Autonomous Agents and Multi-Agent Systems, 8, 165–193, 2004
© 2004 Kluwer Academic Publishers. Manufactured in The Netherlands.

Learning to Share Meaning in a Multi-Agent System[☆]

ANDREW B. WILLIAMS abwill@eng.uiowa.edu
University of Iowa, Department of Electrical and Computer Engineering, Iowa City, IA 52242

Abstract. The development of the semantic Web will require agents to use common domain ontologies to facilitate communication of conceptual knowledge. However, the proliferation of domain ontologies may also result in conflicts between the meanings assigned to the various terms. That is, agents with diverse ontologies may use different terms to refer to the same meaning or the same term to refer to different meanings. Agents will need a method for learning and translating similar semantic concepts between diverse ontologies. Only until recently have researchers diverged from the last decade's "common ontology" paradigm to a paradigm involving agents that can share knowledge using diverse ontologies. This paper describes how we address this agent knowledge sharing problem of how agents deal with diverse ontologies by introducing a methodology and algorithms for multi-agent knowledge sharing and learning in a peer-to-peer setting. We demonstrate how this approach will enable multi-agent systems to assist groups of people in locating, translating, and sharing knowledge using our Distributed Ontology Gathering Group Integration Environment (DOGGIE) and describe our proof-of-concept experiments. DOGGIE synthesizes agent communication, machine learning, and reasoning for information sharing in the Web domain.

Keywords: ontology learning, knowledge sharing, semantic interoperability, machine learning, multi-agent systems.

1. Introduction

Although it is easier for agents to communicate and share knowledge if they share a common ontology, in the real world this does not always happen. People and agents may use different words that have the same meaning, or refer to the same concrete or abstract object [3] or they may use the same word to refer to different meaning [11]. What is needed is a methodology for agents to teach each other what they mean. There are several questions related to this knowledge sharing problem [13] in a multi-agent system setting:

1) How do agents determine if they know the same semantic concepts?
2) How do agents determine if their different semantic concepts actually have the same meaning?
3) How can agents improve their interpretation of semantic objects by recursively learning missing discriminating attribute rules?
4) How do these methods affect the group performance at a given collective task?

[☆] Extended version of "Agents Teaching Agents to Share Meaning" published in conference proceedings for Fifth International Conference on Autonomous Agents (AGENTS 2001), Montreal, Canada, 465-472 May 28–June 1, 2001.

1.1. Ontologies and meaning

The definition of agency used in this paper states that an agent is a computing entity with or without a "body" that has varying degrees of the following capabilities or characteristics: autonomy, reasoning, social, learning, communication, and mobility [19, 30]. A group of these interacting agents are referred to as a multi-agent system [28]. Since the ontology problem [13] deals with how agents share meaning, we must provide a more precise definition of meaning. This requires that we must first differentiate between a conceptualization of the world, which only exists in a human or agent's mind and an *ontology*, which is a mapping of language symbols to that conceptualization and provides meaning to the symbols of the language. A *conceptualization* consists of all the objects and their interrelationships with each other that an agent hypothesizes or presumes to exist in the world and is represented by a tuple consisting of a *universe of discourse*, a *functional basis set*, and *a relational basis set* [9].

An agent's ontology consists of the specification of a conceptualization, which includes the terms used to name objects, functions, and relations in the agent's world [10]. An *object* is anything that we can say something about. An object can be concrete or abstract, primitive or composite, fictional or non-fictional. A *set* of objects can be grouped to form an abstract object called a *class*. We can use machine learning to learn a target function to map individual concrete objects to a particular class [21]. This target function will be referred to as a *concept description* of a class. The entire set of objects that we want to describe knowledge about is called a *universe of discourse*, *U*. *U* consists of both concrete objects, $X = \{x_1, \ldots x_n\}$ and abstract objects, $C = \{c_1, \ldots c_1\}$. This semiotic relationship between the world, universe of discourse (UOD) containing {X} and {C}, and an agent's conceptualization, interpretation function, and ontology is illustrated in Figure 1. A functional basis set contains the functions used for a particular universe of discourse. A relational basis set contains the relations used in a particular universe of discourse. The difference between the UOD and the ontology is that the

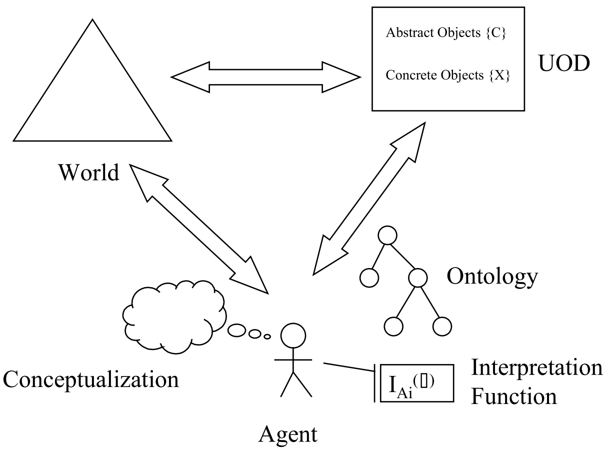

Figure 1. Semiotic relationship between world, universe of discourse (UOD) and conceptualization.

UOD are the objects that exist but until they are placed in an agent's ontology, the agent does not have a vocabulary to specify objects in the UOD. No matter how a human or computing agent conceptualizes the world, there are other conceptualizations that can be created. There may or may not exist a correspondence between two different agents' conceptualizations.

An agent's invention of its conceptualization is its first step towards describing knowledge about the world. Declarative knowledge can be used to represent an agent's environment and guide it in making intelligent decisions regarding its behavior in the environment [25]. This knowledge is represented by describing the world in sentences composed of a language such as natural language or first-order predicate calculus. Declarative semantics gives a precise way of defining meaning for an agent. The particular *meaning* defined for objects in a conceptualization are specified by *elements* in the representational language. The *object constant* is the label given to a particular object using the language. This mapping of objects in the conceptualization to elements in the language for a particular agent A_i can be described by an interpretation function, $I_{Ai}(\sigma)$. If σ is an object constant, then we can say that $I_{Ai}(\sigma) \in U_{Ai}$, where U_{Ai} is the universe of discourse for agent A_i. A *semantic concept*, is a term in a language that represents the meaning of a particular set of objects in the conceptualization. A semantic concept is an abstract object constant for a particular agent that is mapped to a set of concrete objects in the universe of discourse. A *semantic object* is an object taken from the universe of discourse and mapped to a particular semantic concept for an agent. The *semantic concept set* consists of all the semantic objects in a particular agent's semantic concept.

1.2. Distributed collective memory

A *distributed collective memory* (DCM) is the entire set of concrete objects $X = \{x_1 \dots x_n\}$ that exist in the world at a unique location and is accessible by any agent, A_i, in the multi-agent system, $A = \{A_1, \dots A_n\}$, but is only selectively conceptualized by each agent [7, 29]. This means that not every agent has every object in its conceptualization. We will denote U to represent the "global" universe of discourse where each agent A_1 has its own universe of discourse $U_{A1} \subset U$, which is a union of its known concrete and abstract objects, $U_{A1} = X_{A1} \cup C_{A1}$.

Stated another way, our research addresses the ontological diversity of artificial intelligence, which states that any conceptualization of the world can be invented and accommodated based on how useful it is to an agent [9]. With our approach, agents that share a distributed collective memory of objects will be able to overcome their lack of shared meaning to gain the ability to share knowledge between each other. The rest of this paper discusses our approach in more detail in section 2. Section 3 describes how we evaluated our system and section 4 discusses related work. Section 5 presents our conclusions and describes future work.

2. Approach

This section describes how DOGGIE agents addressed various aspects of the ontology problem. Before describing our approach we state the key assumptions of this work. We also describe how supervised inductive learning is used to enable the agents to learn representations for their ontologies. Descriptions for how agents are able to discover similar semantic concepts, translate these concepts, and improve interpretation of these concepts through recursive semantic context rule learning (RSCRL) are also given.

2.1. *Assumptions*

Several key assumptions exist for this work.

1) Agents live in a closed world represented by the distributed collective memory.
2) The identity of the objects in this world are accessible to all the agents and can be known by the agents.
3) Agents use a knowledge structure that can be learned using objects in the distributed collective memory.
4) The agents do not have any errors in their perception of the world even though their perceptions may differ.

2.2. *Semantic concept learning*

In a system of distributed, intelligent agents with diverse ontologies, there are opportunities for both individual and group semantic concept learning. In a following section we describe another type of group learning we employ related to semantic concept translation. Then we describe in detail our two novel algorithms we use for other types of individual learning. Individual learning refers to learning individual ontologies for each agent. Group learning is accomplished as the agents learn agent models. Agent model learning consists of one agent learning that another agent knows a particular concept. For example, Agent A learns that Agent B knows its concepts X, Y, and Z.

2.3. *An example of semantic concept representation in the World Wide Web domain*

An example of how concepts are represented in the World Wide Web domain is given. For the World Wide Web domain, Web pages contain semantic content related to a variety of subjects. The Web page contains text formatted using Hypertext Markup Language (HTML). The HTML may also be used to place images and sound recordings in the Web page. This example focuses on using the symbolic tokens in a Web page rather than actual audio recordings or images. A Web page is located using a Web browser by a unique Internet address specified by its Universal Resource Locator (URL). For this example, a Web page may be thought of as a specific semantic object, which can be grouped with similar Web pages to fit under a generalized class category, or semantic concept. For example, the USA Today Web page, http://www.usatoday.com, contains a variety of news subjects but can be classified by a user as a "Newspaper". Another user may label it, "Web News" or "Gannett Publications". A user can store, or "bookmark", the location of this Web page by placing the URL location in her bookmark list under the class category she defines it as. These bookmark lists are graphical hierarchies that group similar Web pages under categories. This can be viewed as a taxonomy that

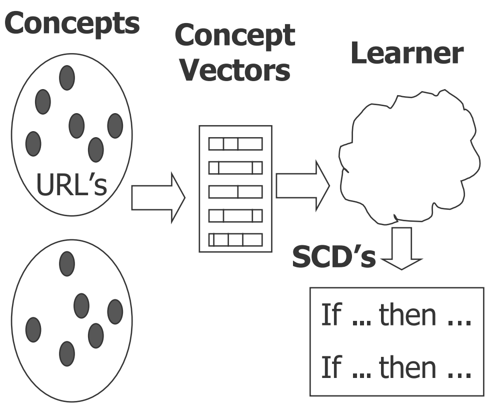

Figure 2. Semantic concept learning.

represents how a user views various Web pages on the Internet and becomes her representation of conceptualization for her agent's ontology. In essence, an agent representing the user can find a semantic object, interpret it according to its existing ontology and then store its location under the same semantic concept of other similar semantic objects. How two or more agents interpret and store the same semantic object will depend upon their individual ontologies.

A semantic concept comprises a group of semantic objects that describe that concept. The semantic object representations we use define each token, i.e. word and HTML tag from the Web page, as a boolean feature. The entire collection of Web pages, or semantic objects, that were categorized by a user's bookmark hierarchy is tokenized to find a vocabulary of unique tokens. This vocabulary is used to represent a Web page by a vector of ones and zeroes corresponding to the presence or absence of a token in a Web page. This combination of a unique vocabulary and a vector of corresponding ones and zeroes makes up an object vector. The object vector represents a specific Web page and the actual semantic concept is represented by a group of concept vectors judged to be similar by the user.

2.4. Ontology learning

Our agents use supervised inductive learning to learn their individual ontologies. The output of this ontology learning is semantic concept descriptions (SCD) in the form of interpretation rules as shown in Figure 2. These are intensional definitions of concepts as opposed to extensional definitions.

Each Web page bookmark folder label represents a semantic concept name. A Web page bookmark folder can contain bookmarks, or URL's, pointing to a semantic concept object, or Web page. A bookmark folder can also contain additional folders. Each set of bookmarks in a folder is used as training instances for the semantic concept learner. The semantic concept learner learns a set of interpretation rules for all of the agent's known semantic concept objects. An entire set of these types of semantic concept descriptions can then be used for future semantic concept interpretation.

```
(defrule Rule_35 (danny 1) => (assert (CONCEPT Arts_Book_Talk_Reviews)))
; 35 [85.1%]

(defrule Rule_34 (ink 1) => (assert (CONCEPT Arts_Book_Talk_Reviews)))
; 34 [50.0%]

(defrule Rule_23 (not (ismap 1)) (not (power 1)) (not (methods 1)) (end 1) =>(assert (CONCEPT
Busi_News_Dir_Gen)))
; 23 [82.0%]

(defrule Rule_31 (not (methods 1)) (learning 1) => (assert (CONCEPT Education_Adult))
; 31 [79.4%]

(defrule Rule_30 (howard 1) => (assert (CONCEPT Education_Adult))
; 30 [50.0%]

(defrule Rule_27 (breeders 1) => (assert (CONCEPT Life_Anim_Pets_Dogs)))
; 27 [70.7%]

(defrule Rule_26 (not (danny 1)) (because 1) => (assert (CONCEPT Life_Anim_Pets_Dogs))
; 26 [63.0%]

(defrule Rule_1 (not (title 1)) => assert (CONCEPT News_About_Current)))
; 1 [50.0%]

(defrule Rule_33 (methods 1) (not (ink 1)) => (assert (CONCEPT Comp_CS_Res_Resources)))
; 33 [70.0%]
```

Figure 3. Example semantic concept descriptions (SCD).

2.5. *Initial ontology learning*

Each agent uses a machine learning algorithm to learn a representation of its ontology. Each of the rules has an associated certainty value which can be used to calculate the positive interpretation threshold that will be described further in this section. Each rule consists of the rule name followed by the rule preconditions, or left hand side, and the rule postconditions, or right hand side. If the descriptors in the rule preconditions exist or do not exist as described in the rule clause, then the concept fact is asserted as stated in the rule postcondition. Examples of a set of concept descriptions for an agent's ontologies are given below.

The above semantic concept descriptions in Figure 3 resulted from learning an ontology consisting of concepts from the Magellan ontology such as *Arts:Books:Talk:Reviews* and *Computer:CS:Research:Resources*. The concept label, *Arts:Books:Talk:Reviews*, is from the ontology hierarchy consisting of the *Arts* superconcept with *Books* being a subconcept of *Arts*, *Talk* a subconcept of *Books* and *Reviews* a subconcept of *Talk*. Some of the descriptors that were learned using the machine learning algorithm appear to be more appropriate then others. For example, rule 31 in Figure 3 says that the presence of the *learning* descriptor and the absence of the descriptor, *methods*, indicates that the object instance belongs in the *Education_Adult* category. However, for rule 26 in Figure 3, the presence of the *because* descriptor and the absence of the *danny* descriptor indicates that the object instance may belong to the *Life_Anim_Pets_Dogs* concept. These semantic concept descriptions result due to the particular object instances each have tokens that result in sometimes a peculiar learned descriptor vocabulary.

2.6. Locating similar semantic concepts

The DOGGIE approach to enabling agents with diverse ontologies to locate similar semantic concepts can be summarized in the following steps:

1. An agent queries acquaintance agents for similar semantic concepts by sending them the name of the semantic concept and pointers to a sample of the semantic objects in the distributed collective memory. In essence, the agent is teaching the other agents what it means by a semantic concept by showing them examples of it.
2. The acquaintance agents receive the query and use their learned representations for their own semantic concepts to infer whether or not they know the same semantic concept. In other words, the acquaintance agents attempt to *interpret* the semantic objects based on their own ontology.
3. The acquaintance agents reply to the querying agent with a a) "Yes, I *know* that semantic concept", b) "I *may* know that semantic concept", or c) "No, I *don't* know that concept". If an acquaintance agent *knows* or *may* know that semantic concept, it returns a sample of pointers to its corresponding semantic concept.
4. The original querying agent receives the responses from the acquaintance agents and attempts to *verify* whether or not the other agents know a similar semantic concept. It does this by attempting its own interpretation of the semantic objects that were sent back to it using pointers.
5. If the original querying agent verifies the acquaintance's semantic concept, then it incorporates this applicable group knowledge into its knowledge base. This group knowledge is, in essence, "My acquaintance agent X knows my concept Y". A related hypothesis investigated dealt with how this type of group knowledge can improve group search performance for similar semantic concepts. Intuitively, the next time an agent can selectively send queries for knowledge regarding semantic concept Y to only agent X instead of *all* of its acquaintance agents.

The concept similarity location situation arises when one agent wants to find other agents in the MAS who know a similar semantic concept. Stated more formally, we have a multi-agent system, $A = \{A_1, \ldots, A_n\}$. Agent A_1 knows the semantic concept ϕ, or $K(A_1, \phi)$. This agent wants to find any other agent, A_i, that also knows the same concept ϕ, or $K(A_i, \phi)$. With our approach, agent A_1 sends a *concept-based query* (CBQ) to its acquaintance agents, $A_{acquaintance} \subset A$. The *concept-based query* is a tuple consisting of the semantic concept and a set of DCM addresses pointing to examples of that concept in the distributed collective memory , or $CBQ = <\phi, X_\phi>$. For each semantic concept ϕ that an agent A_i knows in its ontology, O_i, there is a set of object instances that make up this semantic concept, or $X_\phi = \{x_1 \ldots x_n\}$. For ϕ, there exists a function, c, such that $c(x) = \phi$. Using supervised inductive machine learning [21], the agent can learn the target function, h, such that $h(x) \approx c(x)$. In order to learn this target function, the decision tree [23], k-nearest neighbor [24], and Naïve Bayes [21] supervised machine learning algorithms were used in trial experiments. The initial experiments used the C4.5 decision tree algorithm [23] for its easy production of rules from the decision tree.

Given two agents, A_1 and A_2, that know concept ϕ, we can state the following:

$$K(A_1, \phi) \wedge K(A_2, \phi) \tag{1}$$

However, due to the size of the DCM, it is possible that each semantic concept corresponds to sets of objects that only overlap since the agents may not store the same objects in their local ontologies. That is, given the set of objects for ϕ_{A1} for agent A_1, or $X_{\phi,A1} = \{x_1, \ldots x_n\}$ and the set of objects for ϕ_{A2} for agent A_2, or $X_{\phi,A2} = \{x_1, \ldots x_m\}$, then $((X_{\phi,A1} \subset X_{\phi,A2}) \vee (X_{\phi,A2} \subset X_{\phi,A2})) \vee (X_{\phi,A1} \cap X_{\phi,A2}) \neq \emptyset$. Also, it is possible that there is no overlap of objects in each of the semantic concept sets for each agent, $(X_{\phi,A1} \cap X_{\phi,A2}) = \emptyset$. It was hypothesized that supervised inductive learning can be used to generalize each of the semantic concept sets and to implement an algorithm that will enable the agents to find concept similarity. Since supervised inductive learning is dependent upon the set of example objects used, we cannot assume that the target function learned for concept ϕ_1 is equal to the target function of concept ϕ_2. That is, $h_{\phi 1}(x) \neq h_{\phi 2}(x)$. Because of this, a method for estimating concept membership using the learned target functions was developed. The machine learning algorithm learned a set of concept descriptions for every semantic concept in the agent's ontology. So $H(x) = \{h_1, \ldots h_n\}$ where $h_1(x) = \phi_1$ and $h_n(x) = \phi_n$. This was used as the agent's knowledge base, or set of representations of facts about the world. If agent A_1 wants to determine if agent A_2 knew its concept ϕ, then it sends over a concept-based query consisting of the concept being queried ϕ along with a set of example objects of size k. Some example concept descriptions learned from an agent's ontology are given below:

```
(defrule Rule_33 (methods 1) (not (ink 1)) =>
(assert (CONCEPT Comp_CS_Res_Resources)))
; 33 [70.0%]

(defrule Rule_27 (breeders 1) => (assert
(CONCEPT Life_Anim_Pets_Dogs)))
; 27 [70.7%]
```

These semantic concept descriptions resulted from learning an ontology consisting of the *Life:Animals:Pets:Dogs* and *Computer:CS:Research:Resource* concepts from the Magellan [18] ontology. For each learned concept description h_i in $H_\phi(x)$, there exists a corresponding percentage describing how often this particular concept description correctly determined an object in the training set belonged to concept ϕ. This percentage is called the *positive interpretation threshold* for concept ϕ, or ϕ_+. *The negative interpretation threshold* was initially set at $1 - \phi_+ = \phi_-$. These thresholds were used to develop a similarity estimation function for two semantic concepts. If agent A_2 sends k addresses of its concept ϕ to agent A_1, then agent A_1 uses its set of concept descriptions, $H(x)$, as inference rules and seeks to interpret the example objects sent to it, $X_{A2} = \{x_1, \ldots x_k\}$. Given knowledge base $H(x)$ and the object, x_i, represented as facts, the agent A_1 seeks to determine if it can infer one of its own concepts ϕ_j,

$$H \wedge x_i \vdash \phi_j \tag{2}$$

The *interpretation value*, v, of concept ϕ_j is the frequency concept ϕ_j is inferred, $f_{\phi j}$, versus the total number of objects, k, in the CBQ,

$$\frac{f_{\phi_j}}{k} = v_{\phi_j} \tag{3}$$

The agent then compares the interpretation value $v_{\phi j}$ to that concept's positive interpretation threshold, ϕ_{j+}. If the interpretation value is greater than the concept's positive interpretation value then we say the agent *knows* the concept ϕ_j, or that the interpretation value falls into the K region.

$$\mathrm{v}_{\phi_j} \geq \phi_{j+} \Rightarrow \mathrm{K} \tag{4}$$

If the interpretation value for the concept is less than the negative interpretation threshold, then we say the agent *does not know* the concept ϕ_j designated by the D region.

$$\mathrm{v}_{\phi_j} \leq \phi_{j-} \Rightarrow \mathrm{D} \tag{5}$$

If the resulting interpretation value is between the positive and negative interpretation thresholds then we say the agent *may know* the concept designated by the M region.

$$\phi_{j-} < \mathrm{v}_{\phi_j} < \phi_{j+} \Rightarrow \mathrm{M} \tag{6}$$

Depending on which region the interpretation value falls into, the responding agent A_2 can send back a sample set of semantic objects of size j from its semantic concept set. The original querying agent A_1 can repeat the interpretation and membership estimation process described above. It does this to *verify* whether agent A_2 does in fact know the same semantic concept A_1 knows. If so, agent A_1 can incorporate the following *group knowledge* into its knowledge base,

$$\mathrm{K}(\mathrm{A}_1, (\mathrm{K}(\mathrm{A}_2, \phi))) \tag{7}$$

This states that agent A_1 knows that agent A_2 knows its concept ϕ.

In this context, *group knowledge* consists of any rule describing what semantic concept another agent in the MAS knows. This group knowledge is distinguished from joint or mutual knowledge. Group knowledge as referred to in this paper then refers to knowledge one agent has about what another agent knows rather than global knowledge known by the group. *Individual knowledge* is any rule that an agent knows or learns about its environment that does not incorporate group knowledge. The *verification* process for this knowledge interchange maintains the truth in the original querying agent's knowledge base.

2.7. Translating semantic concepts

The elegance of our approach is reflected in the fact that the algorithm for locating similar semantic concepts is essentially the same as the algorithm for translating semantic concepts. The key is that the algorithm relies on looking at the *semantic concept objects* themselves rather than relying on an inherent definition of meaning for the semantic concept (term) itself. If the querying agent and the responding agents agree to their different semantic concepts' meaning via the interpretation and verification process performed by each agent, then these semantic concepts translate to each other. The main difference between these two algorithms is how the group knowledge is stored. After the verification is successful, the original querying agent examines whether its semantic concept and the other agent's semantic concept are syntactically equivalent (i.e. same symbol). If so, the querying agent stores group knowledge that states "Agent B knows my semantic concept X as Y". This group knowledge will be used to direct the querying agents' future queries for concept X in order to improve the quality of information received in terms of precision and recall. Also, this group knowledge is used to improve group communication costs. It will know to ask agent B about concept Y if it wants to retrieve information on its own semantic concept X.

In this situation, the problem is that one agent may refer to the same semantic concept using different object constants.

$$\mathrm{K}(\mathrm{A}_1, \phi_1) \wedge \mathrm{K}(\mathrm{A}_2, \phi_2) \wedge \mathrm{sim}(\phi_1, \phi_2) \tag{8}$$

The hypothesis we investigated is that it is feasible for two agents to determine whether their semantic concepts are similar using inductive machine learning combined with agent communication. Another related hypothesis states that this knowledge could be used by the group to improve its group task performance. This situation deals with how these agents will be able to determine that their two different semantic concept constants refer to the same concept. Agent A_1 has a set of semantic objects associated with concept ϕ_1, $X_{\phi 1} = \{x_1, \ldots, x_n\}$ and agent A_2 has a set for ϕ_2, $X_{\phi 2} = \{x_1, \ldots, x_n\}$ for its concept. The problem is determining whether concepts ϕ_1 and ϕ_2 are similar, *sim*(ϕ_1, ϕ_2). As in the concept similarity location situation, we have the situation where the sets used by the two agents may have an overlap in their semantic concept sets, $[((X_{\phi,A1} \subset X_{\phi,A2}) \vee (X_{\phi,A2} \subset X_{\phi,A1})) \wedge (X_{\phi,A1} \cap X_{\phi,A2}) \neq \emptyset]$. On the other hand, the two agents might not have an overlap in their semantic concept sets, $\vee\ [(X_{\phi,A1} \cap X_{\phi,A2}) = \emptyset] \vee [(X_{\phi,A1} \cap X_{\phi,A2}) \neq \emptyset]$. Since the notion of strict semantic concept equality, $X_{\phi 1} = X_{\phi 2}$, is improbable due to the relative size of the distributed collective memory, the definition of semantic concept *similarity* described in the previous section was used. A concept is similar to another if their learned target functions can be used to successfully interpret a given set of semantic objects for a particular concept.

We can describe the input/output (I/O) behavior of two agents A_1 and A_2 during a CBQ interchange. The I/O behavior of agent A_2 responding to a CBQ can be described as follows:

Input:

$\mathrm{CBQ} = \{\phi_1, X_{\phi 1}\}$ where $X_{\phi 1} = \{x_1,\ldots,x_n\}$

$H_{A2}(x)$

Output:

$V_{A2} = \{<\phi_1,\ v_1,\ \text{region},\ X_1>,\ \ldots\ ,\ \{<\phi_n,\ v_n, \text{region}, X_n>\}$

The input into the agent A_2 responding to the query from A_1 is $X_{\phi 1}$, a sample set of semantic objects from agent A_1's concept ϕ_1, plus agent A_2's own knowledge base, $H_{A2}(x)$, consisting of semantic concept descriptions learned using the inductive machine learning algorithm. The output that is sent back to the original querying agent A_1 is V_{A2} which consists of a set of tuples, $\{<\phi_1, v_1, \text{region}, X_1>, \ldots, <\phi_n, v_n, \text{region}, X_n>\}$. Each ϕ_i is a possible matching concept, v_i is its interpretation value, *region* is the corresponding *K*, *M*, or *D* region symbol, and a corresponding sample set of semantic object addresses in the DCM.

The I/O behavior when agent A_1 receives its query response from agent A_2 to verify that they are referring to similar semantic concepts is described as follows:

Input:

$V_{A2} = \{<\phi_1, v_1, \text{region}, X_1>, \ldots, <\phi_n, v_n, \text{region}, X_n>\}$

Output:

$KB_{A1} \models K(A_2, \phi_2) \wedge \text{sim}(\phi_1, \phi_2)$

Agent A_1 receives the interpretation value set of tuples from agent A_2 as input and the output is any new knowledge agent A_1 learns regarding agent A_2's known semantic concepts. In the concept translation situation, agent A_1 learns that agent A_2 knows a concept ϕ_2 that is similar to its semantic concept ϕ_1.

Agent A_1 uses this algorithm to verify the results sent back to it by its acquaintance A_2. Agent A_1 will only perform the verification process for those concepts sent back from agent A_2 as *K* region concepts. If this occurs the agent first retrieves the objects by using the addresses received in the interpretation value set. Then the agent computes the frequency of inferences of a particular concept using its semantic concept descriptions. These frequencies are compared to the positive and negative interpretation thresholds to determine whether the candidate semantic concepts are actually known by the agent. If agent A_1 determines a *K* region for a particular candidate concept sent back from agent A_2 then it determines that its concept can be translated by the agent's concept and incorporates this knowledge into its knowledge base containing group knowledge:

$$KB_{A1} \leftarrow K(A_2, \phi_j) \wedge \text{sim}(\phi_1, \phi_2) \tag{9}$$

2.8. *Learning key missing descriptors*

These experiments were done to deal with key missing descriptors that may affect the semantic concept interpretation process. Two similar semantic concepts may not have overlapping semantic objects in the distributed collective memory. If this is the case, the $H_A(x)$ target function learned using supervised inductive learning for agent *A*'s semantic concept descriptions, and agent *B*'s $H_B(x)$ may have different key discriminating descriptors, or attributes, in them. For example, agent *B* attempts to interpret the semantic objects sent to it by agent A in a concept-based query using its knowledge base. The knowledge base H_B contains semantic concept descriptions in the form:

H_B:

Rule 1: $q1 \wedge q3 \Rightarrow C1$

Rule 2: $q5 \wedge \neg q6 \Rightarrow C1$

Rule 3: $q2 \wedge q3 \wedge q4 \Rightarrow C3$

Each q precondition is a proposition representing the presence of a particular attribute, or word, in the semantic object, e.g. Web page. The rule postcondition, C, represents a particular semantic concept known by the agent A_2. For example, if object x_1 contains the following attributes, $x_1 = \{q1, q2, q3, q5, q6\}$, then it would be interpreted as belonging to the concept $C1$. However, it would not be interpreted as belonging to concept $C3$. In this hypothetical example, Rule 2 would state that x_1 does not belong to concept C1. Our approach would deal with this disconfirming evidence by only counting the Rule 1 firing in calculating the interpretation value and ignoring the fact that Rule 2 did not fire for that particular instance.

If object $x_2 = \{q2, q3, q5, q6, q10, q11\}$ then it would not belong to any of the semantic concepts. After attempting interpretation of all the semantic objects in the CBQ, let us suppose that the interpretation value is calculated as explained in the previous section to be 0.6. Let us also suppose that the positive interpretation threshold is 0.7 and the negative interpretation threshold is 0.2. This results in an interpretation value in the M region. We believe that the agents could use Recursive Semantic Context Rule Learning (RSCRL) in order to improve interpretation. Since the original CBQ may have been for concept $C3$ and the agent responding to the query may in fact know concept $C3$ but may be missing a key discriminating attribute. As in our above example, the agent A_1 is missing the attribute $q4$ in the example x_2. RSCRL attempts to learn a *semantic context rule* for attribute $q4$.

The algorithm for RSCRL can be summarized as follows.

1) Determine the names of the semantic concepts in the agent's ontology.
2) Create meta-rules for the semantic concept descriptions using its rules.
3) Use the meta-rules and the interpreter to find which attributes to learn semantic context rules for.
4) Create new categories for these RSCRL indicators.
5) Re-learn the semantic concept description rules.
6) Create the semantic context rules from the semantic concept description rules indicated by the RSCRL indicators.
7) Re-interpret the CBQ using the new semantic context rules and the original semantic concept descriptions.
8) Determine whether the semantic concept was verified with the new semantic context rules.
9) If the concept is verified, learn the applicable group knowledge rules.

If the concept is not verified, recursively learn the next level of semantic context rules by repeating the above steps if the user-defined maximum recursion depth limit is not reached.

This RSCRL algorithm becomes a type of rule search for rules describing missing attributes in a semantic concept description. The meta-rules are automatically generated following the form for rules with two and three preconditions:

```
Rule 1: $q_a \wedge q_b \Rightarrow C2$
    Meta-Rule 1a: $\neg q_a \wedge q_b \Rightarrow$
        Action: Learn semantic context rule for $q_a$
    Meta-Rule 1b: $q_a \wedge \neg q_b \Rightarrow$
        Action: Learn semantic context rule for $q_b$
```

```
Rule 2: $q_a \wedge q_b \wedge q_c \Rightarrow C2$
    Meta-Rule 2a: $\neg q_a \wedge q_b \wedge q_c \Rightarrow$
        Action: Learn semantic context rule for $q_a$
    Meta-Rule 2b: $q_a \wedge \neg q_b \wedge q_c \Rightarrow$
        Action: Learn semantic context rule for $q_b$
    Meta-Rule 2c: $q_a \wedge q_b \wedge \neg q_c \Rightarrow$
        Action: Learn semantic context rule for $q_c$
```

Therefore, using the example Rule_33,

```
(defrule Rule_33 (methods 1) (not (ink 1))
 =>
(assert (CONCEPT Comp_CS_Res_Resources)))
; 33 [70.0%]
```

the following meta-rule is automatically generated for it during the RSCRL process:

```
(defrule Rule_45 (not (methods 1)) (not (ink 1))
 =>
(assert (RSCRL methods)))
```

This meta-rule will flag the agent that the CBQ's example semantic objects do not contain the attributes *methods* and *ink* and that the agent needs to reorganize to learn a *pseudo-concept* for this attribute *methods*. This will enable the agent to learn additional ontology rules for this descriptor. Once these RSCRL tokens are determined, the agent searches each ontology semantic concept directory for that token. If the token exists in a concept instance, it is removed from the current semantic object and placed in a concept holder named after the token. This builds up these pseudo-concepts with semantic objects, i.e. Web pages, which contain these tokens. Then using the supervised inductive learning algorithm, the agent generates additional interpretation rules for the agent's knowledge base. The semantic context rule generated for the descriptor *method* is:

```
(defrule Rule_9 (not (methods 1)) (this 1) (management1)
=>
(assert (methods 1)))
```

This rule states that for the current CBQ, if the *methods* token does not exist but the tokens *this* and *management* do exist, then we can assert the fact that the *methods* token does exist within the context of the current ontology. This is a unique method for determining whether an attribute "exists" given the current attribute set even though the exact attribute symbol is not used in the particular semantic concept set.

2.8.1. Automated meta-rule generation

The RSCL algorithm follows the principle of only learning semantic context rules for descriptors that will cause the original SCD rule to fire if a fact is asserted for that missing descriptor. That is, if the semantic context rule describing the missing descriptor as a pseudo concept fires, this will in turn fire the SCD rule. Therefore, our meta-rules are responsible for determining which descriptors to learn semantic context rules for. The meta-rules we automatically generate assert flags to indicate which descriptors need to be grouped into a new pseudo concept category. If a single descriptor is missing in a SCD rule with two or three precondition clauses, then we learn a meta-rule to indicate we need to learn a semantic context rule for that descriptor. The automated rule generation follows the following form (shown for rules with two and three preconditions):

- If A and B then Concept X
  > If ∼A and B then learn semantic context rule for A
  > If A and ∼B then learn semantic context rule for B

- If A and B and C then Concept X
  > If ∼A and B and C then learn semantic context rule for A
  > If A and ∼B and C then learn semantic context rule for B
  > If A and B and ∼C then learn semantic context rule for C

Meta-rules such as this are used to determine whether the ontology needs to be transformed to create a pseudo concept, or descriptor subdirectory, for methods from the existing semantic objects within the agent's ontology.

2.8.2. Creating descriptor pseudo concepts

After the meta-rules have been created, the original CBQ is sent to the agent's semantic concept interpreter to determine which descriptors need to have semantic context rules learned for them. Given the above example, if the CBQ's example semantic objects do not contain the descriptors "methods" and "ink" then the RSCRL flag for the token "methods" is asserted. This indicates to the agent that it needs to transform its ontology to learn a pseudo concept for this descriptor. A pseudo concept is a concept created for a descriptor by the agent to enable it to learn additional ontology rules for that descriptor. This additional learning will help improve the interpretation process for the given CBQ.

We create these pseudo concepts in the existing ontology from data already found in it. Once the RSCRL tokens are determined, we search each ontology concept directory for that token. If the token exists, we remove it from the current semantic object and place it in a concept holder named after the token. This builds up these pseudo concepts with semantic objects, i.e. Web pages, which contain these tokens. Then using our supervised inductive learning algorithm, we are able to generate additional ontology rules.

2.8.3. Semantic context rules

The semantic context rule generated for the descriptor given in our current descriptor is:

$$\text{(defrule Rule\_29 (not (methods 1)) (this 1) (management 1))} \Rightarrow \\ \text{(assert (methods 1).}$$

This rule states that for the current CBQ, if the "methods" token does not exist but the tokens "this" and "management" do exist, then we can assert the fact that the methods token does exist within the context of the current ontology. This is a novel method for determining whether a descriptor's "meaning" exists given the current vocabulary even though the exact token is not used in the current category. This is a key point in addressing the different vocabularies problem when learning ontologies in a multiagent system comprised of agents having diverse ontologies.

3. Evaluation

This section discusses how we evaluated our approach using DOGGIE and the results for these experiments.

3.1. Experiment design

We used Web search engine ontologies from Magellan [18] and Lycos [17]. Each agent had an ontology constructed from 8 to 12 ontology concepts. With the Magellan ontology data, we ran experiments on groups of 4, 8, and 16 agents using 10 examples per concept. In these experiments, there was no overlap between the training and testing data. With the Lycos ontology, we ran experiments in groups of 4 and 8 agents. In these experiments there was 100% overlap between training and testing data. We compared the group performance when using the C4.5 [23] machine learning algorithm versus the Naïve Bayes [21] learning algorithm in these Lycos experiments.

Unlike traditional information retrieval, our agents do not send individual queries consisting of a few terms. The concept-based queries sent consist of the query concept term along with locations of instances that represent that concept. Since we want to evaluate the overall performance of the agents as a group, we look at average concept precision and recall as well as the communication costs.

3.1.1. Agent ontologies

The concepts used were each assigned a unique ID and are listed in the Table 1 below. For some of the agent ontologies, concepts were randomly chosen to construct the individual agent ontologies. In others, the concepts were selected to build "narrow" ontologies that only included closely related concepts.

The actual agent ontologies used are given in Table 2 below. Agents 1 through 8 were constructed by randomly selecting 12 concepts from this list of 50 concepts. Agents 9 through 16 were hand constructed to be specialized ontologies containing concepts related to certain areas such as Arts, Business, Computing, Health and Regional Travel, Hobbies, Regional Travel and Science, Shopping, and Science and Sports, respectively. The concept ID numbers are used from Table 1 instead of their full titles for each of the concepts.

When performing concept translation, these concept ID's were modified to be unique yet they still referenced the same underlying concept. For example, concept ID *3504* may have been changed to *3504a3* for Agent 3, *3504a4* for Agent 4, and so on.

3.1.2. Performance measurements

In order to measure DOGGIE's performance, each agent keeps logs of its activities in files named for the agent. For example, an agent named, Agent1, will keep track of its Agent Control and Agent Engine components in files named Agent1Control.log and Agent1Engine.log, respectively. Also, some of an agent's activities are written to the standard output and can be redirected and stored for analysis. This information is particularly useful when tracking how queries are forwarded from one agent to another. The statistics used to measure the performance include concept precision, concept recall, and communication costs.

3.1.2.1. Concept precision. Concept precision, *cp*, is the ratio of the number of relevant *concepts* retrieved to the total number of *concepts* retrieved and is given by the following equation.

$$cp = \frac{K}{T}$$

where K is the number of relevant concept retrieved and T is the total number of concepts retrieved.

Concept precision differs slightly from traditional information retrieval (IR) precision since it is actually seeking a particular concept name rather than a particular document.

Table 1. Selected Magellan ontology concepts.

#	ID	Concept Name with Location in Magellan Ontology
1	3	Arts/Architecture/Firms
2	5	Arts/Architecture/Resources_and_Professional_Organizations
3	9	Arts/Books/Authors/Authors_A-H
4	42	Arts/Books/Genres/Non-Fiction
5	57	Arts/Books/Publishers/Educational
6	59	Arts/Books/Publishers/Major_Houses
7	60	Arts/Books/Publishers/Small_Presses
8	86	Arts/Fine_Arts/Galleries/International
9	97	Arts/Fine_Arts/Museums_Around_the_World/International
10	109	Arts/Fine_Arts/Resources/Connections_for_Artists
11	135	Business/Business_News_and_Directories
12	136	Business/Business_News_and_Directories/Directories
13	138	Business/Business_News_and_Directories/Trade_Publications
14	166	Business/Companies/Agriculture_and_Fisheries
15	170	Business/Companies/Chemicals_Petrochemicals_and_Pharmaceuticals
16	172	Business/Companies/Diversified_Biggies
17	1012	Computing/Hardware/LAN_Hardware
18	1014	Computing/Hardware/Microprocessors_And_ICs
19	1017	Computing/Hardware/Peripherals
20	1023	Computing/Internet/Bulletin_Boards
21	1026	Computing/Internet/E-mail
22	1027	Computing/Internet/For_Net_Novices
23	1029	Computing/Internet/Guides_To_The_Net
24	1033	Computing/Internet/Technoland
25	1041	Computing/Multimedia/Service_Providers
26	1047	Computing/Networks/Consultants
27	1051	Computing/Networks/Networking_Software
28	1054	Computing/Networks/Suppliers
29	2024	Health_and_Medicine/Medicine/Clinics/University_Medical_Centers
30	2078	Health_and_Medicine/Medicine/Technology/Radiology_and_Imaging
31	2090	Health_and_Medicine/Mental_Health/Resources
32	2114	Hobbies/Antiques_and_Collectibles/Sundry_Collectibles
33	2120	Hobbies/Arts_and_Crafts/Knitting_and_Stitching
34	2137	Hobbies/Cars_and_Trucks/Sundry_Auto_Info
35	2157	Hobbies/Games/Casino
36	2171	Hobbies/Games/Computer/Game_Biz
37	3504	Regional/Travel/Travel_Agencies/P_through_Z
38	3505	Regional/Travel/Travel_Agencies/A_through_F
39	3535	Science/Academies_and_Organizations/Research_Institutes
40	3549	Science/Astronomy_and_Space/NASA/Resources
41	3561	Science/Astronomy_and_Space/Research_Institutes
42	3562	Science/Astronomy_and_Space/Resources
43	4002	Shopping/Mixed_Bag/Names_All_Sound_The_Same
44	4003	Shopping/Mixed_Bag/Northeastern_USA
45	4004	Shopping/Mixed_Bag/Southeastern_USA
46	4017	Shopping/Novelties/Promotional_Items
47	4030	Shopping/Prized_Possessions/Collectibles
48	4074	Sports/Auto_Racing/Road_and_Track_Talk
49	4115	Sports/Basketball/NCAA
50	4127	Sports/College/School_Home_Pages

Table 2. Agent ontologies by concept ID.

Agent #	Concept ID #											
1	59	138	172	1012	1023	1054	2078	2090	2114	3504	3549	3562
2	3	5	59	86	135	1012	1026	2120	2171	3504	4030	4115
3	3	5	136	170	1029	1041	1047	3504	4002	4017	4030	4127
4	5	42	136	1014	1026	2024	2078	2137	3504	3535	3561	4002
5	5	42	135	170	1041	2137	2157	3504	3549	3561	4002	4017
6	57	97	1027	1029	1033	1041	2090	2157	3504	3505	3562	4074
7	9	59	135	1012	1017	1054	2024	2114	2137	2171	3504	3535
8	3	60	138	1014	1041	2024	2114	2171	3504	3505	4004	4074
9	3	5	9	42	57	59	60	86	97	—	—	—
10	135	136	138	166	170	172	—	—	—	—	—	—
11	1012	1014	1017	1023	1026	1027	1029	1033	1041	1047	1051	1054
12	2024	2078	2090	3504	3505	—	—	—	—	—	—	—
13	2114	2120	2137	2157	2171	—	—	—	—	—	—	—
14	3504	3505	3535	3549	3561	3562	—	—	—	—	—	—
15	4002	4003	4004	4017	4030	—	—	—	—	—	—	—
16	3535	3549	3561	3562	4074	4115	4129	—	—	—	—	—

For our research, relevance is defined as being either similar or related to the target, or query, concept. Since the results of a concept-based query can be either a K (know), M (may know), or D (Do not know) region, we define relevance to be a K region result.

3.1.2.2. Concept recall. Concept recall, *cr,* is the ratio of the number of relevant concepts retrieved to the total number of relevant concepts in the distributed knowledge base (KB) and is given by the following equation.

$$cr = \frac{K}{T_K}$$

where K is the total number of relevant concepts retrieved and T_K is the total number of relevant concepts in the entire distributed knowledge base.

The number of relevant concepts is the number of K region results from the concept-based queries. The total number of relevant concepts is the total number located among *all* the agents in the entire group of agents. For our experiments, this will be either among the group of 4, 8, or 16 agents.

3.1.2.3. Communication costs. Communication costs, C, are the total number of messages transmitted per query.

$$C = \sum m(t)$$

where m is a message sent at time t.

In a basic concept-based query (CBQ), the query sent is counted as one message and the reply is counted as another message. When queries are forwarded, each forwarded query is counted as a CBQ, i.e. two messages. Attempted queries for a concept may not be sent due to knowledge not existing for the concept. This attempted query counts as half a message.

The rest of this section describes the results of our concept similarity, concept translation, and recursive semantic context rule learning experiments. In all of the experiments, the agents sent random queries of their known concepts to each other. Each agent sent queries for each of all of its known concepts twice.

3.2. *Concept similarity experiments*

In these experiments, we determined how well DOGGIE would perform in locating similar semantic concepts in the group. For these experiments, we set up groups of 4, 8, and 16 agents for the experiments. Each agent was set up to randomly send concept-based queries for each of its known concepts in two iterations. That is, each agent sent out a query for one of its concepts, waited for and processed the reply, and then sent out another query until it was finished with all of its concepts. Then it would repeat these queries by going through its lists of concepts again. The first iteration of all of an agent's concepts sent as query is known as the *learning phase*. During the learning phase, agents had the opportunity to send out queries and learn *group knowledge*, or knowledge concerning who in the group knows what concepts. During the *post-learning phase*, agents can use this group knowledge to direct their future queries to improve concept precision. Next we will show the learning phase's *cumulative concept precision* for each agent and contrast it with the post-learning phase's *concept precision* for each agent. Then we will summarize the group performance using the cumulative concept precision and recall measures. Understanding the learning phase behavior of the group compared with the post-learning phase behavior explains the reason for the low concept precision values summarized in the tables. The actual group performance of the system after learning (post-learning phase) shows a much higher precision rate.

Figure 4 show how the concept precision in the concept similarity experiments improved from the learning phase to the post-learning phase on the second iteration due to the learned group knowledge for 4, 8, and 16 agents, respectively. The graphs are given to show how the learning process over time influences the precision. The summary tables provide the cumulative concept precision that is used to measure the group performance (i.e. average performance of individual agents) for the concept similarity, concept translation, and recursive semantic context rule learning experiments.

The equation used to determine the values plotted cumulative concept precision for the learning phase is:

$$cp(x) = \frac{\sum K_M(x)}{\sum T_M(x)}$$

where cp(x) is the value of concept precision taken at a log point before the next query is sent, $K_M(x)$, is the number of relevant concepts found at that point, and $T_M(x)$ is the total number of concepts retrieved at that point. The values plotted in the post-learning phase graphs use the equation for concept precision given in section 3.1. In the above 4, 8, and 16 agent scenarios, the value is a perfect 1.0 since the agent model rules were learned without error. The post learning phase plots have zero values when the concept precision (post-learning phase) is undefined. Since the agents are designed to only send queries

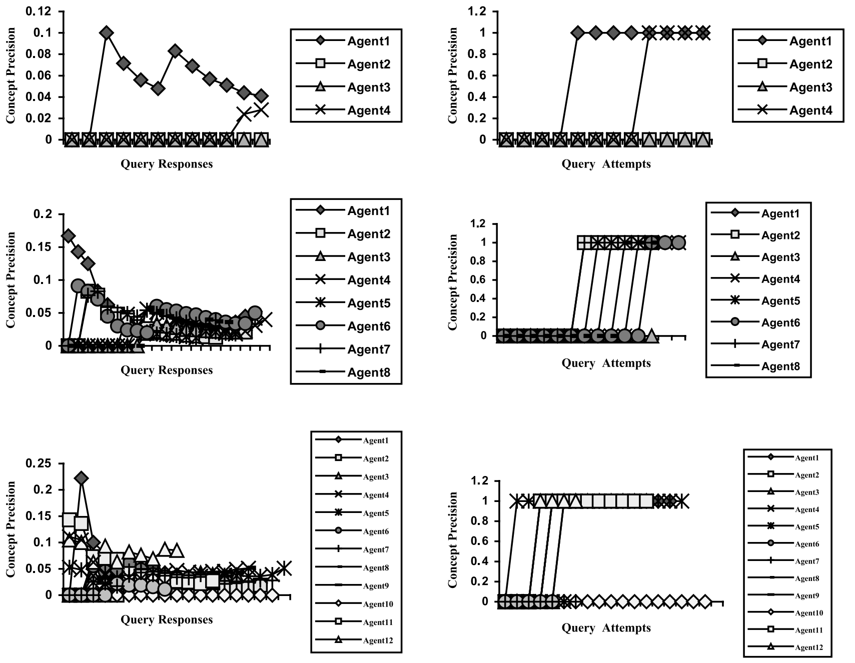

Figure 4. Plots of the concept precision (learning phase then post-learning phase phase) for 4, 8, and 16 agents respectively versus the number of query responses (time). Shows cumulative concept precision in learning phase and the improvement in precision during post-learning phase since agents have learned which agents know particular concepts.

when it has an applicable learned agent model rule, the concept precision in the post-learning phase is very high. The query responses refer to the responses that were received from agents from single queries. The data points were taken from the logs at the time just prior to where each new query was recorded.

The results of these experiments where there was no overlap between the training and testing examples are given in the table below.

These results from these experiments can be summarized as showing that this approach to finding concept similarity in a MAS is feasible. The fact that there is no overlap between the examples used for training and testing along with the very small sample set size of 10 semantic objects for the training examples are reflected in the low precision and recall values. The reason why a small sample size was initially chosen was to reflect the nominal size of a Web bookmark category and to show that our method works for very small sample sizes. Also reflected in these low values are the missing key attributes due to the different ontologies as we will describe in subsection 3.4. Another explanation of why these values are so low is that initially the agents were "blindly" sending queries to agents that may not even know the concepts. However, as the agents were able to learn agent model knowledge, the concept precision and concept recall should increase. The results in Table 3 were an average of all the agents performance and can be used to show the improvement resulting from using the recursive semantic context rule learning algorithm shown in subsection 3.4.

Table 3. Concept similarity MAS performance with no overlap between test and train data (Magellan ontology).

# Agents	Concept Precision	Concept Recall	Comm. Costs
4	0.02	0.09	63
8	0.034	0.126	156.22
16	0.035	0.166	274.29

3.2.1. *Comparison between different learning methods*

In addition to the above experiments using the Magellan Ontology, we ran experiments using DOGGIE with different machine learning methods: C4.5 and the Naïve Bayes using data from the Lycos ontology. In these experiments there was 100% overlap between the training and testing examples used. Also, we varied the number of examples used for each concept ranging from 10 per concept to 40 per concept and plotted learning curves as they related to concept precision and concept recall.

In Figures 5 and 6 below, we see how the performance of the DOGGIE MAS was effected by the type of individual learning algorithm used. There is a general upward learning trend for both the concept precision and recall with 4 and 8 agent configurations between 20 and 40 samples per concept. However, for the 40 samples per concept using C4.5 for 4 and 8 agent configurations there was a downward trend for concept precision and recall. This could possibly be due to overfitting caused by too many training samples for each concept in an agent's ontology. The 40 samples per concept experiments were not

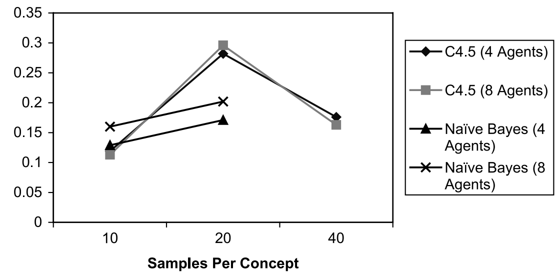

Figure 5. MAS concept precision with 100% overlap between train and test data (Lycos ontology).

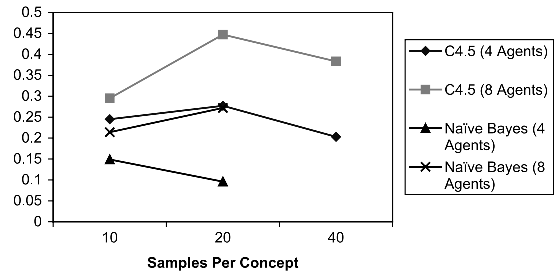

Figure 6. MAS concept recall with 100% overlap between train and test data (Lycos ontology).

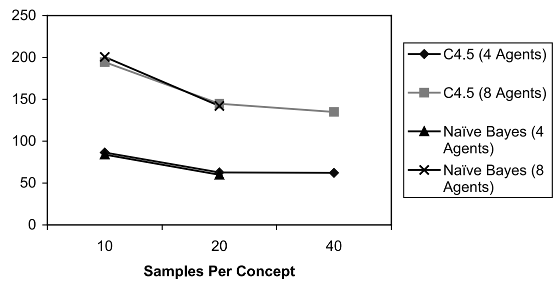

Figure 7. MAS communication costs with 100% overlap between test and train data (Lycos ontology).

performed using the Naive Bayes algorithm due to not having enough training examples to construct the test ontologies.

In Figure 7 above, we see that the communication costs have a downward trend as the number of samples per concept increase. In general we observe that the improvement in individual learning seems to reduce the amount of inter-agent communication required to locate the queried semantic concepts.

3.3. Concept translation experiments

In the concept translation experiments, we set up the agents in the same way we did for the concept similarity experiments except that this time we changed the names, or labels, for each concept to make them unique. The summary of these experiments with no overlap between test and train data using the Magellan ontology is given in Table 4 below.

The results of these experiments in 4-,8-, and 16-agent configurations is similar to the locating similar concepts experiments. Although these experiments demonstrated the feasibility of this approach, they also indicated the problem that two agents with diverse ontologies have, i.e. missing key attributes in their semantic concept sets. This problem is dealt with using the *recursive semantic context rule learning algorithm* (RSCRL). The following results will show that the RSCRL algorithm improves the concept similarity and translation algorithms considerably.

3.4. Recursive semantic context rule learning

In the recursive semantic context rule learning experiments (RSCRL) we demonstrate how this algorithm is used to find key discriminating features in order to improve the MAS performance in locating similar semantic concepts.

Table 4. Concept translation MAS performance with no overlap between test and train data (Magellan ontology).

# Agents	Concept Precision	Concept Recall	Comm. Costs
4	0.022	0.116	63
8	0.047	0.17	159.94
16	0.027	0.132	271.94

Table 5. Learning key missing attributes.

# Agents	Concept Precision	Concept Recall	Comm. Costs
4	0.035	0.142	78.6
8	0.085	0.336	157.5
16	0.069	0.345	273.47

Table 5 shows the summary averages for these experiments testing this algorithm with 4, 8, and 16 agents. This table shows the average values for concept precision, concept recall, and communication costs.

As in the other experiments, the concept precision was relatively small but relative to the baseline, there was much improvement.

The graphs in Figures 8 and 9 show the group precision and recall comparison between the concept similarity location, the concept translation, and RSCRL performance. We can see that the RSCRL outperformed the other methods by attempting to solve the missing key attributes problem. In relation to the baseline experiments, the concept precision improved up to 150% and the concept recall improved up to 166% over the baseline experiments as shown in the graphs below. The results for the sixteen agents were not as good for eight agents due to an unanticipated but insightful occurrence. When sixteen agents were sending and processing concept-based queries, their processing resources were not able to keep up with the demand. Therefore, they were not able to learn the agent

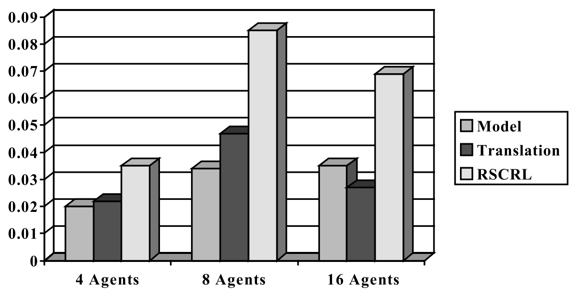

Figure 8. Concept precision MAS performance with no overlap between test and train data (Magellan ontology).

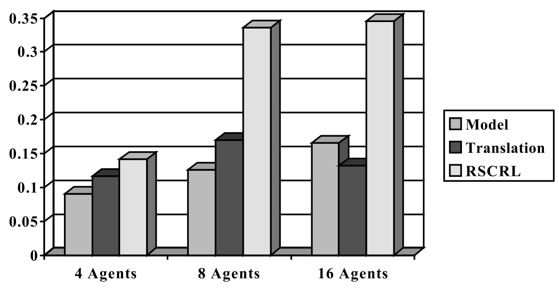

Figure 9. Concept recall MAS performance with no overlap between test and train data (Magellan ontology).

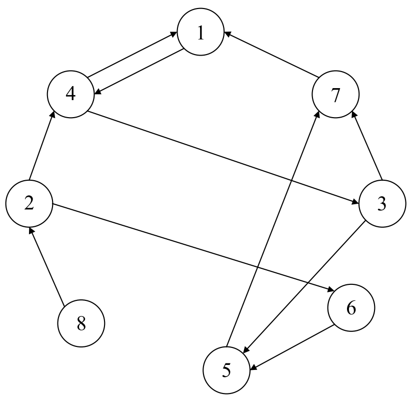

Figure 10. Query forward configuration for RSCRL experiments with eight agents using uni-directional communication.

model rules fast enough to keep with the new queries being sent. We include these results because they show that the amount of processing cycles, available memory, network bandwidth, and frequency of queries impact the quality of the interpretation performance.

3.4.1. Query forwarding in a non-fully connected network

We ran two experiments to compare how the RSCRL algorithm would affect the performance of agents in a non-fully connected network that forwarded their queries. These experiments also used the Magellan ontology data with no overlap between the training and testing data. In the first experiment we set up eight agents with randomly selected neighbors as depicted in Figure 10 below. Each agent was given ontologies with unique concept names. We set up each DOGGIE agent to perform concept translation and query forwarding with a maximum of two hops for a particular query. In the second experiment we had the same configuration except we also used RSCRL to see how the group performance was affected. For each of these experiments, the DOGGIE agents randomly send concept-based queries for two complete iterations of their concepts.

The results of these experiments are summarized in Table 6 below.

This table's "No RSCRL" column corresponds to the first experiment, which performed concept translation and query forwarding *without* the use of the RSCRL algorithm. The "RSCRL" column corresponds to the second experiment, which performed concept translation query forwarding *with* the RSCRL algorithm. The agents were able to find some of the concepts through concept translation and query forwarding as expected. The

Table 6. Query forward / concept translation summary.

	No RSCRL	RSCRL	Change (%)
CP	0.052	0.082	57.8%
CR	0.191	0.265	38.7%
Comm.	37.94	38.28	0.9%

RSCRL algorithm improved group performance as follows. The concept precision improved 57.8% and the concept recall improved 38.7% with the application of the RSCRL algorithm. Only a slight increase in communication costs was incurred at 0.9%. As in our other concept translation experiments with non-overlapping training and testing data, the concept precision and concept recall were relatively small. However, as expected, these values were greater than the fully-connected network configurations since there were fewer neighbors assigned in these experiments. The fewer neighbors in these experiments meant fewer messages were sent out during the group learning phase of the experiment.

4. Related work

In the previous decade, researchers in artificial intelligence realized that different knowledge bases and associated expert systems used different representations for knowledge. This made the re-use of knowledge bases between different expert systems very difficult and expensive. Through the Knowledge Sharing Effort (KSE) researchers developed technologies for sharing knowledge between expert systems and intelligent agents [6]. The KSE resulted in the development of methods for agents to communicate and share knowledge. In particular, the three aspects of communications were part of their focus – the syntax, semantics, and pragmatics of communication. One of the underlying ideas of the KSE paradigm was that in order for agents to communicate, they needed to share a common ontology. However, any human or computing agent has in his or its "mind" what objects and concepts exist in the world and can invent the particular concept label used to represent that concept. This creates difficulty when agents with these diverse ontologies attempt to share knowledge and communicate since they may refer to similar semantic concepts using different terms. An agent's *ontology* consists of the vocabulary of terms in the language mapped to the elements of the agent's conceptualization. A *conceptualization* consists of all the objects and their interrelationships with each other that an agent hypothesizes or presumes to exist in the world [9]. *Meaning* is given to objects in a conceptualization by mapping these objects in the world to elements in the language specified by the agent's ontology.

In recent times, research has been performed that directly addresses this ontology problem dealing with agents that originally handle diverse ontologies. [26] describes an innovative evolutionary computation approach that enables a multi-agent system (MAS) to converge to a common ontology using a language game. Since his research is closely related to the proposed research it is important to point out the similarities and differences. His research addresses a different, albeit important, problem of ontology *consensus*. That is, how can other agents converge to using a common ontology, or a shared lexicon, in what he refers to as a winner-take-all situation [26]. However, our proposed research recognizes that agents may often want to maintain their own diverse ontology but still be able to identify when they are referring to the same concept. For example, in the future, a GE purchasing software agent might want to keep its own ontology but know how it can communicate with a Siemens purchasing software agent. This allows for each agent to maintain control over its own ontology but still be able to communicate with each other. It is true that our proposed agents start off with diverse ontologies but their goal is not to converge to a common one as in [26]. The language game described in [26] relies on the speaker and hearer agents determining whether they use the same name to describe a *single* object but not a *class* of objects (i.e. concept) as in the Distributed Ontology Gathering Group Integration Environment (DOGGIE) system. This game assumes that *meaning* is defined as a pointer to a *single* object. However, in our research, our agents are given the ability to generalize *several* objects in order to find a "generic" description of concepts. In humans, this is analogous to Web users organizing the information and knowledge in Web pages by placing them in groups in a bookmark hierarchy, for example. [26] also demonstrates how agents might finding discriminating features for concepts using sets of feature detectors. The discrimination game is described for *one* object at a time rather than a *class of objects*, or concept, as in our proposed research. Again, this is because our approach makes use of machine learning techniques while [26] uses primarily evolutionary learning techniques.

Differentiated ontologies having terms that are formally defined as concepts and have local concepts that are shared have been addressed [27]. The relations they find between concepts are based on the assumption that local concepts inherit from concepts that are shared. In our approach, ontologies are not assumed to share commonly labeled concepts but rather a distributed collective memory of objects that can be selectively categorized into the agent's ontology. Their approach uses rough mapping to identify syntactic and semantic similarity between graphs of concepts with description logic. Unlike most approaches, they allow agents to communicate directly rather than translating to a central, shared language. However, they assume that the unshared terms inherit from terms in shared ontologies while we assume DOGGIE agents do not use shared ontologies. Their system was evaluated by generating description logic ontologies in artificial worlds while the DOGGIE approach uses Web pages to construct lightweight ontologies.

[8] argues the need for mechanisms for multi-agent systems to interoperate by using a non-learning approach to enable different multi-agent systems to translate messages or queries, for example. We tackle a different but slightly related problem when dealing with agents that have diverse ontologies. Our approach uses learning to enable agents to discover concept translations.

Machine learning algorithms have been used to learn how to extract information from Web pages [4]. Their approach uses manually constructed ontologies with their classes and relations and training data. Their approach uses machine learning to perform a form of text categorization. The objective of their work is to construct a knowledge base from the World Wide Web and not to find translations or relations between concepts in a multi-agent system.

Several information agent systems attempt to deal with some issues in using ontologies to find information. In the SIMS [15] system information agents use domain knowledge and information source knowledge to query multiple, heterogeneous databases. Our work builds upon their general notion of agents learning where to find relevant information and using machine learning to aid in this. However, agents in the DOGGIE system are both information sources and sinks and interact as both clients and servers. IICA [12], or Intelligent Information Collector and Analyzer, gathers, classifies and reorganizes information from the Internet using a common ontology rather than a group of diverse ones.

OBIWAN [31] uses agents to organize information on the Web using ontologies by mapping to centralized ontologies rather than each other agents' ontologies. The InfoSleuth Project [1] uses multiple representations of ontologies to help in semantic brokering. Their agents advertise their capabilities in terms of more than one ontology in order to increase the chances of finding a semantic match of concepts in the distributed information system. The InfoSleuth system, however, is not an attempt to discover translations between concepts in the different ontologies.

The Common Interest Seeker (COINS) uses local concept corpus functions like the common ontology of the SHADE matchmaker [16]. [16] states that the COINS matchmaker clients do not have to agree on a shared ontology explicitly other than sharing a common natural language. They state that COINS attempts to learn the ontology already shared by the other clients by revising its corpus to get a better estimate of the information content of words. Our approach differs from COINS in several key points. First, concepts in DOGGIE system are represented explicitly using symbolic rules rather than just document vectors. Second, our approach develops a method for locating and translating similar semantic concepts, while COINS does not perform semantic translation. In the DOGGIE system, the "matchmaking" is done by individual agents in a distributed manner and not by a centralized matchmaker as in the COINS system.

The DOGGIE approach differs from the natural language processing approach taken in the SENSUS system [14] by using ontologies created by individual agents and seeking to find translations and relationships between these concepts by using inductive learning techniques. Our research looks at how MAS performance is effected when locating and translating concepts using an instance-based approach.

The OBSERVER [20] system uses predefined inter-ontology relations to deals with the vocabulary sharing problem but the method of acquiring relationships whereas our DOGGIE approach builds relationships with different ontology data. [20] outlines a method for estimating loss of information based on terminological differences which may provide valuable performance information for future DOGGIE experiments. The SCOPES [22] multi-agent system views semantic reconciliation as a query dependent process that requires flexible interpretation of the query context. It provides a mechanism to flexibly construct a query context during coordinated knowledge elicitation.

[32] has promising results for mapping between ontologies using machine learning but it is not studied in a multi-agent setting as in this work. Also, their mapping process uses whole ontologies be used as input while our approach uses subsets of instances contained in an agent's single concept within its ontology. [33] demonstrate a method for integrating documents into a master catalog using information present in the source catalog and enhancing the use of the Naive Bayes classifier.

5. Conclusions and future work

We demonstrated how we address the ontology problem in a multi-agent system made up of agents with diverse ontologies. We described how our agents learn representations of their own ontologies using a machine learning algorithm and then seek to locate and/or translate semantic concepts by using examples of their concepts to query each other. In essence, our DOGGIE agents are able to teach each other what their concepts mean using their own conceptualization of the world. The learning method that used C4.5 with RSCRL worked best due to its ability to handle missing descriptors in the ontology rules. We plan to conduct experiments with this methodology for using more complex ontologies represented with DAML + OIL [5]. We will continue to investigate how DOGGIE agents perform when diverse learning styles are used. Although this work mainly deals with polysemy, it is our hope that it can be extended to deal with other relationships such as hyponymy, hypernymy, and meronymy. Classification by committee approaches may also be used in future work to enhance the machine learning techniques used. This work serves to introduce a novel methodology that may, in the future, prove useful for agents on the semantic web [2] when dealing with the knowledge sharing issues resulting from diverse ontologies.

Acknowledgments

This research has been partially funded by NSF grant IIS-0002364. The spidered Magellan and Lycos ontologies were contributed by Dr. Susan Gauch. Dr. Costas Tsatsoulis provided valuable guidance in the original development of the DOGGIE approach. I would also like to acknowledge Zijian Ren for his work on the Lycos experiments. Also, the anonymous reviewers provided excellent feedback that was used to improve this paper.

References

1. R. Bayardo, W. Bohrer, R. Brice, A. Cichocki, J. Fowler, A. Helal, V. Kashyap, T. Ksiezyk, G. Martin, M. Nodine, M. Rashid, M. Rusinkiewicz, R. Shea, C. Unnikrishnan, A. Unruh, and D. Woelk, "InfoSleuth: Agent-based semantic integration of information in open and dynamic environments," in M. Huhns and M. Singh, (eds.), *Readings in Agents*, Morgan Kaufmann: San Francisco, pp. 205–216, 1998.
2. T. Berners-Lee, J. Hendler, and O. Lassila, "The Semantic Web," *Scientific American*, May 2001.
3. A. H. Bond and L. Gasser, (eds.), *Readings in Distributed Artificial Intelligence*, Morgan Kaufmann, 1988.
4. M. Craven, D. DiPasquo, D. Freitag, A. McCallum, T. Mitchell, K. Nigam, and S. Slattery, "Learning to extract symbolic knowledge from the World Wide Web," in *Proceedings of the 15th National Conference on Artificial Intelligence (AAAI-98)*, 1998.
5. "The DARPA agent markup language homepage," http://www.daml.org, 2001.
6. T. Finin, Y. Labrou, and J. Mayfied, "KQML as an agent communication language," in J. Bradshaw, (ed.), *Software Agents*, MIT Press, 1997.
7. A. Garland and R. Alterman, "Multiagent learning through collective memory," in *Adaptation, Coevolution, and Learning in Multiagent Systems*, Technical Report SS-96-01, AAAI Symposium, Stanford, CA, March 25–27, Menlo Park, CA, AAAI Press, pp. 33–38, 1996.
8. J. Giampapa, M. Paolucci, and K. Sycara, "Agent interoperation across multiagent system boundaries," in *Proc. of 4th International Conference on Autonomous Agents*, June 3–7, Barcelona, Spain, 2000.
9. M. Genesereth and N. Nilsson, *Logical Foundations of Artificial Intelligence*, Palo Alto, CA, Morgan Kauffman, 1987.
10. T. Gruber, "The role of common ontology in achieving sharable, reusable knowledge bases," in J. A. Allen, R. Fikes, and E. Sandewall, (eds.), *Principles of Knowledge Representation and Reasoning: Proceedings of the Second International Conference*, Cambridge, MA: Morgan Kauffman, pp. 601–602, 1991.
11. M. Huhns and M. Singh, "Agents and multiagent systems: Themes, approaches, and challenges," in M. Huhns and M. Singh, (eds.), *Readings in Agents*, Morgan Kaufmann: San Francisco, CA, 1998.
12. M. Iwazume, K. Shirakami, K. Hatadani, H. Takeda, and T. Nishida, "IICA: An ontology-based internet

navigation system," *AAAI-96 Workshop on Internet-based Information Systems*, August 5, Portland, OR, 1996.

13. N. Jennings, K. Sycara, and M. Wooldridge, "A roadmap of agent research and development," *Autonomous Agents and Multi-Agent Systems*, vol. 1, pp. 7–38, 1998.
14. K. Knight and S. Luk, "Building a large-scale knowledge base for machine translation," in *Proc. of the National Conference on Artificial Intelligence (AAAI-94)*, 1994.
15. C. Knoblock, Y. Arens, and C. Hsu, "Cooperating agents for information retrieval," in *Proceedings of the Second International Conference on Cooperative Information Systems*, Toronto, Ontario, Canada: University of Toronto Press, 1994.
16. D. Kuokka and L. Harada, "Matchmaking for information integration," *Journal of Intelligent Information Systems*, 1996.
17. Lycos, "Lycos: Your personal internet guide," http://www.lycos.com, 1999.
18. Magellan, http://magellan.mckinley.com, 1999.
19. P. Maes, "Agents that reduce work and information overload," *Comm. of ACM*, vol. 37, no. 7, pp. 31–40, July 1994.
20. E. Mena, A. Illarramendi, V. Kashyap, and A. Sheth, "OBSERVER: An approach for query processing in global information systems based on interoperation across pre-existing ontologies," *International Journal Distributed and Parallel Databases*, vol. 8, no. 2, pp. 223–271, 2000.
21. T. M. Mitchell, *Machine Learning*, McGraw-Hill, 1997.
22. A. Ouksel, "A framework for a scalable agent architecture of cooperating knowledge sources," in M. Klusch, (ed.), *Intelligent Information Agents: Cooperative, Rational and Adaptive Information Gathering in the Internet*, Springer Verlag, 1999.
23. J. R. Quinlan, *C4.5: Programs for Machine Learning*, Morgan Kaufmann: San Mateo, CA, 1993.
24. J. Rachlin and S. Salzberg, "PEBLS 3.0 User's Guide," Department of Computer Science, John Hopkins University, 1993.
25. S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, Prentice Hall, Upper Saddle River, NJ, 1995.
26. L. Steels, "The origins of ontologies and communication conventions in multi-agent systems," *Autonomous Agents and Multi-Agent Systems*, vol. 1, no. 2, Kluwer Academic Publishers, pp. 169–194, October 1998.
27. P. Weinstein and W. Birmingham, "Agent communication with differentiated ontologies: Eight new measures of description compatibility," Technical Report CSE-TR-383-99, Department of Electrical Engineering and Computer Science, University of Michigan, 1999.
28. G. Weiss, *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence*, MIT Press: Cambridge, MA.
29. A. B. Williams, *Learning Ontologies in a Multiagent System*, Ph.D. Thesis, University of Kansas, 1999.
30. M. Wooldridge, "Intelligent agents," in G. Weiss, (ed.), *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence*, MIT Press: Cambridge, MA, pp. 28–77, 1999.
31. X. Zhu, S. Gauch, L. Gerhard, N. Kral, and A. Pretschner, "Ontology-based web site mapping for information exploration," in *Proc. 8th International Conference on Information and Knowledge Management*, Kansas City, Missouri, pp. 188–194, November 1999.
32. A. Doan, J. Madhavan, P. Domingos, and A. Halevy, "Learning to map between ontologies on the semantic web," *WWW 2002*, May 7, Honolulu, Hawaii, 2002.
33. R. Agrawal, and S. Ramakrishnan, "On integrating catalogs," *WWW 2001*, pp. 603–612, 2001.