation (1), the new distance value $D_{e'(x,y)}$ of the merged edge $e'(x,y)$ is:

$$D_{e'(x,y)} = \frac{1}{r_{xy}} = (\sum_{i=1}^{n} \frac{1}{D_{e_i(x,y)}})^{-1} \tag{5}$$

By merging edges, we can now obtain a simpler graph where there is only a single edge with the same direction be-

tween adjacent nodes. Fig. 5 describes an example illustration of edge merging.

C. Path Distance Calculation

Now we need to consider the relevance between non-adjacent nodes. For two non-adjacent nodes, there may or may not exist a path. In our algorithm, a path is valid if and only if it contains no repeated nodes (i.e. simple path). If there does not exist a path between them, it means that they are not related and we can consider their relevance value as 0. If there exist one or more paths, we first need to calculate the distance value of each path.

In graph theory, the weight(or distance) of a path in a weighted graph is the sum of the weights of each edge in the path. Following this, for a path $P(a_1 a_n)$ that visits n nodes $a_1, a_2, \ldots, a_n \in V$, the path distance $D_{P(a_1 a_n)}$ is:

$$D_{P(a_1 a_n)} = \sum_{k=1}^{n-1} D_{e'(a_k a_{k+1})} \tag{6}$$

However, instead of selecting a single path, we consider all relations in our relevance calculation, and it is discovered that this can produce undesirably close relevance results for between two nodes which are connected via long paths. Precisely, from a node x, a node y should be closer than a node z but the result can be the opposite because of the number of paths from x to z even though each path is very long. Hence, it is necessary to make the distance value between indirectly related nodes higher than the simple sum, and this is done by multiplying the edge count k to the distance value of each edge. Therefore, the above equation (6) is replaced with the following:

$$D_{P(a_1 a_n)} = \sum_{k=1}^{n-1} (k \times D_{e'(a_k a_{k+1})}) \tag{7}$$

D. Path Merging

The last step of the relevance calculation is handling multiple paths between two nodes, and this is done by following the same principle as edge merging. For n paths $P_1, P_2, \ldots, P_n$ from node $x \in V$ to node $y \in V$, the relevance value R_{xy} is:

$$R_{xy} = \sum_{k=1}^{n} \frac{1}{D_{P_k(xy)}} \tag{8}$$

R_{xy} represents the final semantic relevance value of node y from node x.

VI. Implementation Examples

The semantic relevance measure introduced in this paper has been implemented in two systems—RIKI and SPIP.

A. RIKI

RIKI[15] is a Wiki-based portal supporting group communication and knowledge sharing within a collaborative research project. Unlike other *Semantic Wikis*[16][17][18], ontological knowledge management is applied in RIKI mainly to provide easier knowledge access within a specific project's context, rather than rich semantic annotation. Its knowledge space is developed following the knowledge model and structure introduced in this paper (Section III), and the knowledge space is structured in F-Logic[19] ontology. Two top-level concepts are defined in this ontology, the `Resource` containing the Wiki articles and the `Context` representing the context information. These are further divided into four sub-concepts, classifying the article types and describing different types of the contexts. The overview of this structure is depicted in Fig. 6.

RIKI provides two ways of navigating the articles—the *Structured Browsing* and the *Article Recommendation*. The *Structured Browsing* provides a tree-based structure of the contexts, so that users can see the overall structure of the information in RIKI and reach the desired article(s) related to a certain topic, task, event, or person. By the *Article Recommendation* function, while a user is viewing an article, the system generates and displays a list of its relevant articles. To create the *Relevance Graph* from the ontology, first, all the instances of the `Resource` concept and its subconcepts (i.e. articles) are created as nodes. Then, for edge creation, we defined 50 rules representing the indirect relations between those articles via the given contexts, and those rules became edges of our graph. From this *Relevance Graph*, when a user opens an article, the system finds the top 10 relevant articles from the current article by the algorithm presented in this paper. The *Edge Scaling* part is not included in this RIKI implementation. Fig. 7 presents the RIKI interface, with the *Structured Browsing* on the top-left and the *Article Recommendation* on the bottom-left. The details of the overall system can be found in [15].

B. Smart Personalized Information Provider

Another implementation example of the semantic relevance measure is the *Smart Personalized Information Provider*, which is sponsored by the KIST-SRI PAS "Agent Technology for Adaptive Information Provisioning" grant. It aims for adaptive information provisioning in an agent-based virtual organisation, and provides a personalised information by ontological resource matching. The knowledge space is developed in OWL (details in [20]), and it provides two different services—the *Grant Announcement Service* and the *Duty Trip Support*. In both services, for resource matching, the relevance calculation algorithm is utilised, along with a SPARQL[21] engine for resource filtering and the *GIS Subsystem* for geospacial information management. The matching process is described in detail in [14] for the *Grant Announcement Service* and also in [2] for the *Duty Trip Support*.

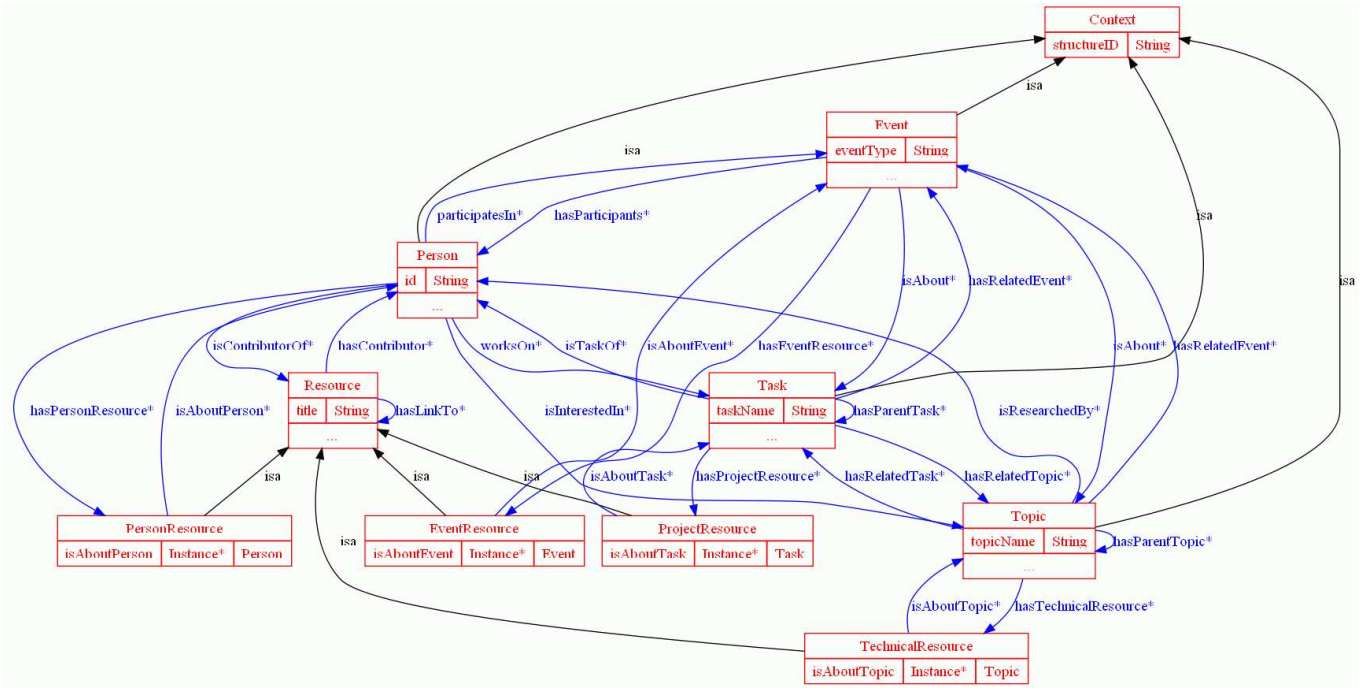

Fig. 6. The RIKI ontology.

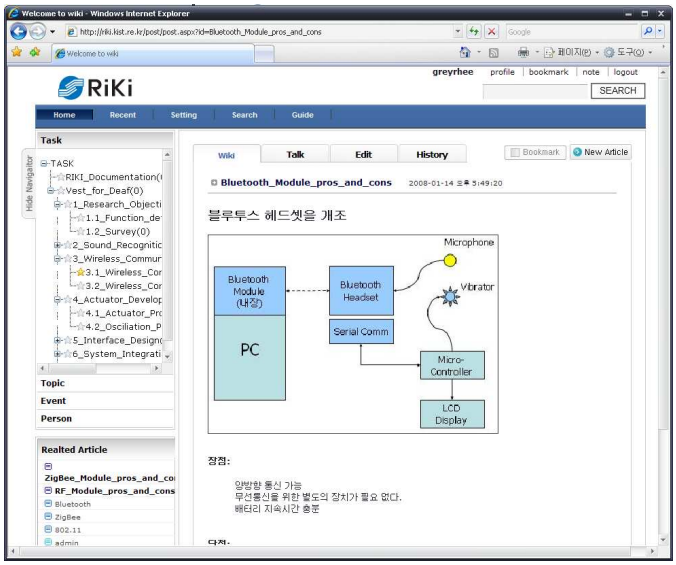

Fig. 7. The RIKI interface.

VII. Concluding Remarks

In this paper, we discussed our view of the concept *semantic relevance* and presented a structured view of a semantic knowledge space, with the semantic relevance measure algorithm between the resources based on a graph structure. The relevance measure algorithm can be implemented on its own, but the efficiency will be more evident when it is combined with existing searching or matching algorithms, such as the text-based comparing methods and/or tagging-based relevance measure. Our future work will include the integration of our method with those other techniques, as well as exploring a semi-automatic method of initiating and updating the edge labels, experimenting with various implementations and evaluating throughout extensive testing.

References

[1] S. K. Rhee, J. Lee, and M.-W. Park, "Ontology-based Semantic Relevance Measure," *in Proceedings of the 1st SWW Workshop(ISWC)*, Korea, 2007.

[2] M. Szymczak, G. Frackowiak, M. Ganzha, M. Paprzycki, S. K. Rhee, J. Lee, Y. T. Sohn, J. K. Kim, Y.-S. Han, and M.-W. Park, "Ontological Matchmaking in an Duty Trip Support Application in a Virtual Organization," *in Proceedings of the IMCSIT'08 Conference*, 2008, in press.

[3] R. R. Korfhage, *Information Storage and Retrieval*, Wiley, 1997.

[4] S. T. Dumais, G. W. Furnas, T. K. Landauer, S. Deerwester, and K. Harshman, "Using latent semantic analysis to improve access to textual information," *in Proceedings of the CHI'88 Conference*, ACM Press, 1988, pp. 281–283.

[5] P. Resnic, "Using Information Content to Evaluate Semantic Similarity in a Texonomy," *IJCAI*, 1995, pp. 448–453.

[6] D. Lin, "An Information-Theoretic Definition of Similarity," *in 15th International Conference on Machine Learning*, Morgan Kaufmann, San Francisco, CA, 1988, pp.296–304.

[7] J. J. Jiang and D. W. Conrath, "Semantic Similarity Based on Corpus Statistics and Lexical Taxonomy," *in Proceedings of the International Conference on Research on Computational Linguistics*, Taiwan, 1997.

[8] C. Leacock and M. Chodorow, "Combining Local Context and WordNet Similarity for Word Sense Indentification," *WordNet: An Electronic Lexical Database*, MIT Press, Cambridge, MA, 1998, pp. 265–283.

[9] S. Brin and L. Page, "The Anatomy of a Large-Scale Hypertextual Web Search Engine," *in Proceedings of the 7th WWW Conference*, Brisbane, Australia, 1998.

[10] A. Java, P. Kolari, T. Finin, and T. Oates, "Modeling the Spread of Influence on the Blogosphere," *in Proceedings of the 15th WWW Conference*, 2006.

[11] F. Crestani, *Application of Spreading Activation Techniques in Information Retrieval*, Artificial Intelligence Review, 1997.

[12] H. Cho, T. Ishida, R. Inaba, T. Takasaki, and Y. Mori, "Pictogram Retrieval Based on Collective Semantics," *in Proceedings of the 12th HCI Conference*, LNCS 4552, 2007, pp. 31–39.
[13] SWRL: A Semantic Web Rule Language Combining OWL and RuleML, http://www.w3.org/Submission/SWRL/.
[14] M. Szymczak, G. Frackowiak, M. Ganzha, M. Paprzycki, S. K. Rhee, J. Lee, Y. T. Sohn, J. K. Kim, Y.-S. Han, and M.-W. Park, "Infrastructure for Ontological Resource Matching in a Virtual Organization," *in Proceedings of the IDC'2008 Conference*, Studies in Computational Intelligence, N. Nguyen and R. Katarzyniak, Eds., vol. 134, Heidelberg, Germany: Springer, 2008, pp. 111–120.
[15] S. K. Rhee, J. Lee, and M.-W. Park, "Riki: A Wiki-based Knowledge Sharing System for Collaborative Research Projects," *in Proceedings of the APCHI 2008 Conference*, LNCS 5068. Springer, 2008.
[16] M. Krotzsch, D. Vrandecic, M. Volkel, H. Haller, R. Studer, "Semantic MediaWiki," *Journal of Web Semantics*, 2007, pp. 251–261.
[17] E. Oren, "SemperWiki: A Semantic Personal Wiki," *in Proceedings of the Semantic Desktop Workshop*, 2005.
[18] S. Schaffert, "IkeWiki: A Semantic Wiki for Collaborative Knowledge Management," *in Proceedings of the 1st International Workshop on Semantic Technologies in Collaborative Applications*, 2006.
[19] M. Kifer, G. Lausen, and J. Wu, "Logical Foundations of Object-Oriented and Frame-Based Languages," *Journal of ACM*, 1995.
[20] M. SzymczakMichal, et al.M. Szymczak, G. Frackowiak, M. Gawinecki, M. Ganzha, M. Paprzycki, M.-W. Park, Y.-S. Han, and Y. Sohn, "Adaptive Information Provisioning in an Agent-Based Virtual Organization – Ontologies in the System," *in Proceedings of the KES-AMSTA Converence*, LNAI, N. Nguyen, Ed., vol. 4953. Heidelberg, Germany: Springer, 2008, pp. 271–280.
[21] SPARQL Query Language for RDF, http://www.w3.org/TR/rdf-sparql-query/.