Semantic Resolution for E-Commerce

Yun Peng, Youyong Zou, and Xiaocheng Luan

Department of Computer Science and Electrical Engineering
University of Maryland Baltimore County (UMBC)
Baltimore, MD 21250, USA
{ypeng, yzou1, xluan1@cs.umbc.edu}

Nenad Ivezic, Michael Gruninger, Albert Jones

National Institute of Standards and Technology (NIST)
100 Bureau Drive, Gaithersburg, MD 20899, USA
{nivezic, gruning, ajones}@nist.gov

Abstract. We describe a research project on resolving semantic differences for multi-agent systems (MAS) in electronic commerce. The approach can be characterized as follows: (1) agents in a MAS may have their own specific ontologies defined on top of a shared base ontology; (2) concepts in these ontologies are represented as frame-like structures based on DAML+OIL language; (3) the semantic differences between agents are resolved at runtime through inter-agent communication; and (4) the resolution is viewed as an abductive inference process, and thus necessarily involves approximation reasoning.

1 Introduction

Understanding the meaning of messages exchanged between software agents has long been recognized as a key challenge to interoperable multi-agent systems (MAS). Forcing all agents to use a common vocabulary defined in shared ontologies is an oversimplified solution when agents are designed independently. This is the case for agent applications in E-Commerce which (1) is a huge, ***open*** marketplace accommodating many companies capable of entering and leaving the market freely; (2) involves ***dynamic*** partnerships which are formed and dissolved easily and frequently; and (3) contains ***heterogeneous*** representations of agents for different enterprises [4]. It is, therefore, impractical to restrict all agents to use the same vocabulary or to require the availability of inter-ontology translation services prior to the deployment of the agent systems. Semantic differences between individual agents in the system should be

allowed and be resolved when they arise during agent interaction. These points are captured by the following assumptions, which are similar to those made in [1, 20]:

1. Interacting agents share one or more base ontologies;
2. Agents use different ontologies defined on top of the base ontology; and
3. Runtime, semantic resolution is unavoidable.

Assumption 1 is reasonable because it is hard to imagine heterogeneous agents built in a total vacuum – at least some shared vocabulary and understanding of that vocabulary should be assumed. The base ontology can be viewed as an ontology for a community, it defines general terms shared by members of that community, and should be relatively stable (any change must be based on a community-wide consensus). It can be defined either in some agreed-upon ontology specification languages (e.g., Ontolingua [7] or DAML+OIL) or in some other forms (e.g., WordNet, a natural language-based taxonomy, as in work in [1]). Assumption 2 allows each agent to develop its own specialized vocabulary, reflecting its particular needs or perspectives. Usually, the agent-specific ontologies are changed more frequently than the base ontology. Since these ontologies are defined on top of the base ontologies, they are also called *differentiated ontologies* in the literature [20].

Research work on ontology engineering attempts, in part, to provide semantics for information exchanged over the Internet [5, 6, 12]. The most noticeable, recent development in this direction is the *Semantic Web* effort jointly launched by W3C [2, 16], the DARPA Agent Markup Language Project [5], and EU's Information Society Technologies Program (IST) [12]. One result from this effort is the set of DAML+OIL specifications, a language for ontology definition, manipulation, and reasoning [5]. Although the technologies developed in this effort are aimed at making Web pages understandable by programs, they may serve, we believe, as a basis for resolving semantic differences between heterogeneous agents. However, additional methodology and mechanisms need to be developed if semantic resolution is to be done at runtime through agent interaction. This is the primary objective of our project, which is performed jointly by The Laboratory for Advanced Information Technology at UMBC and the Manufacturing Systems Integration Division at NIST.

The rest of this paper is organized as follows. Section 2 further motivates our approach for semantic resolution with a simple E-Commerce scenario of buying and selling computers over the internet; Section 3 describes how the base and agent-specific ontologies are defined using DAML+OIL language; Section 4 defines the two basic operations needed for our semantic resolution approach; Section 5 presents an agent communication protocol; and Section 6 outlines several approximate algorithms for semantic mapping. Section 7 concludes the paper with directions of future research.

2 A Simple E-Commerce Scenario

Consider the following simple, E-Commerce scenario of **RFQ** (Request For Quote) involving two agents: the buyer A1 representing a whosaler of computers and the seller A2 representing a computer manufacturer. Both A1 and A2 share a common

ontology ONT-0, which gives semantics of some basic terms that describe business transactions such as RFQ and generic names for computer systems and components such as notebooks, CPU, and memory. Each of the two agents has its own specialized ontology. ONT-1 defines semantics of products to order for A1, organized to meet the intended usage of its customers. ONT-2 defines items in the product catalog for A2, based on technical specifications of manufactured computer systems.

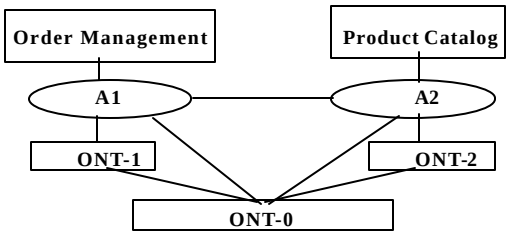

Figure 1. A simple RFQ scenario involving two agents

Suppose A1 sends an RFQ to A2 for a number of "PC_for_Gamers", a term defined in ONT-1. Before A2 can determine a quote, it needs to understand what A1 means by this term and if a semantically similar term is in its catalog as defined in ONT-2. We use phrase "***Semantic Resolution***" for the process of identifying the meaning of terms defined in different ontologies and, if possible, matching these terms semantically.

3 Ontology Design and Representation

The bulk of the base ontology ONT-0 is devoted to define the common terms for computer systems and their components. Here we adopt part of the classification of UNSPSC (*Universal Standard Products and Services Classification Code* by United Nations Development Program and Dun & Bradstreet) [19], and organize these terms as a taxonomy. For example, "notebook-computers" is defined as a subclass of "computers", which is in turn defined a subclass of "Hardware", etc. Common terms used in RFQ such as price, weight, size, date, are also defined in ONT-0.

Agent specific ontologies ONT-1 and ONT-2 define terms that reflect different views of A1 and A2, respectively, of computer systems. As a computer retailer, A1 names their computer systems according to different usage of these computers by its customer, e.g., "PC for Gamers", PC for Family", "PC for Students", etc. On the other hand, A2, as a computer manufacturer, organizes its catalog of products according to their technical and configuration specifications, e.g., "Entry Level", Professional Level", "Portable", etc. ONT-1 and ONT-2 organize their respective terms into taxonomies. In addition, each term is also given a set of properties. Therefore, each term is defined by the set of its superclasses in the taxonomy and its properties. Also note that, these two agent specific ontologies are defined on top of the base ontology

ONT-0, this can be seen in the example in Figure 2 where the term "PC for Gamers" in ONT-1 is defined in part by terms from ONT-0.

Although some researchers have used full first-order logic for ontology representation (see Ontolingua [7]), the current trend has been to use description logics (DL) of different flavors [5, 6, 17 - 20]. DAML+OIL can be seen as a combination of DL and web standards such as RDF, RDF Schema [10], and XML. One of its useful features is the use of namespaces to reference individual ontologies. We use ns0, ns1, and ns2 as namespaces for the three ontologies ONT-0, ONT-1, and ONT-2 in the above E-Commerce scenario.

The following is an example of an XML-encoded DAML+OIL definition of a class of "PC_for_Gamers" in ONT-1. Symbols starting with "#" are terms defined in the home ontology ONT-1, whose namespace ns1 is omitted, and prefix symbols "daml" and "rdfs" denote namespaces for DAML and RDF Schema specification, their URIs (e.g., xmlns:daml = http://www.daml.org/2001/03/daml+oil#) are given as part of XML Schema at the beginning of the ontology definition.

In essence, this definition says that the concept of "PC_for_Gamers" is a sub-class of "Computers-to-order" in ONT-1 and sub-class of "Workstations, desktop-computers" defined in ONT-0, with "good video card", "good sound card", and "fast CPU", the meanings of these terms are also defined in ONT-1, the home ontology and ONT-0, the base ontology.

```
<daml:Class rdf:ID="PC_for_Gamers ">
   <rdfs:subClassOf rdf:resource="#Computers-to-order"/>
   <rdfs:subClassOf rdf:resource="
      ns0: Workstations, desktop-computers "/>
   <rdfs:subClassOf>
      <daml:Restriction>
         <daml:onProperty rdf:resource="ns0:hasVideoCard"/>
            <daml:hasValue rdf:resource="#GoodVideoCard "/>
      </daml:Restriction>
   </rdfs:subClassOf>
   <rdfs:subClassOf>
      <daml:Restriction>
         <daml:onProperty rdf:resource="ns0:hasSoundcard"/>
         <daml:hasValue rdf:resource="#GoodSoundcard"/>
      </daml:Restriction>
   </rdfs:subClassOf>
   <rdfs:subClassOf>
      <daml:Restriction>
         <daml:onProperty rdf:resource="ns0:hasCPU"/>
         <daml:hasValue rdf:resource="#FastCPU "/>
      </daml:Restriction>
   </rdfs:subClassOf>
</daml:Class>
```

Figure 2. An example ONT-1 class defined in DAML-OIL.

4 Operations for Semantic Resolution

Our approach to semantic resolution is motivated by the way humans resolve their semantic differences. When two people engage in a conversation and one does not understand a term mentioned by the other, the listener would *ask* the other to clarify or explain the meaning of the term. The other person would try to answer it by define the term in terms she thinks the listener would understand. If the answer is not understood, more questions may follow. This process may continue until the term in question is completely understood (either the term is mapped to one the listener is familiar with or a new term with clear semantics is learned) or the listener gives it up. The listener can understand a foreign term because the two people share the meanings of some common terms, which we attempt to model by the base ontology in our approach. The process of achieving semantic resolution here involves two basic operations, *Semantic Querying*, which gradually reveals the definition of the foreign term in the terms of the base ontology, and *Semantic Mapping*, in which the definition of the foreign term is mapped to a term in the listener's ontology. Each of these two operaions has its own research issues. We briefly describe each in the following subsections, and address technical issues involved in the subsequent sections.

Semantic Querying. Following the example in the simple E-Commerce scenario, since A2 only understands ONT-0 and ONT-2, it does not understand the term such as ns1:*PC_for_Gamers* in the RFQ from A1 defined in ONT-1. Similar to a conversation of two strangers, A2 would ask what A1 means by this term via some agent communication language. We call this process of obtaining the description of a term from a different ontology *Semantic Querying*, and the two agent-specific ontologies ONT-1 and ONT-2 in our example are called the *source* and *target* ontologies. The description of a source term includes both slot name and filler name of each slot in its definition in the source ontology. In our example, the first semantic query to A1 gives A2 the following information (with proper namespace designations).

ns1:PC_for_Gamers
List of primitive super-classes
- ns1: Computers-to-order
- ns0:Workstations, desktop-computers

List of properties
- ns0:HasGraphics_card = ns1:GoodGraphicCard
- ns0:HasSound_card = ns1:GoodSoundCard
- ns0:HasCPU= ns1:FastCPU
- ns0:Memory=ns1:BigMemory

Additional queries on ns1 terms in the above description gives

ns1:PC_for_Gamers
List of primitive super-classes
- ns1: Computers-to-order
- ns0:Workstations, desktop-computers
- ns0:Computers

List of properties

- ns0:HasGraphics_card = (ns0:size >= 1000)
- ns0:HasSound_card = (ns0:size >= 24)
- ns0:HasCPU = (ns0:size >= 1000)
- ns0:Memory = (ns0:size >= 256).

This can be viewed as an extended normal form of the given ONT-1 concept with respect to ONT-0[1].

Semantic Mapping. The extended normal form of ns1:PC_for_Gamers from the semantic querying step provides much information to A2. However, for A2 to truly understand this concept, it needs to *map* or *re-classify* this description into one or more concepts defined in its own ontology ONT-2. This is accomplished by the *Semantic Mapping* step. Note that due to the structural differences, concepts from different ontologies are likely to match each other only partially.

Semantic resolution is thus similar to abductive reasoning process, semantic querying corresponding to evidence collection, and semantic mapping to hypothesis generation. All partially matched target concepts are considered candidate or hypothesized maps of the source concept, each of which can explain the source concept to different degrees based on the base ontology. If the best candidate is satisfactory, then a quote is generated by A2 and sent to A1. Otherwise, additional steps of inter-agent interactions may be taken. For example, if the best candidate, although unsatisfactory, is sufficiently better than all others, then its description is sent back to A1 for confirmation. If the first few leading candidates have similar level of satisfaction, then questions that discriminate some candidates over others will be sent to A1. The details of the algorithms are described in Section 6.

5 Communication Protocol for Semantic Resolution

To support agent communication for both semantic querying and semantic mapping, we need to have (1) an agent communication language (ACL) to encode messages, (2) a content language to encode the content of a message, and (3) a communication protocol that specifies how these messages can be used for meaningful conversations. For reasons including clearly defined semantics and standardization support, we have selected FIPA ACL [9] as the ACL for our project. We choose DAML+OIL as the content language because it is also the language for ontology specification. The most relevant work to date on developing agent communication protocols for semantic resolution between different ontologies can be found in [1]. Their Ontology Negotia-

[1] In description logics, a normal (or canonical) form of a concept *C* consists of two lists: a list of all of *C*'s primitive super-classes and a list of all of *C*'s properties, including those inherited from its super-classes. These two lists are called *P* list and *R* list in this paper.

tion Protocol is an extension of KQML [8] with additional performatives, such as *Request Clarification*, *Clarification*, *Interpretation*, *Confirmation*, etc.

Our *Semantic Resolution Protocol* combines our earlier work [4] and the work in [1]. The design follows FIPA Interaction Protocol convention, which requires the definitions of (1) the acts involved in interaction processes, (2) the roles played by the actors in interaction processes, and (3) the phase transitions of the interaction process. There are two players in our protocol (it may be easily extended to involving multiple players), the buyer (A1) and the seller (A2). The buyer plays the role of *the initiator* while the seller is the *participant*. Performatives used in the protocol represent the communicative acts intended by the players. The following FIPA performatives are selected for the protocol.

- **call-for-proposal** (CFP): the action of calling for proposals to perform a given action. This is used by buyer to ask the seller to propose a quote for a RFQ.
- **propose**: the action of submitting a proposal to perform a certain action, given certain preconditions. This is used to turn a proposed quote.
- **accept-proposal**: the action of accepting a previously submitted proposal to perform an action.
- **reject-proposal**: the action of rejecting a submitted proposal to perform an action
- **terminate**: the action to finish the interaction process.
- **inform**: the action of informing that certain propositions are believed true.
- **not-understood**: the action of informing the other party that its message was not understood. This is used by the seller to request the buyer to send the description of a term it does not understand in the previous message.
- **query-if**: The action of asking another agent whether or not a given proposition is true. This is used by the seller in semantic mapping to ask the buyer to confirm if a candidate concept is an acceptable match for the given source concept.
- **confirm**: the action of confirming that given propositions are believed to be true. This is used by the buyer to confirm a target concept received in the incoming "query-if" message from the seller.
- **disconfirm**: the action of informing that given propositions are believed false

The first 5 performatives are for RFQ; the rest are for semantic querying and mapping. (See [9] for a detailed description of these performatives.) The phase transitions in the protocol are given in the message-flow diagram in Figure 3.

6 Algorithms for Semantic Mapping

The objective of semantic resolution is to find a concept in the target ontology whose description best matches the description of a given concept defined in the source ontology. Because agent-specific ontologies often have different structures and use different concept names, concept matching is seldom exact. Partial matches, which can occur even if a single ontology is involved, become more prevalent when different

agent-specific ontologies are involved. Consequently, the simple techniques used in DL for partial matches (e.g., most general subsumees and most specific subsumer)

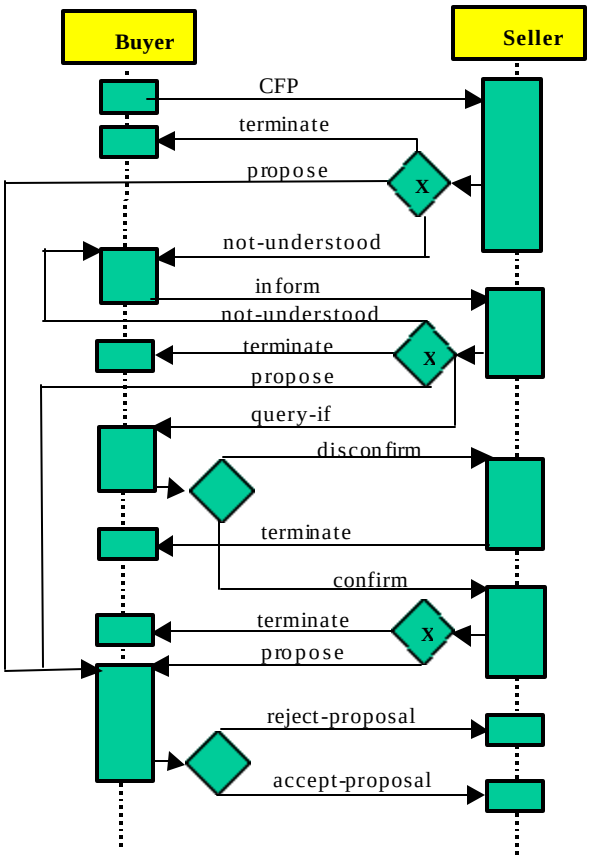

Fig. 3. State transition diagram of the Semantic Resolution Protocol

are no longer adequate. Approximate reasoning that at least gives a ranking for all partially matched target concepts is required. Commonly used approximate reasoning techniques include *rough set theory* [18], fuzzy *set theory* [15], and *probabilistic classification* [13, 15]. In many applications, these more formal approaches may not work, either because the assumptions made for them cannot be met or the information needed is not available. Heuristic approximation becomes necessary [18].

In this section, we focus on heuristic methods for approximating partial matches. The main algorithm *subsumption*(*A, B, theta*) is an extension of the structural comparison for subsumption operation in DL. It returns a numeric score, theta, in [0, 1] that quantifies the degree that concept A subsumes concept B. In DL, A subsumes B if and only if every object in A is also an object in B. A structural comparison approach [3, 11] works with normal forms of concepts, which include a list of all primitive super-

classes *S* and a list of all properties *P* for a concept, and requires that (1) *Sa* is a subset of *Sb*, and (2) constraints on *Pb* is compatible with (i.e., is at least as strict as) that of *Pa*. These requirements cannot be established logically if the normal forms of A and B involve terms from different ontologies. This can be seen by comparing the extended normal form of ns1:PC_for_Gamers in Section 4 obtained via semantic querying operation and the extended normal form for ns2:Professional_Use_Desktop given below. Besides ns0 terms, these two normal forms contain ns1 and ns2 terms from ONT-1 and ONT-2, respectively.

ns2:Professional _Use_Desktop

List of primitive super-classes

- ns2:Desktop
- ns0:Workstations, desktop-computers
- ns2:Copmuter_Systems
- ns0:Computers

List of properties

- ProductName = "xxx4"
- ProductNumber = "yyy4"
- ns0:HasSound_card = (ns0:size = 24)
- ns0:HasCPU = (ns0:size = 1800)
- ns0:Memory = (ns0:size = 512)
- ns0:Price = (ns0:size = 2300)
- ns2:HasColorMonitor = subproperty(ns0: HasMonitor ns0:size = 19)

One may suggest that we ignore all of these ns1 and ns2 terms and conduct the subsumption operation based solely on those ns0 terms. However, doing so would overlook the important information on the structural differences. Moreover, it is generally believed that if two concepts are far apart in structure, they are less likely to match each other, even if they agree well on terms of the base ontology. In what follows we describe the methods to compute a measure to compare two concepts' *P* and *R* lists and the method to combine them into a single score.

Comparing the superclass lists *Sa* and *Sb*. The objective of this comparison is to obtain a measure for the degree that *Sa* is a subset of *Sb*. First, we check if any member Sa_i in *Sa* is logically inconsistent with any member Sb_j in *Sb*, e.g., if (**and** Sa_i Sb_j) is unsatisfiable. One type of inconsistency would be that Sa_i and Sb_j are disjoint. For example, as defined in ONTO_0, "ns0:Notebook-computers", "ns0:Workstations, desktop-computers", and "ns0:Servers" are disjoint with each other. If inconsistency is detected, then A cannot subsume B. Otherwise, we proceed to compute a heuristic measure of the degree that *Sa* is a subset of *Sb* (e.g., the degree that A subsumes B in terms of their respective super classes).

$$\textit{inclusion\_measure}\ (\textit{Sa}, \textit{Sb}) = \begin{cases} -1 & \textit{if Sa and Sb are inconsistent} \\ |\textit{Sa} \cap \textit{Sb}| / |\textit{Sa}| & \textit{otherwise} \end{cases}$$

If this measure is –1, then the entire matching process stops (no comparison of properties will be performed), and returns –1, meaning that A cannot subsume B.

This measure is 1 when *Sa* is a subset of *Sb*, 0 if none of the members of *Sa* is also a member of *Sb*. One benefit of this heuristic rule is that is can be viewed as the conditional probability *Pr*(*x* in *Sb* | *x* in *Sa*) when members of *Sa* and *Sb* are treated as sample points from the same space. This allows us to generalize the measure with more sophisticated probabilistic computation when the interdependency of these members are known.

Applying this rule to our example of ns1:PC_for_Gamers and ns2:Professional_Use_Desktop, we have the inclusion measure of 2/3 because 2 of the 3 members in superclass list of the former are members of superclass list of the latter.

Comparing the property lists *Pa* and *Pb*. This comparison is done in two steps.

Step 1: Identify all matching pairs between *Pa_i* in *Pa* and *Pb_j* in *Pb*. *Pa_i* matches *Pb_j* if 1) they have the identical property name, including the name space, or 2) *Pb_j* is a sub-property of *Pa_i* or vise versa. For any *Pa_i* in *Pa* that does not pair with any member of *Pb*, then a measure of –1 is given for that *Pa_i*.

Step 2: Compute compatibility measure for each matching pair Pa_i in Pa and Pb_j in Pb. If their constraints (i.e., cardinalities and value ranges) are incompatible (i.e., the logical expressions of their constraints are not satisfiable simultaneously). If incompatibility is detected, a measure of –1 is given to that Pa_i. Otherwise (i.e., they are compatible), use some heuristic rule to compute a (positive) measure for that pair. This is summarized by the following rule.

$$\textbf{\textit{compatibilty}} = \begin{cases} -1 & \textbf{\textit{if Pa\_i does not have a a match in Pb}} \\ -1 & \textbf{\textit{if range\_i}} \cap \textbf{\textit{range\_j}} = \varnothing \\ 1 & \textbf{\textit{if range\_j}} \subseteq \textbf{\textit{range\_i}} \\ \boldsymbol{\alpha}_{ij} & \textbf{\textit{otherwise}} \end{cases}$$

where $\boldsymbol{\alpha}_{ij}$ is the overlapping ratio between *range_i* and *range_j*, which can be computed by additional rules that handle different types of value ranges such as close intervals, open intervals, and intervals involving infinities.

Applying this rule to our example, we have one –1 measure (for GraphicCard) and three 1 measures (for all other proerties).

Combining comparison results. When inclusion_measure retunes a positive value, then this value and all measures of property comparisons (some may be positive and some may be negative) are combined to generate an overall score. Here we use rules similar to those given for certainty factors in MYCIN. First, each measure is given a weight $\boldsymbol{w}_i$, reflecting the importance that property is for establishing subsumption relation for concept A. Recall that each measure is for one property (plus one more for superclass) of A. Therefore, there are total of |*Pa*| + 1 weights. When such weights are not provided by the designer of the ontology of A, we use 1/(|*Pa*| + 1) as the default weight for each of them. Then the combination takes the following steps

Step 1: Combine all positive measures as $\boldsymbol{C}1 = 1 - \Pi_i (1 - \boldsymbol{w}_i \cdot \textbf{\textit{measure}}_i)$, and combine all negative measures as $\boldsymbol{C}2 = \Pi_i (1 + \boldsymbol{w}_j \cdot \textbf{\textit{measure}}_j) - 1$.

Step 2: Combine C1 and C2 as $\boldsymbol{C} = (\boldsymbol{C}1 + \boldsymbol{C}2)/(1 - \min\{|\boldsymbol{C}1|, |\boldsymbol{C}2|\})$

Step 3: Finally, normalize *C* by the weights as $\boldsymbol{CN} = \boldsymbol{C}/(1 - \Pi_i(1 - \boldsymbol{w}_i))$, where, *i* is over all |Pa| + 1 weights. *CN* is then returned as *theta*, the final score of A subsuming B. The rationale for normalization is that when all measure are +1 then *CN* = 1, and when all measures are –1 then *CN* = –1.

Applying this rule to our example, we have five measures (2/3, -1, 1, 1, 1), each with a weight 1/5. This yields *C1* = 0.55626, *C2* = -0.2, *C* = 0.4457, and the overall score *CN* = 0.6629.

Search for the plausible subsumees. The semantic resolution seeks a most plausible target concept B that either approximately subsumes or is subsumed by A, as measured by the heuristic score theta. Finding the most plausible subsumee can be done by a depth-first search plus backtracking or more efficiently by a best-first style search of the target ontology graph. Candidate target concepts are normalized when they are generated during the search.

7 Conclusions

The work presented in this paper represents the first step of our ongoing effort toward a comprehensive solution to the problem of semantic resolution. Many issues, both practical and theoretical, remain to be addressed. To answer some of them, we will continue our project along the following directions. First, we plan to build a prototype agent system based on the approach outlined in this paper. This system will be used as a testbed to validate the methods we develop and to test emerging tools and approaches. It can also serve as a bridge connecting the research community and the industry by incorporating ontologies of real-world enterprises engaged in E-Commerce activities. Second, we plan to develop a more formal treatment for approximating semantic mapping with partially matched concepts. One approach is to incorporate probability theory, in particular the Bayesian belief network [13, 14], into the ontology class hierarchies. Finally, we plan to extend the semantic resolution process to become a cycle of *hypothesize-and-test*, as with most abductive, evidential reasoning systems. Instead of separating semantic querying and mapping as two steps, they will be interwoven together so that additional evidence will be collected only when it is needed, and the hypothesized mappings are refined and discriminated against each other with each new evidence until the solution is gradually emerged [14].

Disclaimer

References

1. Bailin, S.C. and Truszkowski,W., Ontology Negotiation between agents supporting intelligent information management, *Workshop on Ontologies in Agent Systems*, 2001.
2. Berners-Lee, T. What the Semantic Web can represent, http://www.w3.org/DesignIssues/RDFnot.html, 1998.
3. Borgida, A., *et al*. CLASSIC: A Structural Data Model for Objects. In *Proc of ACM SIGMOD Intern'l Conference on Management of Data*, Portland, OR, June 1989, 59-67.
4. Chen, Y., *et al*. A Negotiation-Based Multi-agent System for Supply Chain Management. In *Proc. of The International Conference on Autonomous Agents*, Seattle, May, 1999..
5. DAML home page, http://www.daml.org/.
6. Decker, S. et al, Knowledge representation on the Web. *Proceedings of the 2000 International Workshop on Description Logics*.
7. Farquhar, A., Fikes, R., and Rice, J. The Ontolingua server: a tool for collaborative ontology construction. *Intl. J. Of Human-Computer Studies* **46**(6):707-727, 1997.
8. Finin, T., Labrou, Y., and Mayfield, J. KQML as an Agent Communication Language, in Jeff Bradshaw (ed.), *Software Agents*, MIT Press, Cambridge, MA, 1997.
9. FIPA (The Foundation of Intelligent Physical Agents) home page, http://www.fipa.org.
10. Lassila, O. and Swick, R. Resource Description Framework (RDF) Model and Syntax Specification, http://www.w3.org/TR/1999/REC-rdf-syntax-19990222/, Feb, 1999.
11. MacGregor, R.M. The Evolving Technology of Classification-based Knowledge Representation Systems. In *Principles of Semantic Networks: Explorations in the Representation of Knowledge*, J. Sowa (ed.), Morgan Kaufmann, 1991.
12. OIL (Ontology Inference Layer) home page, http://www.ontoknowledge.org/oil/.
13. Pearl, J. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*, Morgan Kaufman, San Mateo, CA, 1988.
14. Peng Y and Reggia J: *Abductive Inference Models for Diagnostic Problem Solving*, Springer-Verlag, New York, NY,1990.
15. Russell, S. and Norvig, P. *Artificial Intelligence, A Modern Approach*. Prentice Hall, Englewood Cliffs, NJ, 1995.
16. Semantic Web home page, http://www.w3.org/2001/sw/.
17. Stuckenschmidt, H., Using OIL for semantic information Integration. *Proceedings of the ECAI workshop on Ontologies and PSMs 2000*
18. Stuckenschmidt,, H. and Visser. U., Semantic translation based on approximate reclassification. *Proceedings of the Workshop "Semantic Approximation, Granularity and Vagueness, KR'00*
19. UNSPSC, home page, http://www.unspsc.org.
20. Weinstein, P. and Birmingham, W.P., Comparing concepts in differentiated ontologies. Proceedings of the 12th Workshop on Knowledge Acquisition, Modeling and Management (KAW'99).