Position papers of the Federated Conference on Computer Science and Information Systems, pp. 25–32 DOI: 10.15439/2017F554 ISSN 2300-5963 ACSIS, Vol. 12

Concepts Ontology Algebras and Role Descriptions

Cyrus F Nourani*
Akdmkrd AI Research Affiliate
TU Berlin, Germany
Acdmkrd@gmail.com &
cyrusfn@alum.mit.edu

Patrik Eklund
Umeå University,
Department of Computing Science
Umeå, Sweden
peklund@cs.uum.se

***Abstract*—A heterogeneous computing model with ontology preserving functions are applied to present concept learning across domains with structural agent morphisms. A computing models based on a novel multi-agent competitive learning with multiplayer game tree plans are applied. Agents are assigned to transform the models to reach goal plans. Goals are satisfied based on competitive game tree learning. Agent tree computing models are example prototypes for modeling ontology algebras. Specific agents are assigned to transform the models to reach goal plans where goals are satisfied based on competitive game tree learning. Cooperating agents, that have opened new avenues in modeling and implementing agent teams, are ingredients to specific application modeling. Applications to Formal Concept Description are developed with new description logic algebraic models. Novel Description algebras with concept description ontology algebras and description ontology preservation morphisms are presented.**

***Index Terms*—Game Learning Ontology Algebras, Description Ontology, Ontology Preservation Morphisms, Competitive Model Ontology, Agent Ontology Models, Game Tree Learning.**

I. Desire Models

AN OVERVIEW to a practical agent learning based on new competitive modeling a technique applying what the first author developed since 2004 is presented with augmentation to standard agent modeling [11]. A specific agent might have internal state set I, which the agent can distinguish its membership. The agent can transit from each internal state to another in a single step. With our multi-board model agent actions are based on I and board observations. Transfer learning is carried on with agent morphisms. Predictive and competitive model learning is presented applying agent game trees. Ontology preservation principles are introduced for learning ontology. The preservation principles are further applied to the knowledge bases that support the transfer learning. Competitive game tree learning is the basis to the authors' application to business and economics game modeling. Deduction models attain a new perspective with the techniques here. Context abstraction and met-contextual reasoning is introduced as a new field. Multi-agent visual multi-board planning has been applied in the first author's projects to space navigation and spatial computing learning. In a haptic computing logic [8] the learning process can be seen as an emotional and personal, game based, and proactive Game-based Learning, emotions and emotional agents, henceforth abbreviated as the BID model [16].

The section overviews are as follows. Section two develops the stage for the agent computing models that are applied to characterize agent computations based on standard Desire modeling augmented with newer agent module algebras. Section 3 presents the competitive modeling techniques with signatured trees. Tree computations to realize goals for competitive models are the bases for model compatibility characterizations on realizing goals on computation trees. Generic model diagrams are applied to compare models. Section 4 presents signatured tree morphisms and module preservation techniques based on alternative agent computing techniques. Agent algebras and morphisms render a basis for defining ontology preservation principles. Section 5 applies the techniques to model-based concept learning with preservation morphism mappings for transfer learning across domains. Secion 6 develops the new basis for ontology algebras on Concept Descriptions. A categorical characterization encompasses a constructive description logic with concept description algebra monads on agent signature trees. Based on that new concept ontology algebras with description ontology algebra preservation theorem are presented. Newer application areas that can be explored are mutual robot learning – a robot introducing a structure to a new robot. These areas have started being explored at Singularity university affiliate groups, for example. Robot learning based on watching the task being performed by a human or by a second robot are model-based learning but troublesome due to a mismatch between the model structure problems e.g. [21]. Newer examples are on learning topological spaces [20]. Our more functional approach to learning about the world can be applied to physical robots transformed into an abstract model, and then converting it back into a functional representation.

II. Desire Models

Let us start with the popular agent computing model the Beliefs, Desire, and Intentions, BID is a generic agent computing model specified within the declarative compositional modeling framework for multi-agent systems, DESIRE. The model, a refinement of a generic agent model, explicitly specifies motivational attitudes and the static and dynamic relations between motivational attitudes. Desires, goals, intentions, commitments, plans, and their relations are modeled [6] . Different notions of strong and weak agency are presented at [22]. To apply agent computing with intelligent multimedia some specific roles and models have to be presented for agents. Beliefs, intentions, and commitments play a crucial role in determining how rational agents will act. Beliefs, capabilities, choices, and commitments are the parameters making component agents specific. DESIRE is the framework for design, and the specification of interacting reasoning components is a framework for modeling, specifying and implementing multi-agent systems, see [6], [22]. The interaction between components, and between components and the external world is explicitly specified. Components can be primitive reasoning components using a knowledge base, but may also be subsystems that are capable of performing tasks using methods as diverse as decision theory, neural networks, and genetic algorithms.

A. Specifying BID Agents

The BID design specifications in our papers apply agent signature trees. Information is encoded with a predicate logic on a hierarchically ordered sort structure (order-sorted predicate logic). Newer techniques with levels of signatures [17], [8] can be applied to the encoding. Units of information including sorts and operators of different arities are represented on the signature on the first level. On the second level, operators are type constructors, so that the set of variable-free terms are shifted down to the sort set for the signature on level three. In this way, different (meta)levels may be distinguished and richer type constructions can be obtained and used. Some specifics and a mathematical basis to such models with agent signatures might be obtained from [1] where the notion had been introduced since 1994. Meta-level information contains information about object-level information and reasoning processes; for example, for which atoms the values are still unknown (epistemic information). Similarly, tasks that include reasoning about other tasks are modeled as meta-level tasks with respect to object-level tasks.

III. Competitive Models and Signatured Trees

Planning is based on goal satisfaction at models. Multi-agent planning, in this paper is modeled as a competitive learning problem where the agents compete on game trees as candidates to satisfy goals hence realizing specific models where the plan goals are satisfied. When a specific agent group "wins" to satisfy a goal the group has presented a model to the specific goal, presumably consistent with an intended world model. For example, if there is a goal to put a spacecraft at a specific planet's orbit, there might be competing agents with alternate micro-plans to accomplish the goal [4]. While the galaxy model is the same, the specific virtual worlds where a plan is carried out to accomplish a real goal at the galaxy via agents are not. The plan goal selections and objectives are facilitated with competitive agent learning. The intelligent languages [15] are ways to encode plans with agents and compare models on goal satisfaction to examine and predict via model diagrams why one plan or model is better than another or to prevent traversing unsuccesful routes.

B. Intelligent AND/OR Trees and Search

AND/OR trees Nilsson e.g. [23] are game trees defined to solve a game from a player's stand point.

Formally a node problem is said to be solved if one of the following conditions hold.

1. The node is the set of terminal nodes (primitive problem – the node has no successor).

2. The node has AND nodes as successors and the successors are solved.

3. The node has OR nodes as successors and any one of the successors is solved.

A solution to the original problem is given by the subgraph of AND/OR graph sufficient to show that the node is solved. A program which can play a theoretically perfect game would have task like searching and AND/OR tree for a solution to a one-person problem to a two-person game. An agent AND/OR tree [1] is and AND/OR tree where the tree branches are intelligent trees. The branches compute a Boolean function via agents. The Boolean function is what might satisfy a goal formula on the tree. An intelligent AND/OR tree is solved iff the corresponding Boolean functions solve the AND/OR trees named by agent functions on the trees. Thus node m might be f(a1,a2,a3) & g(b1,b2), where f and g are Boolean functions of three and two variables, respectively, and ai's and bi's are Boolean valued agents satisfying goal formulas for f and g. An intelligent AND/OR tree is solved iff the corresponding Boolean functions solve the AND/OR trees named by intelligent functions on the trees. Thus node m might be f(a1,a2,a3) & g(b1,b2), where f and g are Boolean functions of three and two variables, respectively, and ai's and bi's are Boolean valued agents satisfying goal formulas for f and g.

A tree game degree is the game state a tree is at with respect to a model truth assignment, e.g. to the parameters to the Boolean functions above. Let generic diagram or G-diagrams be diagrams definable by specific functions. Intelligent signatures [1] are signatures with designated multiplayer game tree function symbols. A soundness and completeness theorem is proved on the intelligent signature language by the first author [7]. The techniques allowed us to present a novel model-theoretic basis to game trees, and generally to the new intelligent game trees.

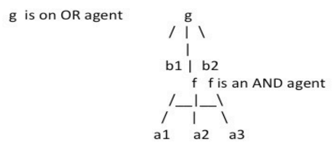

Figure 1: Agent Logic Tree

C. Trees and Model Compatibility

Now let us examine the definition of situations from 1985 times and view it in the present formulation.

Definition 3.1 A situation consists of a nonempty set D, the domain of the situation, and two mappings: g,h. g is a mapping of function letters into functions over the domain as in standard model theory. h maps each predicate letter, pn, to a function from Dn to a subset of {t,f}, to determine the truth value of atomic formulas as defined below. The logic has four truth values: the set of subsets of {t,f}.{{t},{f},{t,f},0}. the latter two is corresponding to inconsistency, and lack of knowledge of whether it is true or false.

The above truth value assignments indicate that the number of situations exceeds the number of possible worlds. The possible worlds being those situations with no missing information and no contradictions. From the above definitions the mapping of terms and predicate models extend as in standard model theory. Next, a compatible set of situations is a set of situations with the same domain and the same mapping of function letters to functions. In other worlds, the situations in a compatible set of situations differ only on the truth conditions they assign to predicate letters.

Definition 3.2 Let M be a structure for a language L, call a subset X of M a generating set for M if no proper substructure of M contains X, i.e. if M is the closure of X U {c_M : for c, a constant symbol of L}. An assignment of constants to M is a pair <A,G>, where A is an infinite set of constant symbols in L and G: A M, such that {G[a]: a in A} is a set of generators for M. Interpreting a by g[a], every element of M is denoted by at least one closed term of L[A]. For a fixed assignment <A,G> of constants to M, the diagram of M, D<A,G>[M] is the set of basic [atomic and negated atomic] sentences of L[A] true in M. [Note that L[A] is L enriched with set A of constant symbols.]

Generic diagrams, denoted by G-diagrams, were what we defined since 1980's to be diagrams for models defined by a specific function set, for example Σ_1 Skolem functions.

Remark: The functions above are those by which a standard model could be defined by inductive definitions.

The frist author proved [5] that situations are compatible iff their corresponding generalized diagrams are compatible with respect to the Boolean structure of the set to which formulas are mapped (by the function h above, defining situations). To examine compatibility on model diagrams minimal prediction was developed around 1994. The artificial intelligence technique defined since the author's model-theoretic planning project, is a cumulative nonmontonic approximation attained with completing model diagrams on what might be true in a model or knowledge base. The predictive diagrams [9] are applied to discover models to the intelligent game trees. Prediction is applied to plan goal satisfiability and can be combined with plausibility [5] probabilities, and fuzzy logic, e.g. [13], [17] to obtain, for example, confidence intervals.

IV. SIGNATURED MORPHISMS AND MODULE PRESERVATION

From the software agent designer's viewpoint, however, there is modularity with artificial structures. Artificial structures [7] implemented by agent morphisms. Knowledge acquisition requires either interviewing an expert, brainstorming with a group of experts, or structuring one's thoughts if the specifier is the expert. For multi-agent designs there are active learning agents and automatic learning. The author first author had presented the notion of Nondeterministic Knowledge (Design_Agents) [7]. Design_Agents is formulated to deal with the conceptualization stage and is being applied by the present project to define active learning by agents.

Design_Agents requires the user to inform the specifier as to the domains that are to be expected, i.e. what objects there are and what the intended actions (operations) on the objects are, while fully defining such actions and operations. The actions could be in form of processes in a system. The relations amongst the objects and the operations (actions) can be expressed by algebras and clauses, which the specifier has to present. The usual view of a multi-agent systems might convey to an innocent AI designer that an agent has a local view of the environment, interacts with others and has generally partial beliefs (perhaps erroneous) about other agents. On the surface the Design_Agents specification techniques might appear as being rigid as to what the agents expect form other agents. The Design_Agents specification does not ask the agents be specified up to their learning and interaction potential. Design_Agents only defines what objects might be involved and what might start off an agent. It might further define what agents are functioning together. Thus specifications are triples <O,A,R> consisting of objects, actions and relations. Actions are operations or processes.

A.The Formal Basis

Starting with what are called hysterectic agents [11]. A hysterectic agent has an internal state set I, which the agent can distinguish its membership. The agent can transit from each internal state to another in a single step. Actions by agents are based on I and board observations. There is an external state set S, modulated to a set T of distinguishable subsets from the observation viewpoint. An agent cannot distinguish states in the same partition defined by a congruence relation. A sensory function s :S → T maps each state to the partition it belongs. Let A be a set of actions which can be performed by agents. A function action can be defined to characterize an agent activity action:T →A. There is also a memory update function mem: I x T → I. To define agent at arbitrary level of activity knowledge level agents are defined. All excess level detail is eliminated. In this abstraction an agent's internal state consists entirely of a database of sentences and the agent's actions are viewed as inferences based on its database. The action function for a knowledge level agent maps a database and a state partition t into the action to be performed by an agent in a state with database and observed state partition t. action: Dx T→ A. The update function database maps a state and a state partition t into a new internal database. database: D x T → D. A knowledge-level agent is an environment is an 8-tuple shown below. The set D in the tuple is an arbitrary set of predicate calculus databases, S is a set of external states, T is the set of partitions of S, A is a set of actions, see is a function from S into T, do is a function from A S into S, database is a function from D x T into D, and action is a function from D x T into A.

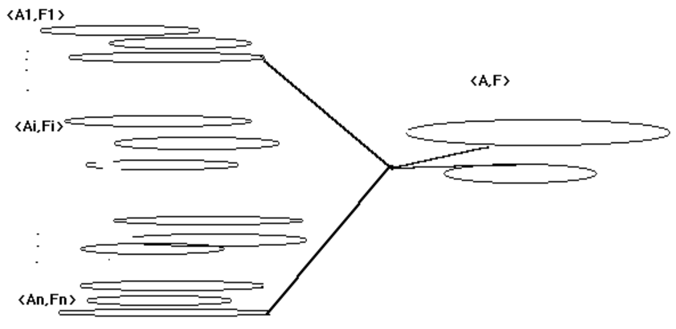

Figure 2 Heterogenous Module Computing Model

B. Agent Model Morphisms

Let A be a set of actions which can be performed by agents. A function action can be defined to characterize an agent activity action:T → A. There is also a memory update function. A hysterectic agent HA defined by a sextuple <I,S,T,A,s,d,internal,action> where d is a function form A x S → S and internal I x T → I. Let HA be a set of sextuples defining a hysterectic agents. Define HA morphims by a family of functions defined component-wise on the sextuple above.

Definition 4.1 A HA morphism is a function F : HA → HA' defined component-wise by F[i]: I→ I'; F[S]: S → S', F[T]: T →T', F[A]: A →A'; F[s]: S→ T'; F[d]: A' x S' → S' and F[internal]: I' x T'→ I'.

Definition 4.1 implies F defines a new hysterectic agents from HA by a morphism. Component-wise definitions for a morphism might be viewed as functions on a multi-sorted signature carrying the sextuple. Similar morphisms can be defined for knowledge level agents which we can refer to by KD-morphisms.

C. Agents, Modules, and Algebras

The computing enterprise requires more general techniques of model construction and extension, since it has to accommodate dynamically changing world descriptions and theories. The models to be defined are for complex computing phenomena, for which we define generalized diagrams. They were designed to build models with prespecified generalized Skolem functions. The specific minimal set of function symbols is the set with which a model fro a knowledge base can be defined. The G-diagram techniques allowed us to formulate AI worlds, KB's in a minimal computable manner to be applied to agent computation. The techniques in [5] for model building as applied to the problem of AI reasoning allow us to build and extend models through diagrams. A technical example of algebraic models defined from syntax had appeared in defining initial Σ algebras for equational theories of data types [2] and our research in [1]. In such direction for computing models of equational theories of computing problems are presented by a pair (Σ,E), where is a signature (of many sorts, for a sort set S and E a set of -equations.

Definition 4.2 An s-sorted signature Σ or operator domain is a family <w,s> of sets, f or s S and w S* (where S* is the set of all finite strings from S , including the empty string). call f <w,s> and operation symbol of rank w,s; of arity w, and of sort s.#

We apply multi-sorted algebras via Definition 2.3 to multi-agent systems.

Definition 4.3 Let Σ be an S-sorted signatures. A Σ-algebra A consists of a set As for each s S (called the carrier if A of sort s) and a function <A>: As1 x As2 x....xAsn As for each <w,s>, with w=s1s2...sn (called the operation named by). For <,s>, A As, i,e the (set of names) of constants of sort s. #

Definition 4.4 If A and B are Σ algebras, a -homomorphism h:A → B is a family of functions <hs:As Bs> s in S that preserve the operations, i.e. that satisfy (h0) For <,s>, the hs(A) = B; (h1) If , For <w,s>, with w=s1s2...sn and <a1,...,an> As1 x As2 x....xAsn, then hs[A(a1,...,an)] = B(hs(a1),...,hs(an)).

From [1], [7] we have the following notions:

Definition 4.5 A signature is intelligent iff it has intelligent function symbols. We say that a language has intelligent syntax if the syntax is defined on an intelligent signature.

Definition 4.6 A language L is said to be an intelligent language iff L is defined from an intelligent syntax.

A practical example of intelligent languages was presented composed from <O,A,R> triples as control structures, e.g. SERF [15]. The functions in AF are the agent functions capable of message passing. The O refers to the set of objects and R the relations defining the effect of A's on objects. Amongst the functions in AF only some interact by message passing. The functions could affect objects in ways that affect the information content of a tree. There you are: the tree congruence definition thus is more complex for intelligent languages than those of ordinary syntax trees. Let us define tree information content for the present formulation. Hence there is a new frontier for a theoretical development of the <O,A,R> algebras and that of the AII theory. <O,A,R> is a pair of algebras, <Alg[A],Alg[F]>, connected by message passing and AII defines techniques for implementing such systems. To define AII we define homorphisms on intelligent signature algebras. For an intelligent signature IΣ, let $T_{I\Sigma}$ be the free tree word algebra of signature IΣ. The quotient of $T_{I\Sigma}$ the word algebra of signature , with respect to the I-congruence relation generated by a set of equations E, will be denoted by T<IΣ,E>, or T<P> for presentation Component-wise definitions for a morphism might be viewed as functions on a multi-sorted signature carrying the sextuple. Similar morphisms can be defined for knowledge level agents which we can refer to by KD-morphisms. The techniques in [5] for model building as applied to the problem of AI reasoning allows us to build and extend models through diagrams. The notion of an intelligent signature [1] is simply a designation that there is a subsignature with specific properties, for example all the functions are 1-1.

Definition 4.7 A IΣ-homomorphism is a I-homomorphism defined on algebras with intelligent signature IΣ. To define agent specific designs we apply HA-morphisms via the following definition.

Definition 4.8 Let A and B be IΣ-algebras with signatures containing an agent signature HA.

A HA-homomorphism from A to B is an IΣ-homorphism with defined HA-morphism properties.

V. Learning , Concepts, and Ontology Preservation

Our transfer learning model applies the BID model to specify learning areans M1 and M2. Each arean's BID is presented with intelligent signatures IΣ1 and IΣ2. Predictive model compatibility techniques are presented with agent signature game trees where the above fomalizm can is applied to realize competitive learning models. The following process is applied to transfer game tree and competive model learning across domains since modeling and realizability are based on morphism preserved formulas.

The term ATL here refers to the process of abstract transfer leraning from an abstract characterization of a world, or leraning domain to a second arena or world. Thus ATL express the relationship between two forms of representations. The notion of abstract transfer learning are either algebraic or model-theoretic (algebraic logic) definitions. We refer to specifications of the form <O,A,R> as presentations that present an IM_BID system. We also expect a presentation of the form <I[O],I[A],I[R]> [15] for the implementing abstract or concrete machine. The former could be the designer's conceptualization, and the latter the specification of the syntax and semantics of a programming language. Informally the A TL process is that of encoding the algebraic structure of the conceptualization of a problem onto the algebra that specified an learning machine, or a secondary BID specifed world. The ATL process becomes that of defining specific agent and structural morphisms on the above BID algebras. Each of the functions defined by <O,A,R> are implemented by agents, that characterize the implementation function

I:<O,A,R> → <I[O],I[A],I[R]> is to be defining a mapping I: <Alg[A],Alg[F]> → <Alg[I(A)],Alg[I(F)]>. We refer to Alg[A] and Alg [F] are what we call ontology algberas. The implementation mapping I defines wrappers to resources in a manner preserving the ontology algebra. Ontology algerbas are multi-sorted algerbas defining multi-agent systems defined by formal agents, e.g., hysterectic or knowledge level agents and agent morphisms [14], [15].

Example 1: Data and Knowledge Bases: The ATL Ontology Preservation Principle, following is the first author's 1997 ontology preservation principles: The ATL is a valid transfer only if it preserves the ontology algebras. Since the knowledge-base is essential to learning designs, let us carry on the ontology preservations to Widerhold's domain knowledge base algebra DKB [16] consists of matching rules linking domain ontologies. There are three operations defined for DKB.

Example 2: Mutual Robot Learning

Based on that new concept ontology algebras with description ontology algebra preservation theorem are presented. Newer application areas that can be explored are mutual robot learning – a robot introducing a structure to a new robot. These areas have started being explored at Singularity university affiliate groups, for example. Newer examples are on learning topological spaces [20]. Our more functional approach to learning about the world can be applied to physical robots transformed into an abstract model, and then converting it back into a functional representation.

The operations are: Intersection – creating subset ontology and keeping sharable entries; Union – creating a joint ontology merging entries; Difference – creating a distinct ontology and removing shared entries. Mapping functions must be shown to preserve ontologys. Structural morphism allow ontology structures to be mapped from one robot ontology Knowledge base to a new robot with alternate ontology descripptions and structures, thereby transfer learning to a new robot with alternate ontology with structure preserving morphisms.

Based on that new concept ontology algebras with description ontology algebra preservation theorem are presented. Newer application areas that can be explored are mutual robot learning – a robot introducing a structure to a new robot. These areas have started being explored at Singularity university affiliate groups, for example. Robot learning based on watching the task being performed by a human or by a second robot are model-based learning but troublesome due to a mismatch between the model structure problems e.g. [21]. Newer examples are on learning topological spaces [20]. Our more functional approach to learning about the world can be applied to physical robots transformed into an abstract model, and then converting it back into a functional representation. Let us apply the definition for HA agents and HA morphisms to state a preservation theorem. Let A and B be IΣ-algebras with the signature IΣ containing HA agents. Let Alg[B] be an IΣ-algebra defined from B implementing, e.g. [15] a specified functionality defined by A. An ATL is an implementation for Alg[A] by Alg[B]. Theorems 5.1 and 5.2 are from the first author's 2001 times, c.f. [15].

Definition 5.1 Let A and B be IΣ-algebras with intelligent signature IΣ containing agents. An I-ontology is an IΣ-algebra with axioms for the agents and functions on the signature.

Theorem 5.1 Let A and B be IΣ -algebras with the signature IΣ containing HA agents. The AII with HA morphisms defined from A to B preserve IΣ-ontology algebras iff defined by HA-homorphisms.

Proof definition for the ontologies, HA morphism, definition 4.7 and 4.8, IΣ-algebras and IΣ-homorphisms entail the IΣ-ontology axioms are preserved iff agents are carried by HA-homorphisms from A to B.

Theorem 5.2 Let A and B be IΣ-algebras with the signature I containing KD agents. The AII with KD morphisms preserve IΣ -ontology algebras iff defined by KD-homorphisms.

Proof Similar to 5.1. DKB mappings are specific ATL's were the ontology algebra operations are the same at source and target. We can prove based on the above that DKB mappings are DKB preservation consistent.

VI. Formal Concept Description Ontology Algebra

FCA is abstracted on so called "context", or "formal context", but is in the end just a relation on sets, $I \subseteq G \times M$, often written as and said to be a triple (G,M,I). G is called these to f"objects"and M these to f"attributes". However, neither objects nor attributes are given any specific syntactic structure. The call for intuitive meaning, but as such there is no syntactic structure [24,25] whatsoever based on which objects and attributes move beyond being just points in sets. This obviously makes real-world applications difficult to develop, and application content is all in that intuitive structure, and none of it is embrace by the syntactic notion itself. Basically, in FCA, G and M are indeed just plain sets, but in this starting point they can be seen as objects in the category Set of sets and functions. Further, even if in traditional FCA, the elements of those sets have no structure whatsoever, these sets can be provided with generalized structure [17], which formalizes FCA categorically, thereby opening up possibilities to give "object" and "attribute" more precise meanings given their syntactic structure, also going beyond just using Set as the underlying category for FCA, and, adopting a much more generalized view on relations.

In traditional FCA (Wille 1982), a so called "formal concept", or just a "concept", is a pair (A, B), with $A \subseteq G$ and $B \subseteq M$, such that $A = \{g \in G \mid gIm \text{ for all } m \in B\}$ and $B = \{m \in M \mid gIm \text{ for all } g \in A\}$. A lattice, the so called "formal concept lattice", is given for the set of all concepts by $(A1,B1) \leq (A2,B2)$ if and only if $A1 \subseteq A2$ (or, equivalently, $B1 \supseteq B2$). Since there is no convention about how to use given names for objects and attributes in "informally constructed" names for formal concepts, combining names into names for concepts, or simply inventing the names otherwise, has become tradition within FCA. This, however, means that there is no terminological or ontology basis for FCA, but concepts themselves are seen as ontology objects. The ontology preservation areas will be further developed to present concept ontology mappings and preservations.

In the following subsection we show how constructive and type-theoretic methodology can provide enriched structures for FCA. The constructive approach Paive (2002) adapts classical ALC to a constructive system using the two routes outlined above. The syntax of such constructive system is the same in both cases. Concept descriptions in this constructive description logic CDL language obey the following syntax rule

$$C, D \rightarrow A \mid T \mid \perp \mid C \sqcap D \mid C \sqcup D \mid C \rightarrow D \mid \forall R.C \mid \exists R.C$$

where C, D range over concepts, A is an atomic concept and R ranges over names of roles, as before. As usual in constructive logics, since $\neg C$ is simply an abbreviation for $C \rightarrow \perp$ we do not need to consider it. In compensation we must add in the constructive implication of concepts, which in classical description logic is a derived concept. Also it is just a convenience to have the true concept T, as it could be defined as $\neg\perp$. We are then within the realm of first order logic IFOL.

The type-theoretic approach shows how concepts as singleton concepts correspond to "individual concept", whereas syntactic powers of concepts correspond to "concept".

A. Categorical Characterizations

In this subsection we point out that $\exists$ in $\exists R.C$ as a modality is actually an informal symbol. Further, as typing comes into play, we show how C is syntactically ambiguous in this context as the underlying signature is not precisely described.

In the following we use notations from (Schmidt-SchaussSmolka 1991). Note that D for the universe should not be confused with D as used for concept descriptions, e.g., in expressions like $C \sqcup D$, D is not to be understood as D in D^I, where I is the interpretation. With C as a "concept", we have C^I as a subset of D^I, which in turn is an element PD^I, where P is the powerset functor. The "existential quantifier" in $\exists R.C$ is an "R-modality" applied to the powerconcept C.

The definition for the semantic expression $(\exists R.C)^{I}$ uses the existential quantifier that appears in the assumed underlying set theory. Concerning the underlying signature and related variables, in (Schmidt-SchaussSmolka1991) the situation is unclear, given the assumption about the existence of two further disjoint alphabets of symbols, which are called individual and concept variables. Logically, variables are not part of any alphabet. Variables are terms, and as such they are terms of a certain type. We should therefore speak of "individual concept" rather than "individual variable". Now typing of "concept" and "individual concept" comes into play, and we will need type constructors on level two of the so called three-level arrangement of signatures [17]. As opposed to (Schmidt-SchaussSmolka1991), we say "concept" instead of "individual concept", and "powerconcept" instead of "concept". The underlying signature must be formalized, where concept is a sort in the given underlying signature on level one. On level two, Pconcept becomes a constant operator, and a type constructor P is then used to produce a new type Pconcept, which in their `algebra' will be understood, respectively, as D^{I} and PD^{I}. Simply typed description logic can now be formally defined in lambda-calculus [17]. A concept on level one becomes a "singleton powerconcept" on level three, and the syntactic expression $\exists R.C$ $app_{P(Pconcept),\ Pconcept}$ (m, $app_{Pconcept,\ P(Pconcept)}$ (R,C)) where m is the multiplication of the underlying monad, and app is the function type constructor.

For transforming description logic into our categorical framework, we use notations in [6]. Interpretations $I = (D^{I}, I)$, where I maps every coFncept description to a subset of DI, use D for that universe, which should not be confused with D as used for concept descriptions, e.g., in expressions like C D, where D is not to be understood as the "D in D^{I}". With C as a "concept", we have $C^{I} \subseteq D^{I} \in P\ D^{I}$. This means that $P\ D^{I}$ is the actual 'algebra'. Roles R are semantically described as relations $RI \subseteq D^{I} \rightarrow D^{I}$, i.e., we can equivalently write it as a substitution $R^{I} : D^{I} \rightarrow D^{I}$. The observation that relations $R \subseteq X \rightarrow X$ correspond precisely to functions (in form of substitutions) $R : X \rightarrow PX$, where P is the powerset functor over the category of sets and functions, is the basis for viewing generalized relations as morphisms (substitutions) in the Kleisli category over generalized powerset monads. With C as a "concept", we have $C^{I} \subseteq D^{I} \rightarrow PD^{I}$. This means that PD^{I} is the actual 'algebra'.

Definition 6.1 A Description algebra morphism h: $PD^{I} \rightarrow PD^{I'}$ where I and I' are alternate interpretation functions such that h preserves roles R on D.

Following definitions on HA morphims and the state space agent model above, we have description algebras defined on an agent signature Σ. Considering a sequence description competitive model [10] a concept interpretation I corresponds to a competitive model on an agent learning tree. Signatured agent trees satisfy goals to complete a model diagram realizing a role R. Concept descriptions are presented with an agent signature tree $T_{I\Sigma}$ with a role R defined on the signature agents.

Proposition 6.1 A description algebra morphism on free signature tree $\wp(T_{I\Sigma})$such that roles are preserved on $T_{I\Sigma}$ algebra is definable by algebraic extension on an agent signature algebra $T_{I\Sigma}$.

Let us present the agent competitive instance for an algebraic description model platform.

Definition 6.2 Let A and B be description algebras with intelligent signature $I\Sigma$ containing agents. An $I\Sigma$-ontology description is an $I\Sigma$ description algebra with a prescribed role R: $X \rightarrow \wp(I\Sigma)$ for the agents and functions on the $I\Sigma$ signature.

Remark: $X \subseteq I\Sigma$, so for a set monad, there is an assignment for all $I\Sigma$ well-formed trees. Example well-formed agent trees were presented in the first auhtors publications around 2007 on ISL algebras with 1-1 signature trees.

Theorem 6.1 Let A and B be $I\Sigma$ description algebras with the signature $I\Sigma$. Then the agent homomorphisms defined from A to B preserve $I\Sigma$ -ontology iff defined by a description algebra homorphisms by algebraic extension on free signature tree $\wp T_{I\Sigma}$ such that roles are preserved on $T_{I\Sigma.}$

Proof Theorems 5.1, 5.2, and Proposition 6.1.

Theorem 6.2 Let A and B be $I\Sigma$ -description algebras with the signature $I\Sigma$ containing KD agents. The AII with KD morphisms preserve $I\Sigma$-description ontoltogy algerbas iff defined by KD-Description ontology homorphisms.

Proof Similar to 6.1. DKB mappings are specific ATL's were the ontology algebra operations are the same at source and target. We can prove based on the above that DKB mappings are DKB preservation consistent.

VII. CONCLUDINGCOMMENTS

A sound computing basis for ontology structures descriptions, and preservation theorems are accomplished with ontology preserving functions and morphisms that are applied to transform learning across domains. Competitive learing models based on a novel multi-agent have increasing important applications ranging from structural learning to predictive data analytics based on goal plans. Roles and description are developed with new algebraic models with newer applications to concept description ontology algebras and description ontology preservation. The areas are a basis to future reseach on ontology structures with a comprehensive mathematical basis. Newer areas to explore are ATL principle for mutual robot learning based on ontology preservation morphisms.

ACKNOWLEDEGMENTS

We thank our colleague Prof. Dominik Slezak, Warsaw University for his comments on the presentation.

* Sequnent Description Logic computing was developed at Computation logic Lab., Burnaby, Canada: Akdmkrd.tripod.com

References

[1] Nourani, C. F. 1996, Slalom Tree Computing – A Computing Theory For Artificial Intelligence, June 1994 (Revised December 1994), A.I. Communication Volume 9, Number 4, December 1996, IOS Press, Amsterdam.

[2] ADJ-Goguen, J.A., J.W. Thatcher, E.G. Wagner and J.B. Wright, A Junction Between Computer Science and Category Theory (parts I and II), IBM T.J. Watson Research Center, Yorktown Heights, N.Y . Research Report, 1975. 350.

[3] Nourani,C. F. 2009, A Descriptive Computing, Information Forum, Liepzig, Germany, March 2009. SIWN2009 Program, 2009. The Foresight Academy of Technology Press International Transactions on Systems Science and Applications, Vol. 5, No. 1, June 2009, pp. 6069. M. Wooldridge and N.R. Jennings, Intelligent Agents. (1993) 51- 92.

[4] Koehler, J. 1986, Planning From Second Principles, AI 87.

[5] Nourani, C. F. 1991, Planning and Plausible Reasoning in Artificial Intelligence, Diagrams, Planning, and Reasoning, Proc. Scandinavian Conference on Artificial Intelligence, Denmark, May 1991, IOS Press.

[6] Brazier, F.M.T. Dunin-Keplicz, B., Jennings, N.R. and Treur, J. (1997) DESIRE: modelling multi-agent systems in a compositional formal framework, International Journal of Cooperative Information Systems, M. Huhns, M. Singh, (Eds.), special issue on Formal Methods in Cooperative Information Systems, vol. 1. Knowledge-based Systems workshop, KAW'95, Calgary: SRDG Publications, Department of Computer Science.

[7] Nourani,C. F. 1995, Intelligent Languages - A Preliminary Syntactic Theory, May 15, 1995, Mathematical Foundations of Computer Science;1998, 23rd International Symposium, Brno, Czech Republic, August The satellite workshop on Grammar systems. Silesian University, Faculty of Philosophy and Sciences, Institute of Computer Science, Science;1450, Springer, 1998, ISBN 3-540- 64827-5, 846 pages.

[8] Nourani, C. F. 2005, A Haptic Computing Logic – Agent Planning, Models, and Virtual Trees, 286-311., Affective And Emotional Aspects Of Human-Computer Interaction: Game-Based and Innovative Learning Approaches, Edited by Maja PIVEC, IOS PRES, 2006, pp. 317, ISBN1-58603-572-X.

[9] Nourani, C. F. and T. Hoppe 1994, "GF-Diagrams for Models and Free Proof Trees," Proceedings the Berlin Logic Colloquium, Universitat Potsdam, Organized by Humboldt Universtitat Mathematics, Berlin. May 1994.

[10] Cyrus F. Nourani and Oliver Schulte, Multiagent Decision Trees, Competitive Models, and Goal Satisfiability. DICTAP, Ostrava, Czech Republic, July 2013.

[11] Genesereth, M. R. and N. J. Nilsson 1987, Logical Foundations ofArtificial Intelligence, Morgan-Kaufmann,1987.

[12] Nourani, C. F. 2005, Agent-based Structures, Agent Ontology Preservation and Enterprise Modeling Workshop on Ontologies in Agent Systems 5th International Conference on Autonomous Agents Montreal, Canada.

[13] U. Straccia, A fuzzy description logic, in: J. Mostow, C. Rich (Eds.), AAAI/IAAI, AAAI Press / The MIT Press, 1998, 594-599.

[14] Nourani. C. F. "Design with Software Agents, Parallel Module Coordination and Object Languages, February 3, 1997. TU Berlin, Fachbereich 13 - Informatik, Sekretariat FR5-13, Berlin, Germany.

[15] Nourani, C. F. "Abstract Implementation Techniques for A. I. By Computing Agents,: A Conceptual Overview," Technical Report, March 3, 1993, Proceedings SERF-93, Orlando, Florida, November 1993. Published by the University of West Florida Software Engineering Research Forum, Melbourne, Florida.

[16] Gio Wiederhold: ``Interoperation, Mediation and Ontologies''; Proc.Int.Symp. on Fifth Generation Comp Systems, ICOT, Tokyo, Japan, V ol.W3, Dec.1994, pages 33-48.

[17] Eklund, P., Galán, M. A., Kortelainen, J., Ojeda-Aciego, M. (2014). Monadic formal concept analysis, RSCTC 2014, (Eds. C. Cornelis et al.), Lecture Notes in Artificial Intelligence 8536, 201-210. pp. 473-484.

[18] Eklund, P., Galán, M.A., Gahler, W.: Partially ordered monads for monadic topologies, Kleene algebras and rough sets. Electronic Notes in Theoretical Computer Science 225(5), 67–81 (2009).

[19] Rao, A. S. and Georgeff, M.P. (1991). Modeling rational agents within a BID-architecture. In: R. Fikes and E. Sandewall (eds.), Proceedings of the Second Conference on Knowledge Representation and Reasoning, Morgan Kaufman.

[20] Sebastian Thrun - Artificial Intelligence, 1998 – Elsevier, Learning metric-topological maps for indoor mobile robot navigation Pittsburgh, PA 15213, USA Received June 1996; revised October 1997.

[21] R A Brooks, M J Mataric 1993- Robot learning, 1993 – Springer

[22] Dunin-Keplicz, B. and Treur, J. (1995). Compositional formal specification of multi-agent systems. In: M. Wooldridge and N.R. Jennings, Intelligent Agents, Lecture Notes in Artificial Intelligence, Vol. 890, Springer Verlag, Berlin, pp. 102-117.

[23] Nilsson, N. J. 1969,"Searching, problem solving, and game-playing trees for minimal cost solutions." In A.J. Morell (Ed.) IFIP 1968 Vol. 2, Amsterdam, North-Holland, 1556-1562, 1969.

[24] M. Schmidt-Schauß, G. Smolka, Attributive concept descriptions with complements, Artificial Intelligence 48.

[25] Giancarlo Guizzardi , 2005, Ontological Foundations for Structural Conceptual Models. CTIT, UTwenty, The Netherlands, PhD Thesis Series, No. 05-74 Telematica Instituut No. 015 (TI/FRS/015).