Local Consensus Ontologies for B2B-Oriented Service Composition

Andrew Williams
University of Iowa
5015 Seamans Center
Iowa City, IA 52242
(319) 384-0833
abwill@engineering.uiowa.edu

Anand Padmanabhan
University of Iowa
14 MacLean Hall
Iowa City, IA 52242
(319) 335-3650
apadmana@cs.uiowa.edu

M. Brian Blake
Georgetown University
234 Reiss Science Building
Washington, DC 20057
(202) 687-3084
blakeb@cs.georgetown.edu

ABSTRACT

Agents seeking to discover and compose needed Web services may face knowledge sharing interoperability problems due to differing ontologies. In practice, agents may not have a global consensus ontology that will facilitate knowledge sharing and integration of required services. We investigate a method for agents to develop local consensus ontologies to aid in the communication within a multi-agent system of business-to-business (B2B) agents. We compare variations of syntactic and semantic similarity matching to form local consensus ontologies with and without the use of a lexical database.

Categories and Subject Descriptors

I.2.4 [**Computing Methodologies**] Distributed Artificial Intelligence–*multiagent systems.*

General Terms

Algorithms, Experimentation.

Keywords

Ontologies in agent-based information systems and knowledge management; agent- mediated electronic commerce.

1. INTRODUCTION

Automated cross-organizational service composition has become more of a reality with the maturity and acceptance of *web services* [5]. The composition of web services has opened new possibilities for business-to-business (B2B) electronic commerce [9]. A major barrier to automation in this domain is understanding the commonalities and differences of services that have similar functionality. One method of categorizing services is by examining their message flow. An emerging protocol in web services is the use of the Simple Object Access Protocol (SOAP) [8] to specify messages passed to and from web services that are defined by the Web Service Description Language (WSDL) [7]. Both SOAP and WSDL have Extensible Markup Language (XML)-based notations. A technique in this work is the ontological representation of these SOAP messages. Therefore, this categorization that takes place in web services discovery and composition can be related to area of knowledge interoperability, or ontology issues.

We propose the use of B2B agents [10] to manage the automated composition of required Web services stored in registries, such as the Universal Description, Discovery and Integration (UDDI) framework [17]. A first step in this composition is for the agents to be able to find related services (i.e. ontology concepts) between inter-organization ontologies and intra-organization ontologies. There are various approaches to dealing with these challenges. The DARPA knowledge sharing format sought to restrict intelligent agents to using a common syntax, common ontology construction language, and common communication protocol [20]. This approach depends on enforcing the agents to using a common global ontology from the beginning. Another approach is to translate between ontologies using a platform independent language based on XML (e.g. DAML [21]) . This approach is useful if there are very large ontologies that need a thorough translation of the concepts and relationships between ontologies and the computational and time costs are not critical. Also, agents may wish to start with their individualized ontologies and collaboratively develop a global, consensus ontology. This differs from the first approach in that it recognizes that agents may not have a common, global ontology but may attempt to construct one through voting and other consensus methods. This method is useful if the time and effort can be coordinated and the benefit of forming a global, consensus ontology outweighs the costs involved.

Our approach acknowledges that agents may have diverse ontologies, but before extensive translation can take place, these agents must be able to relate services at a slightly higher level. Therefore, in this approach, agents create only relatively small local consensus ontologies to facilitate the discovery and understanding of web services for current and future B2B systems. This paper continues in Section 2 with a description of our approach to using ontologies for B2B-oriented web service discovery and composition methods. Section 3 describes our approach and current implementation. Section 4 how we evaluated our approach and compared it to an existing approach. Section 5 discusses related B2B work and ontology merging approaches. We conclude and discuss future work in Section 6.

2. Using Ontologies and Web Services for B2B Electronic Commerce

B2B commerce has long been associated with the transfer of messages, while the underlying services are initiated and executed through local human-driven means. A forward-looking prospect of web services technology is the seamless connection of these distributed services through automated means, such as agents. In this section, there is a brief discussion of the history of B2B systems and how web service composition provides a new aspect to this domain. Also, there is a discussion of how ontologies can be extracted from web service specifications.

2.1 Brief History of B2B Systems

Even before the acceptance of the Internet, corporations understood the benefit that electronically transmitted data could bring to transactions between businesses. This notion of electronically transmitted messages for such things as collaborating purchase orders and assisting order enactment and delivery led to the development of the first B2B systems over 25 years ago. These message transfer systems included B2B protocol standards, such as Electronic Document Interchange (EDI), over mediums called value-added networks (VANS). Over the past two decades, these systems have gained and acceptance and maturity where corporations or trading partners are able to collaborate electronically through messaging. However, these systems are known for their inflexibility. In addition, there are huge expenses associated with both the B2B protocol and mapping software and the usage fees of the proprietary VANs. The benefits and acceptance of the Internet and its capabilities has opened new possibilities to assist B2B interoperability. An emerging area for B2B commerce is the actual composition of their services as in cross-organizational workflow and ad-hoc supply chain management [3] [18][13].

2.2 Web Services and Ontologies

Middle agents have been used to serve as proxies for humans or machines. In earlier work [10], middle agents can represent multiple web services. If each of these middle agents, or *service proxy agents,* has ontological representations of the messages that their web services receive and transmit, then an external cross-organizational workflow/supply chain management agent for *service composition* has a measurable task in relating these ontologies when composing multiple services. An illustration of this multiple agent scenario is shown in Figure 2.1.

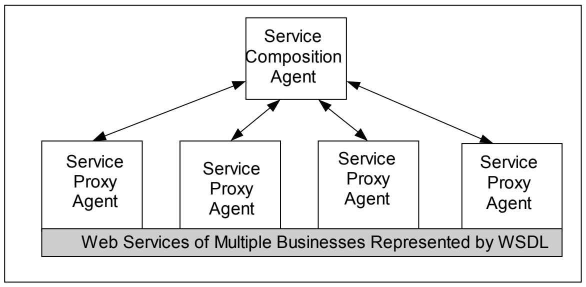

Service Composition Agent
Service Proxy Agent
Service Proxy Agent
Service Proxy Agent
Service Proxy Agent
Web Services of Multiple Businesses Represented by WSDL

Figure 2.1 Collaborative Agents for Web Service Composition

Each of the service proxy agents in Figure 2.2 help moderate and execute a subset of web services. These agents maintain the WSDL and SOAP documents of each of the web services that they proxy. Therefore, agents should be able to express their underlying services via ontological representations derived from WSDL and SOAP specifications.

The focus of this paper is not on the transformation of WSDL/SOAP to ontological notation, but on the aggregation of the multiple web service-based ontologies. Of significance, however, is what parts of specifications are captured as ontologies. Each web service can be specified with a WSDL document. In general terms this document contains service specific information, such as naming, location, and access protocols, in addition to pointers to input/output message (externally represented in SOAP). This general description of a WSDL document can be illustrated as in Figure 2.2.

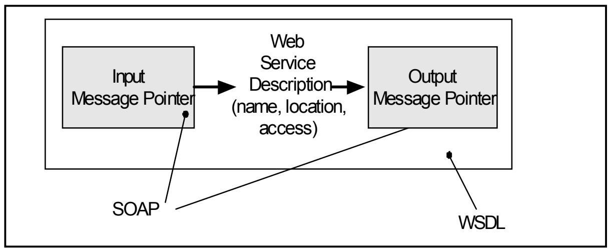


Figure 2.2 Composition of Web Service Description

The approach in this work is focused toward the development of local consensus ontologies of similar services. These technologies will allow agents to communicate more effectively about similar services. In addition, service composition agents become better prepared for introduction of future similar services by containing a more complete understanding of the domain. The initial direction in this work is to separate the WSDL/SOAP specification into three separate aspects, the input message information, the output message information, and the service-oriented information. Consensus can be made for multiple similar services by analyzing the ontology of each of these aspects. This approach toward *separation of concerns* [12] can be illustrated in Figure 2.3 for a ticket purchase web service. By separating these concerns, the agent also has flexibility in our future work in composing heterogeneous services. Further examples in this paper will demonstrate how consensus ontologies can be created from the varying input/output messages of similar services.

3. Local Consensus Ontologies

As discussed in Section 2, when an agent is searching for a particular Web service, its ontology of Web services may not match the ontologies of other agents listing their Web services. These agents need to be able to relate their ontologies to each other to form local consensus ontologies to find matches between their needed services and those advertised by another agent. In this section, we describe an approach for agents to form these local consensus ontologies. An ontology is a specification of an agent's known concepts and the relationships between those concepts [19].

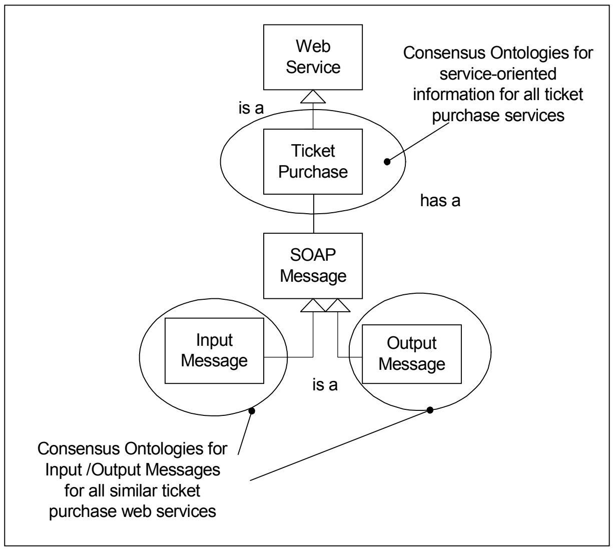
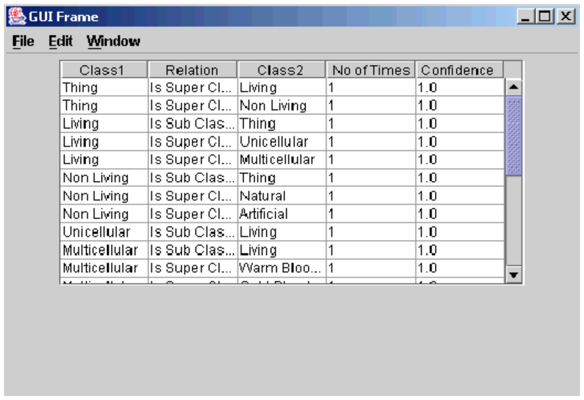

Web Service
Consensus Ontologies for service-oriented information for all ticket purchase services
is a
Ticket Purchase
has a
SOAP Message
Input Message
Output Message
is a
Consensus Ontologies for Input /Output Messages for all similar ticket purchase web services

Figure 2.3 Multiple Ontologies for Web Services

Our approach involves having agents autonomously merging ontologies to form relatively small, local ontologies between the agents they wish to communicate and query for a particular Web service. We assume that the ontologies being merged are in similar domains but there are not restrictions on the words, or terms, being used. Our approach allows us to find semantic and syntactic similarity by comparing two ontologies at a time with each other without the use of a global, common ontology and then merges these ontologies into a local, consensus ontology. With this approach, each agent maintains its own perspective on the concepts and relationships that exist between the agents' ontologies. First we explain how we create an ontology. Then we describe how we search for syntactic and semantic similarities, add new relations, and "learn" new concepts.

3.1 XML-based Ontology Creation

One of our aims is to make this local ontology consensus process easy to use for a non-knowledge engineer. Therefore, we provide a straightforward tool for generating ontologies (Figure 3.1).



Class1	Relation	Class2	No of Times	Confidence
Thing	Is Super Cl...	Living	1	1.0
Thing	Is Super Cl...	Non Living	1	1.0
Living	Is Sub Clas...	Thing	1	1.0
Living	Is Super Cl...	Unicellular	1	1.0
Living	Is Super Cl...	Multicellular	1	1.0
Non Living	Is Sub Clas...	Thing	1	1.0
Non Living	Is Super Cl...	Natural	1	1.0
Non Living	Is Super Cl...	Artificial	1	1.0
Unicellular	Is Sub Clas...	Living	1	1.0
Multicellular	Is Sub Clas...	Living	1	1.0
Multicellular	Is Super Cl...	Warm Bloo...	1	1.0

Figure 3.1 XML Ontology Creation GUI

The relationships used in these ontologies are:
1. is super class
2. is sub class
3. is part of
4. contains
5. equivalent

The ontologies are stored and represented in XML. Using XML makes the ontologies platform dependent and extensible for related formats such as DAML. Internally, the concepts and relationships are listed as a list of nodes. Each node has a name that represents a concept. This node also contains information about all the relationships of the node with other nodes in the ontology. The structure can be conceptually visualized as a tree structure and all the operations on it can be performed accordingly. Next we explain how we determine similarity identification.

3.2 Similarity Identification

Similarity identification involves the detection of concepts that are similar to concepts already understood by the agent. The agents look to identify whether two concepts are syntactically or semantically similar, or equivalent. First we look at how we determine syntactic equivalence and then semantic equivalence.

3.2.1 Syntactic Equivalence

Syntactic equivalence identification occurs when an agent attempts to find those concepts that are syntactically similar to those it already understands. For example, we would want to be able to determine that the concepts *Airplane* and *Airplanes* are syntactically equivalent.

We use edit distance [14] to measure syntactic equivalence. Edit distance is a well-established method for weighting the difference between two strings. It uses the minimum number of token insertions, deletions, and substitutions required to transform one string into another using a dynamic programming algorithm [14]. For example, the edit distance, *ED*, between the two lexical entries "The_Engine" and "TheEngine" equals 1, i.e. *ED("The_Engine"; "TheEngine") = 1.* This is because one delete operation changes the string "The_Engine" into "TheEngine". Based on Levenshtein's edit distance method, we calculate the syntactic similarity measure [15] for two strings using the following formula:

$$\boxed{SSM(s_i, s_j) = \max(0, \frac{\min(|s_i|, |s_j|) - ED(s_i, s_j)}{\min(|s_i|, |s_j|)})}$$

, where s_i and s_j are the two strings (i.e. concept names) that are being compared for syntactic similarity and *ED* is the edit distance. The SSM returns a degree of similarity between 0 and 1. A SSM measure of one would suggest an identical match between the two strings and a similarity of zero indicates a bad match. This method considers the number of changes that must be made to change one string into the other and weighs the number of these changes against the length of the shortest string. In our previous example, we compute the SSM("The_Engine"; "TheEngine") to be .883.

3.2.2 Semantic Equivalence

After checking for *syntactic equivalence*, the next phase of similarity identification is to determine *semantic* equivalence between one agent's concept and another agent's concept. For this we make use of the lexical database, WordNet [1]. We use synsets provided by the lexical database to get the meaning of the word in various contexts. Depending on the present context we choose the proper synset to compare to. Semantic matching is required since designers of ontologies often use different words to express the same concept. In this case, neither simple string comparisons nor syntactic comparisons lead to the identification of equivalence between these concepts.

For example, consider if one organization *A* called a concept *Jet* that has *Vehicle* as a parent with many children. If organization *B* has named a concept *Airplane* that has *Craft* as a parent, there would be no way of syntactically determining that *Airplane* and *Jet,* and *Craft* and *Vehicle* are semantically equivalent without the use of a lexical database such as WordNet.

3.3 Concept and Relation Discovery and Integration

3.3.1 New Concept and Relation Discovery

If one agent determines that another agent's concept is similar, or equivalent, to its own concept either syntactically or semantically, it can add this concept and its associated relations to its local consensus ontology. Associated with this new equivalent concept, are relations that were "attached" to this concept. These new relations can be added to the target local consensus ontology along with its relations which we will refer to as semantic relation *discovery*. For example, one agent *A* has the concepts *Vehicle, Car,* and *Engine*, with the relations *Car IS-A Vehicle* and *Car HAS-A Engine*. Agent B may have a concept *Auto* that is semantically equivalent to *Car*. Along with the new concept, agent B it may have the relation *Auto* HAS-A *Wheels*. When agent A incorporates this new relation into its local consensus ontology we say that Agent A has *discovered* the new concept *Auto* and the relation *Auto HAS-A Wheels*. The next step is described in the following subsection as "learning" or integrating the new relation into the local consensus ontology.

3.3.2 Semantic Relation Integration

There are two possible scenarios for semantic relation integration. Either the newly discovered concept is equivalent to a local ontology concept or it is not equivalent. First, we look at the case in which a semantically equivalent concept is discovered between two agent's ontologies. After the agent has discovered the new concept and its relations, it attempts to integrate these new relations into its local consensus ontology if possible. We will refer to this process as semantic relation *learning.* This is a form of "learning" of new concepts but not in the machine learning sense. Using the previous example, Agent A has discovered the new concept *Auto* and this concept's HAS-A relation, i.e. *Auto HAS-A Wheels*. Since it has found that *Auto* is semantically equivalent to *Car* it can use the lexical database to determine if *Car* also has wheels, or *Car HAS-A Wheels.* If it successfully incorporates this new relation we say that the agent has successfully integrated, or "learned", the new semantic relation.

The second scenario in semantic relation integration is if Agent A does not have a semantically equivalent concept from Agent B. In this case, we need to search for how the new concept from Agent B can be semantically integrated into Agent A's local consensus ontology. A concept from one agent's ontology may not be semantically equivalent to another agent's concept but may be related to another agent's concept. The discovery of this relationship could happen either through human intervention or through the use of a lexical database. In our approach to forming local consensus ontologies through autonomous merging, we use a lexical database, WordNet, to search for relationships between two concepts. Using the previous example, let us suppose that Agent B has the concept *Mercedez-Benz* that Agent A is trying to integrate into its ontology rather than the *Auto* concept. In this case, our Agent A uses the lexical database to attempt to integrate the new concept *Mercedez-Benz.* Further, Agent A discovers the relation, *Mercedez-Benz HAS-A Engine* from Agent B. Using the lexical database, Agent A can learn the relation *Mercedez-Benz IS-A Car.*

WordNet provides the ability to find several types of relationships including hypernyms, hyponyms, holonyms, and meronyms. A *hypernym* identifies concepts that could be the super class of the current concept. For example, *computer* is the hypernym of *supercomputer*. The inverse of this relation is the *hyponym*. The hyponym relation identifies all concepts that could be the sub class of the current concept. The hyponym relation can be thought of as the IS-A relationship. Our approach also uses the *holonym* relation provided by WordNet. The holonym relation identifies the 'whole' that a particular concept is a part of, or the HAS-A relation. For example, a computer can be a holonym of memory. The inverse of this relation is the *meronym*. The meronym identifies all concepts that is part of a given word, or the PART-OF relation. Using the same example, a memory chip is a meronym of a computer. We can specify number of levels that WordNet searches for these types of relations, or the relation search level depth (RSLD).

3.4 Local Ontology Consensus Merging Algorithm

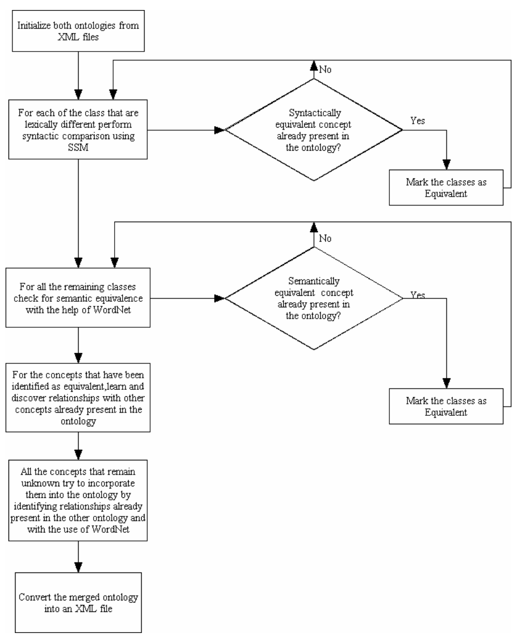
The overall algorithm for our local ontology consensus autonomous merging process is summarized in the following diagram (Figure 3.2).

4. Evaluation

We evaluated our approach using syntactic similarity, semantic similarity, semantic relation discovery in terms of performance measures such as concepts merged, relations discovered, and computing time. The purpose of these experiments were to determine if our approach reaches a consensus as we merge ontologies and to determine the impact of using a lexical database for learning new concepts and relationships.

4.1 Experimental Setup

Each of the tests that are described in this chapter were run on a Pentium 4 1.7GHz processor with a 512 MB RAM and a 18.6 GB hard-disk running a Windows 2000 operating system. We used Java as the programming language for creating the software used JDK version 1.4.1 for the experiments. We used 26 ontologies when running these experiments; these ontologies were obtained from the website and were created and used for experiments in [11]. Our experiments simulate having agents that wish to communicate with other agents but must merge and form local consensus ontologies prior to communicating to search for required services.



Initialize both ontologies from XML files
For each of the class that are lexically different perform syntactic comparison using SSM
Syntactically equivalent concept already present in the ontology?
No
Yes
Mark the classes as Equivalent
For all the remaining classes check for semantic equivalence with the help of WordNet
Semantically equivalent concept already present in the ontology?
No
Yes
Mark the classes as Equivalent
For the concepts that have been identified as equivalent,learn and discover relationships with other concepts already present in the ontology
All the concepts that remain unknown try to incorporate them into the ontology by identifying relationships already present in the other ontology and with the use of WordNet
Convert the merged ontology into an XML file

Figure 3.2 Autonomous Local Consensus Ontology Formation Algorithm

4.2 Ontology Consensus Experiments

To test whether our approach would successfully merge several agents' ontologies to a local, consensus ontology, we performed several merges between an agent and twenty-five other agents' ontologies. In this set of experiments, we simulate one agent forming a local, consensus ontology by autonomously merging with twenty-five other agents' ontologies. The average number of concepts per ontology was 13.5 concepts. We expected that the number of concepts would grow and that the agents would be able to reach a consensus. The results of these experiments are show in Figure 4.1.

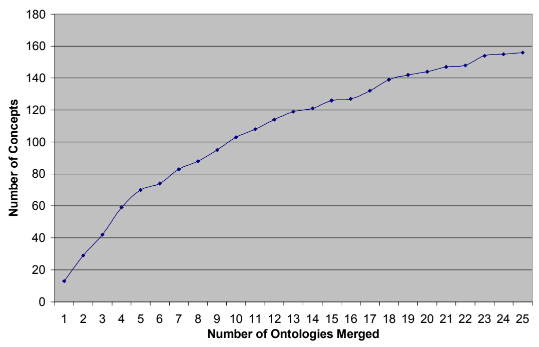


Figure 4.1 Number of concepts in base ontologies as ontologies are merged

This plot shows that the rate of increase in the number of new concepts decreases as the number of ontologies merged increases. This demonstrates a steady approach to consensus. We can clearly see that at the early stages, i.e. when two or three ontologies are being merged, there is a rapid increase in the number of concepts known to the agent, i.e. the agent does not understand many concepts and is rapidly discovering new concepts. As the number of ontologies merged with the agent increases, it understands most of the concepts though it still keeps learning a few more. So we can say that we have reached some sort of common level of understanding. This common level will have the same effect as if the agents had agreed on a common set of lexical words or ontology and so we might say that we have reached a consensus between these ontologies in this domain.

4.3 Semantic Relation Discovery and Integration Experiments

To see if there was a relationship between the number of concepts and relations discovered or learned (i.e. integrated) we plotted the number of new relations discovered, learned by the agent, as the number of ontologies being merged increases (figure 4.2). We have also plotted the total new relationships added to the ontology of the agent after each merge process. It is expected that the number of relationships discovered depends on the ontology with which the merge is taking place. It is also expected that the number of relationships learned will depend on the number of new concepts added at each stage as well as its relationships with the other concepts in the agent.

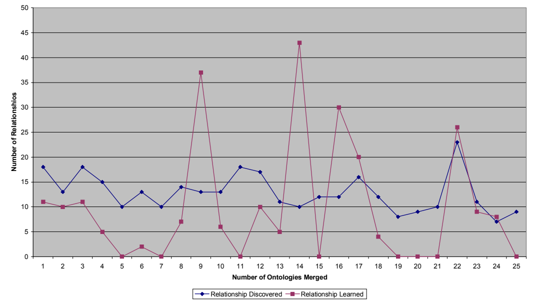


Figure 4.2 Number of new relationships being discovered, learned and total new relationships added after each merging process.

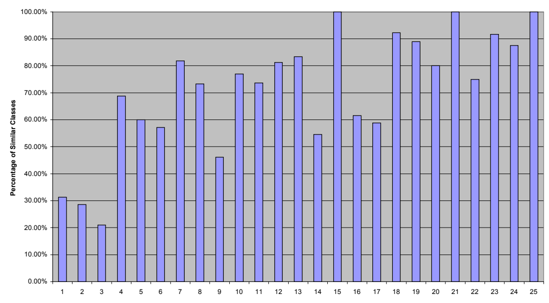


Figure 4.3 The percentage of similar concepts (i.e. syntactically or semantically equivalent) as the number ontologies merged increases.

One obvious question at this point will be what the graph exactly shows. We can see from the graph that the number of concepts discovered remains approximately the same, through out the process. This is because of the fact the most of the ontologies we are using have approximately the same number of concepts and relationships. The series showing the number of relation learned by the agent does not seem to show any pattern.

But a closer observation of this series along with the series of percentage of similar concepts observed (figure 4.3) revels that as the percentage of similar concepts increases, the number of new concepts decreases, and therefore the number of new relationships learned decreases. On the other hand when the percentage of similar concepts decreases, the agent learns more new concepts, as a result it learns more new relation.

4.4 Lexical Database Experiments

These sets of experiments are used to determine the effects of using and not using a lexical database on the local consensus ontology formation process. We setup our experiments as before with one agent merging with twenty-five agents' ontologies except we performed the merging with and without the lexical database.

In the figure 4.4, the graph shows the percentage of concepts that are found equivalent both with and without WordNet.

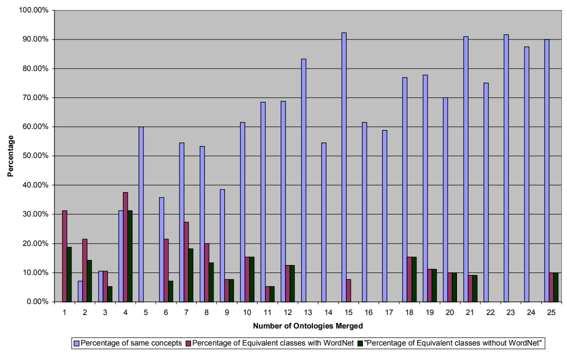


Figure 4.4 The percentage of equivalent concepts in experiments carried with and without the use of lexical database

In this case we consider equivalent concepts as only those that are lexically different, i.e. the classes don't have the same name, but they do represent the same concept, i.e. they are syntactically or semantically same. From the graph it can be seen that we achieve a much improved performance, i.e. semantic matches, when using WordNet in the initial stages while the percentage of matches decreases as the number of ontology merged increases. This will require some explanation, in the initial stages if we observe the percentage of same classes plotted this percentage is low, as this percentage increases the number of classes being tested for semantic match decreases, i.e. the number of classes being made available for equivalence match decreases as most of the concepts in the ontology is already understood by the agent and so the usage of WordNet seems to decrease as we approach a consensus. This is a general behavior, but as we will see later in this section, WordNet still helps in learning many new relations even as we approach consensus. We will also illustrate the use of WordNet with a simple ontology merge conducted both with and without the use of WordNet.

In the next graph (figure 4.5) we observe the number of new relationships that were added to the ontology both when WordNet was used and when it was not used. If this graph is observed we find that the number of new relationships added with WordNet is always better than the one without WordNet. Further we can observe that even after a number of ontologies have been merged, we still find that the case in which we use WordNet identifies and learns more relationship than the case we do not use WordNet. Further observation and comparison with Figure 4.6 will show that when the percentage of same concepts are high, few new concepts are identified and therefore few new relationships are learned and vice-versa. That will imply that the valleys of new relationships added with WordNet in Figure 7 will collide with the peaks of the percentage of similar concepts in Figure 6 and vice versa.

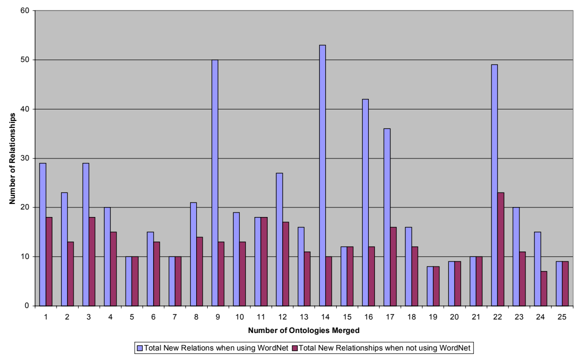


Figure 4.5 Number of new relationships discovered and learned with and without WordNet as the number of ontologies merged increases

Though the above graph illustrates the usefulness of WordNet, it might be argued that it might be better to just ignore WordNet and obtain better performance in terms of speed. From figure 9 we can see that the use of the lexical database use more time relative to not using the lexical database.

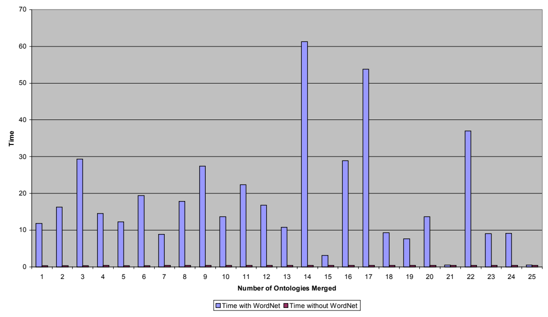


Figure 4.6 Time for ontology merging with and without use of lexical database

4.5 Merging Diverse Web Service-Oriented Ontologies

In this section we will examine the results of using our approach to merge two hypothetical but diverse ontologies from a B2B travel domain. In this scenario, we look at the input messages of two web services for ticket purchasing. Using these web services creates the potential for airlines to offer ticket purchasing services to other airlines. Though it is a fact that the system not using WordNet provides a much better performance in terms of speed, we might miss important semantic matches, further we might miss out on learning new relationships. We will take two ontologies (figure 4.6 and figure 4.7) and merge them with both the methods, i.e. with and without WordNet and observe the results.

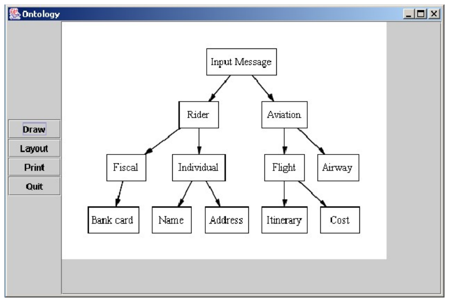


Figure 4.7 Base ontology of agent A. This ontology was merged with the ontology of agent B, both with and without the use of the lexical database.

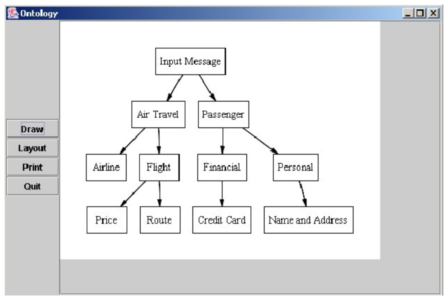


Figure 4.8 Base ontology of agent B. This ontology was merged with agent B's ontology both with and without the use of the lexical database.

In figure 4.9, we see the results of the merge between agent A's ontology (figure 4.7) and agent B's ontology (figure 4.8) *without* the use of the lexical database. We notice that there was no learning, or integration, of semantic relations and no semantically equivalent concepts found (figure 4.9). For example, the concepts *Passenger* and *Rider* were not found to be semantically equivalent.

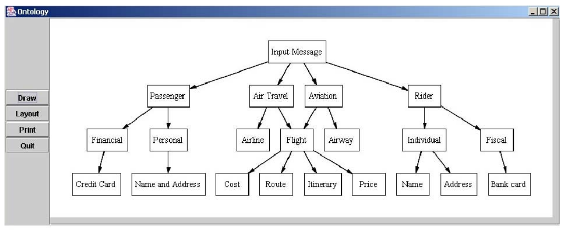


Figure 4.9 Merged ontology from Agent A and B WITHOUT the use of lexical database for semantic relation discovery and integration

However, we see in figure 4.10 the results of the merge between agent A's ontology and agent B's ontology *with* the use of the lexical database. We notice several equivalent concepts were discovered and semantic relations were integrated into the local consensus ontology (figure 4.10).

For example, the concepts *Rider* and *Passenger* were found to be semantically equivalent using the lexical database while they were not when the lexical database was not used in our local consensus ontology consensus process.

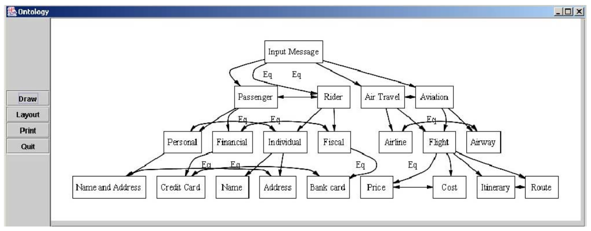


Figure 4.10 Merged ontology from Agent A and B WITH the use of lexical database for semantic relation discovery and integration

Thus we see how our approach can improve the forming of local consensus ontologies for agents in the B2B domain.

5. Related Work

While our work focuses on fully-automated merging of ontologies, there are several semi-automatic ontology merging tools with such as Chimera [24] , IDOCS [28], SMART [25], and PROMPT [25] . Another ontology merging project at ISI [26] attempted to build very large top level properties by merging ontologies like PENMAN Upper Model [27] and WordNet [1] These tools are used to assist a human knowledge engineer. One of the goals of our approach is to form local consensus ontologies through a fully automatic merging process. OntoMerge [23] is a semi-automated approach for agents and humans to deal with notational differences between ontologies with overlapping subject areas. It makes use of an inference engine and DAML ontologies but does not use a lexical database. [22] describes this approach using a planning and automated reasoning approach for self-describing service agents.

6. Conclusion and Future Work

We have shown how autonomous ontology merging for local consensus ontologies has potential for improving how agents conduct B2B Web service discovery and composition. The use of a lexical database increases the number of relationships found but increases the amount of time required to form the consensus ontologies. We also showed that rate of the number of concepts merged appears to decrease as the number of merge operations increases. We also demonstrated how our approach works at merging two diverse web service messaging ontologies. Future work will involve implementing this approach into a system of distributed B2B agents for further prototyping and evaluation. In addition to the ontologies created in this paper for input/output messages, we intend to create ontologies for other service-oriented concerns such as location and access protocols.

7. REFERENCES

[1] Miller, G. A. *WordNet: A Lexical Database for English*. Communications of ACM 38(11):39-41, 1995.

[2] Anderson, R.E. Social impacts of computing: Codes of professional ethics. Social Science Computing Review, 2 (Winter 1992), 453-469, 1992.

[3] Blake, M.B. " Agent-based Workflow Configuration and Management of On-line Services", Proceedings of the International Conference on Electronic Commerce Research (ICECR-4), pp 567-588, Dallas, TX, November, 2001.

[4] Ciancarini, P., Tolksdorf, R., Zambonelli, F., "Coordination Middleware for XML-Centric Applications", Proceedings of the 16th ACM Symposium on Applied Computing, Madrid (E), March 2002.

[5] McIlraith ,S. A., Son, T. C., and Zeng, H. . Semantic Web Services. IEEE Intelligent Systems 46-53, 2001.

[6] Web Services http://www.w3.org/2002/ws/desc/ , 2002.

[7] WSDL, http://www.w3.org/TR/wsdl, 2002.

[8] SOAP, http://www.w3.org/TR/soap12-part0/ , 2002.

[9] Blake, M.B. " B2B Electronic Commerce: Where do Agents Fit In? ", Proceedings of the AAAI-2002 Workshop on Agent Technologies for B2B E-Commerce (AAAI Press, Technical Report WS-02-01, ISBN 1-57735-154-1), pp 1-8, Edmonton, Alberta, Canada, July 28, 2002

[10] Blake, M.B.," An Agent-Based Cross-Organizational Workflow Architecture in Support of Web Services", Proceedings of the WETICE2002: The 2002 Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises/ IEEE Computer Society Press, pp 176-182 Pittsburgh, PA, 2002.

[11] Stephens, Larry, M., Huhns, Michael,N., *Consensus Ontologies Reconciling the Semantics of Web Pages and Agents.* In Proceedings of IEEE Internet Computing, p. 92-95, Sept 2001.

[12] Kiczales,G.,Lamping, J., Mendhekar, A., Maeda,C., Lopes, C.V., Loingtier, J., Irwin, J. " Aspect-Oriented Programming". In proceedings of the European Conference on Object-Oriented Programming (ECOOP), Finland. Springer-Verlag LNCS 1241.June 1997.

[13] Grefen, P., Aberer, K., Hoffner,Y., Ludwig,H.; CrossFlow: Cross-Organizational Workflow Management in Dynamic Virtual Enterprises; International Journal of Computer Systems Science & Engineering, Vol. 15, No. 5; pp. 277-290, 2000.

[14] Levenshtein, I. Binary Codes capable of correcting deletions, insertions, and reversals. Cybernetics and Control Theory, 10(8):707–710, 1966.

[15] Alexander Maedche and Steffen Staab. *Comparing Ontologies— Similarity Measures and a Comparison Study*, Institute AIFB, University of Karlsruhe, Internal Report, 2001.

[16] Sycara, K. and Zeng, D., "Dynamic Supply Chain Structuring for Electronic Commerce Among Agents" Intelligent Information Agents: Cooperative, Rational and Adaptive Information Gathering on the Internet, Matthias Klusch, ed., Springer Verlag, 1999.

[17] Universal Description, Discovery and Integration (UDDI) : http://www.uddi.org/, 2002.

[18] Walsh, M.P. Wellman and F. Ygge "Combinatorial Auctions for Supply Chain Formation "(Second ACM Conference on Electronic Commerce, 260-269, 2000.

[19] T. Gruber, "The Role of Common Ontology in Achieving Sharable, Reusable Knowledge Bases", J. A. Allen, R. Fikes, & E. Sandewall eds., *Principles of Knowledge Representation and Reasoning: Proceedings of the Second International Conference*, Cambridge, MA, Morgan Kauffman, 601-602, 1991.

[20] T. Finin, Y. Labrou, and J. Mayfied, "KQML as an Agent Communication Language", J. Bradshaw, ed., *Software Agents*, MIT Press, 1997.

[21] The DARPA Agent Markup Language Homepage", http://www.daml.org, 2001.

[22] McDermott, D., Burstein, M. and Smith, D. Overcoming ontology mismatches in transactions with self-describing service agents. International Semantic Web Workshop, Stanford, CA, July, 2001.

[23] Dejing, D., McDermott, D., and Peishen, Q. Ontology translation by ontology merging and automated reasoning, *Proc. EKAW Workshop on Ontologies for Multi-Agent Systems*, Siguenza, Spain , September 2002.

[24] McGuinness, D. ,Conceptual Modeling for Distributed Ontology Environments, *Proc. 8th International Conf. on Conceptual Structures Logical, Linguistic, and Computational Issues (ICCS 2000),* Darmstadt, Germany, August 14-18, 2000.

[25] Noy, N. and Musen, M. PROMPT: Algorithm and Tool for Automated Ontology Merging and Alignment *In the Proceedings of the Seventeenth National Conference on Artificial Intelligence (AAAI-2000),* Austin, TX, 2000.

[26] Chapulsky, H., Hovy, E. and Russ, T. 1997. *Progress on an Automatic Ontology Alignment Methodology*. ANSI Ad Hoc Group on Ontology Standards, 1997.

[27] Penman, *The Penman documentation*. Technical report, USC/Information Sciences Institute, Marina del Rey, CA, 1989.

[28] Williams, A.B., Krygowski, T., Thomas, G. "Using Agents to Reach an Ontology Consensus," *First International Joint Conference on Autonomous Agents and Multi-Agent Systems,* Bologna, Italy, July 15-19, 2002