Scalable Computing: Practice and Experience
Volume 7, Number 1, pp. 11–22. http://www.scpe.org
ISSN 1895-1767

EXPLOITING SHARED ONTOLOGY WITH DEFAULT INFORMATION FOR WEB AGENTS

YINGLONG MA*, BEIHONG JIN†, AND MINGQUAN ZHOU‡

Abstract. When different agents communicate with each other, there needs to be some way to ensure that the meaning of what one agent embodies is accurately conveyed to another agent. It has been argued that ontologies play a key role in communication among different agents. However, in some situations, because there exist terminological heterogeneities and incompleteness of pieces of information among ontologies used by different agents, communication among agents will be very complex and difficult to tackle. In this paper, we proposed a solution to the problem for these situations. We used distributed description logic to model the mappings between ontologies used by different agents and further make a default extension to the DDL for default reasoning. Then, base on the default extension of the DDL model, a complete information query can be reduced to checking default satisfiability of the complex concept corresponding to the query.

Key words. Ontology, Description Logic, Multi-agent System, Satisfiability, Default reasoning.

1. Introduction. Agents often utilize the services of other agents to perform some given tasks within multi-agent systems [1]. When different agents communicate with each other, there needs to be some ways to ensure that the meaning of what one agent embodies is accurately conveyed to the other agent. Ontologies play a key role in communication among different agents because they provide and define a shared vocabulary about a definition of the world and terms used in agent communication. In real-life scenarios, agents such as Web agents [2] need to interact in a much wider world. The future generation Web, called Semantic Web [3] originates from the form of decentralized vocabularies - ontologies, which are central to the vision of Semantic Web's multi-layer architecture [4]. In the background of the future Semantic Web intelligence, there are terminological knowledge bases (ontologies), reasoning engines, and also standards that make possible reasoning with the marked concepts on the Web. It now seems clear that Semantic Web will not be realized by agreeing on a single global ontology, but rather by weaving together a large collection of partial ontologies that are distributed across the Web [5]. In this situation, the assumption that different agents completely shared a vocabulary is unfeasible and even impossible. In facts, agents will often use private ontologies that define terms in different ways making it impossible for the other agent to understand the contents of a message [6]. Uschold identifies some barriers for agent communication, which can be classified into language heterogeneity and terminological heterogeneity [7]. In this paper, we will focus on terminology heterogeneity and not consider the problem of language heterogeneity.

To overcome these heterogeneity problems, there is a need to align ontologies used by different agents, the most often discussed are merging and mapping of ontologies [8, 9]. However, it seems that the efforts are not enough. For communication among agents with heterogeneous ontologies, there are still some problems that require to be solved. In some situations, only incomplete information can be got. These happen sometime as unavailability of pieces of information, sometime as semantic heterogeneities (here, terminological heterogeneities are focused on) among ontologies from different sources. Another problem is that there always exist some exceptional facts, which conflict with commonsense information. For example, commonly bird can fly, penguin belongs to bird, but penguin couldn't fly. In these situations, communication among agents will be more complex and difficult to tackle. We must consider not only the alignment of ontologies used by different agents, but also the implicit default information hidden among these ontologies. Then, information reasoning for query should be based on both these explicitly represented ontologies and implicit default information. This form of reasoning is called default reasoning, which is non-monotonic. Little attention, however, has been paid to the problem of endowing these logics above with default reasoning capabilities.

For a long time, representation and reasoning in description logic (DL) [10] have been used in a wide range of applications, which are usually given a formal, logic-based semantics. Another distinguished feature is the emphasis on reasoning as a central service. Description logic is very useful for defining, integrating, and

*Computer Sciences and Technology Department, North China Electric Power University, Beijing 102206, P. R. China, m_y_long@otcaix.iscas.ac.cn,
†Technology Center of Software Engineering, Institute of Software, Chinese Academy of Sciences, P. O. Box 8718, Beijing 100080, P. R. China, jbh@otcaix.iscas.ac.cn,
‡School of Information Sciences, Beijing Normal University, Beijing 100875, P.R.China, mqzhou@bnu.edu.cn

maintaining ontologies, which provide the Semantic Web with a common understanding of the basic semantic concepts used to annotate Web pages. They should be ideal candidates for ontology languages [11]. DAML+OIL [12, 15] and OWL [13] are clear examples of Description Logics. Recently, Borgida and Serafini proposed an extension of the formal framework of description Logic to distributed knowledge models [14], which are called distributed description logic (DDL). A DDL consists of a set of terminological knowledge bases (ontologies) and a set of so-called bridge rules between concept definitions from different ontologies. Two kinds of bridge rules are considered in DDL. Another important feature of DDL is the ability to transform a distributed knowledge base into a global one. In other words, the existing description logic reasoners can be applied for deriving new knowledge.

We adopt the view of [6] that the mappings between ontologies will mostly be established by individual agents that use different available ontologies in order to process a given task. In our opinion, it is a solution to model the mappings between ontologies used by different agents using a DDL and further make a default extension to the DDL for default reasoning. Then, base on the default extension of the DDL model, a complete information query can be reduced to check default satisfiability of the complex concept corresponding to the query.

This paper is organized as follows. Section 2 presents our motivation in making default extension to DDL for communication among multiple agents. Section 3 introduces representation and reasoning related to ontology. Distributed description logic are introduced particularly. In Section 4, we provide a formal framework for default extension to description logic. Default reasoning based on an EDDT is discussed in Section 5. Meanwhile, an algorithm is proposed for checking the default satisfiability of a given concept or a terminological subsumption assertion. Section 6 and Section 7 are related work and conclusion, respectively.

2. Motivation. In order to process a given task in multi-agent systems, it is important and essential to communicate with each other among different agents. However, there often exist some terminological heterogeneities and incompleteness of pieces of information among ontologies used by different agents, which make an agent not completely understand terms used by another agent. In the situations, it is difficult and even impossible to realize the communication among agents. We propose a solution to this problem for the situation. We model the knowledge representation of multi-agents using distributed description logic. The internal mappings between ontologies used by different agents are defined using the so-called bridge rules of distributed description logic. Then, by default extension to the DDL model, we can express explicitly some default information hidden among these ontologies. Based on the extension to DDL model, a query can be reduced to checking the default satisfiability of a concept or an assertion corresponding to the query. More precisely, an adapted algorithm is proposed for checking default satisfiability.

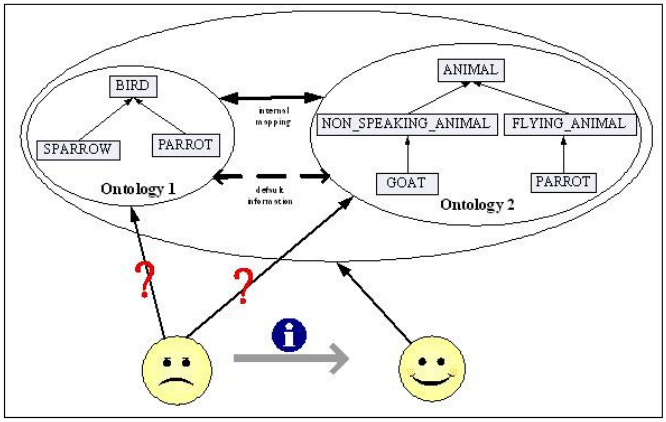

Fig. 2.1. *The situation of communication problem between two agents*

In order to express the problem to be resolved clearer, we make some assumption for simplicity. We only consider communication between two agents, whose ontologies are encoded on the same language. Then, we assume that ontologies used by the two agents have sufficient overlap such that internal mappings between them can be found. The following example shown in Figure 2.1 illustrates the situation described in the paper for our application.

In multi-agent systems, ontologies are used as the explicit representation of domain of interest. To process a given task, an agent perhaps uses multiple ontologies, which usually supplement each other and form a complete model. However, in the model, the default information among these ontologies is not considered. For example, we perhaps establish the internal mapping specifying that BIRD is a subclass of NON_SPEAKING_ANIMAL. Through the agent using ontology 1, we take the following query BIRD∧NON_SPEAKING_ANIMAL. The found question is that the agent using ontology 1 doesn't know the meaning of the term

NON_SPEAKING_ANIMAL

which only can be understood by the agent using ontology 2. To get complete and correct results of query, the two agents must coordinate each other. Another question is that the query results of the agent will include SPARROW and PARROT. We will find the results are partially correct because the class of PARROT can speak like man. The reason getting partially correct results is that we have not considered the default fact: in most cases, birds cannot speak; parrots belong to the class of birds but they can speak. In our opinion, default information should be considered and added into the model with multiple ontologies, which will form a sufficiently completely model. Then, available reasoning support for ontology languages is based on the model with default information.

3. Representation and Reasoning Related to Ontologies. A formal and well-founded ontology language is the basis for knowledge representation and reasoning about the ontologies involved. Description Logic is a formalism for knowledge representation and reasoning. Description logic is very useful for defining, integrating, and maintaining ontologies, which provide the Semantic Web with a common understanding of the basic semantic concepts used to annotate Web pages. It should be ideal candidates for ontology languages. One of the important proposals that have been made for well-founded ontology languages for the Web is DAML+OIL. Recently, description logic has heavily influenced the development of the semantic Web language. For example, DAML+OIL ontology language is just an alternative syntax for very expressive description logic [12]. So in the following sections, we use syntax and semantic representations of description logic involved instead of DAML+OIL. Description Logics is equipped with a formal, logic-based semantics. Its another distinguished feature is the emphasis on reasoning as a central service.

3.1. Description Logic. The basic notations in DL are the notation of concepts embracing some individuals on a domain of individuals, and roles representing binary relations on the domain of individuals. A specific DL provides a specific set of constructors for building more complex concepts and roles. For examples:

— the symbol $\top$ is a concept description which denotes the top concept, while the symbol $\bot$ stands for the inconsistent concept which is called bottom concept.

— the symbol $\sqcap$ denotes concept conjunction, e. g., the description Person$\sqcap$Male denotes the class of man.

— the symbol $\forall$R.C denotes the universal roles quantification (also called value restriction), e. g., the description $\forall$hasChild.Male denotes the set of individual whose children are all male.

— the number restriction constructor ($\geq$nR.C) and ($\leq$nR.C), e. g., the description ($\geq$1 hasChild.Doctor) denotes the class of parents who have at least one children and all the children are doctors.

The various description logics differ from one to another based on the set of constructors they allow. Here, we show the syntax and semantics of $\mathcal{ALCN}$ [16], which are listed as Figure 3.1.

Then we can make several kinds of assertions using these descriptions. There exist two kinds of assertions: subsumption assertions of the form $C \sqsubseteq D$ and assertions about individuals of the form $C(a)$ or $p(a, b)$, where C and D denote Concepts, p denotes role, and a and b are individual, respectively. For examples, the assertion $Parent \sqsubseteq Person$ denotes the fact the class of parents is subsumed by the class of person. The description $Person(a)$ denotes that the individual a is a person while the description $hasChild(a, b)$ denotes a has a child who is b. The collection of subsumption assertions is called Tbox, which specifies the terminology used to describe some application domains. A Tbox can be regarded as a terminological knowledge base of the description logic.

An interpretation for DL $\mathcal{I} = (\Delta^{\mathcal{I}}, \bullet^{\mathcal{I}})$, where $\Delta^{\mathcal{I}}$ is a domain of objects and $\bullet^{\mathcal{I}}$ the interpretation function. The interpretation function maps roles into subsets of $\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$, concepts into subsets of $\Delta^{\mathcal{I}}$ and individuals into elements of $\Delta^{\mathcal{I}}$. Satisfactions and entailments in DL Tbox will be described using following notations:

— $\mathcal{I} \models C \sqsubseteq D$ iff $C^{\mathcal{I}} \sqsubseteq D^{\mathcal{I}}$

— $\mathcal{I} \models T$, iff for all $C \sqsubseteq D$ in T, $\mathcal{I} \models C \sqsubseteq D$

— $C \sqsubseteq D$, iff for all possible interpretations $\mathcal{I}$, $\mathcal{I} \models C \sqsubseteq D$

DL Syntax	DL Semantic
$\neg C$	$\mathbf{L} \backslash C^{\mathcal{I}}$
$C \sqcap D$	$C^{\mathcal{I}} \cap D^{\mathcal{I}}$
$C \sqcup D$	$C^{\mathcal{I}} \cup D^{\mathcal{I}}$
$\exists P.C$	$\{x \mid \exists y.(x,y) \in P^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\}$
$\forall R.C$	$\{x \mid \forall y.(x,y) \in P^{\mathcal{I}} \rightarrow y \in C^{\mathcal{I}}\}$
$C \sqsubseteq D$	$C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$
$P \sqsubseteq R$	$P^{\mathcal{I}} \subseteq R^{\mathcal{I}}$
$C \sqsubseteq \neg D$	$C^{\mathcal{I}} \cap D^{\mathcal{I}} = \emptyset$
$\geq nP.C$	$\{x \in \mathbf{L} \mid \parallel \{y \in \mathbf{L} \mid (x,y) \in P^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\} \parallel \geq n\}$
$\leq nP.C$	$\{x \in \mathbf{L} \mid \parallel \{y \in \mathbf{L} \mid (x,y) \in P^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\} \parallel \leq n\}$
$C(a)$	$a \in C^{\mathcal{I}}$
$P(a,b)$	$(a,b) \in P^{\mathcal{I}}$

FIG. 3.1. *Syntax and semantics of ontology representation*

— $T \models C \sqsubseteq D$, iff for all interpretations $\mathcal{I}$, $\mathcal{I} \models C \sqsubseteq D$ such that $\mathcal{I} \models T$

3.2. Distributed Description Logic. A DDL is composed of a collection of "distributed" DLs, each of which represents a subsystem of the whole system. All of DLs in DDL are not completely independent from one another as the same piece of knowledge might be presented from different points of view in different DLs. Each DL autonomously represents and reasons about a certain subset of the whole knowledge. Distributed description logic (DDL) can better present heterogeneous distributed systems by modeling relations between objects and relations between concepts contained in different heterogeneous ontologies.

A DDL consists of a collection of DLs, which is written $\{DL_i\}_{i \in \mathrm{I}}$, every local DL in DDL is distinguished by different subscripts. The constraint relations between different DLs are described by using so-called "bridge rules" in an implicit manner, while the constraints between the corresponding domains of different DLs are described by introducing the so-called "semantics binary relations". In order to support directionality, the bridge rules from DL_i to DL_j will be viewed as describing "flow of information" from DL_i to DL_j from the point of view of DL_j. In DDL, $i : C$ denotes the concept C in DL_i, $i : C \sqsubseteq D$ denotes subsumption assertion $C \sqsubseteq D$ in DL_i. A bridge rule from i to j is described according to following two forms: $i : C \xrightarrow{\sqsubseteq} j : D$ and $i : C \xrightarrow{\sqsupseteq} j : D$. The former is called into-bridge rule, and the latter called onto-bridge rule. A DDL embraces a set of subsumption assertions, which are called DTB. A distributed Tbox (DTB) is defined based on Tboxes in all of local DLs and bridge rules between these Tboxes. A DTB $DT = (\{T_i\}_{i \in \mathrm{I}}, B)$, where T_i is Tbox in DL_i, and for every $i \neq j \in \mathrm{I}$, $B = \{B_{ij}\}$, where B_{ij} is a set of bridge rules from DL_i to DL_j. A DTB can be regarded as a distributed terminological knowledge base for the distributed description logics.

The semantics for distributed description logics are provided by using local interpretation for individual DL and connecting their domains using semantics binary relations r_{ij}. A distributed interpretation $\mathcal{J}=(\{\mathcal{I}_i\}_{i \in \mathrm{I}}, r)$ of DT consists of interpretations $\mathcal{I}_i$ for DL_i over domain $\Delta^{\mathcal{I}_i}$, and a function r associating to each $i, j \in \mathrm{I}$ a binary relation $r_{ij} \subseteq \Delta^{\mathcal{I}_i} \times \Delta^{\mathcal{I}_j}$. $r_{ij}(d) = \{d' \in \Delta^{\mathcal{I}_j} | (d,d') \in r_{ij}\}$, and for any $D \in \Delta^{\mathcal{I}_j}$, $r_{ij}(D) = \cup_{d \in D} r_{ij}(d)$. Note that semantic relation r must be bold everywhere.

A distributed interpretation $\mathcal{J}$ d-satisfies (written $\models_d$) the elements of DTB $DT = (\{T_i\}_{i \in \mathrm{I}}, B)$ according to following clauses: For every $i, j \in \mathrm{I}$

— $\mathcal{J} \models_d i : C \xrightarrow{\sqsubseteq} j : D$ if $r_{ij}(C^{\mathcal{I}_i}) \subseteq D^{\mathcal{I}_j}$
— $\mathcal{J} \models_d i : C \xrightarrow{\sqsupseteq} j : D$ if $r_{ij}(C^{\mathcal{I}_i}) \supseteq D^{\mathcal{I}_j}$
— $\mathcal{J} \models_d i : C \sqsubseteq D$ if $\mathcal{I}_i \models C \sqsubseteq D$
— $\mathcal{J} \models_d T_i$, if for all $C \sqsubseteq D$ in T_i such that $\mathcal{I}_i \models C \sqsubseteq D$
— $\mathcal{J} \models_d DT$, if for every $i, j \in \mathrm{I}$, $\mathcal{I}_i \models_d T_i$ and $\mathcal{I}_i \models_d b$, for every $b \in \cup B_{ij}$
— $DT \models_d i : C \sqsubseteq D$, if for every distributed interpretation $\mathcal{J}$, $\mathcal{J} \models_d DT$ implies $\mathcal{J} \models_d i : C \sqsubseteq D$

4. Default Extension to DDL. DDL is used to better model knowledge representation in a multi-agent systems, where ontologies are used as the explicit representation of domain of interest. The internal mappings

DT={{T_1={PARROT⊑BIRD,SPARROW⊑BIRD},
T_2={PARROT⊑FLYING_ANIMAL,
GOAT⊑ ¬SPEAKING_ANIMAL}
B={1:PARROT⊑2:PARROT}}
DF={BIRD(x):PARROT(x)/¬SPEAKING_ANIMAL(x)}

FIG. 4.1. *DT and D of the DDT*

$\Delta^{\mathcal{I}_1}$={parrot1,parrot2,sparrow,swan}, PARROT$^{\mathcal{I}_1}$={parrot1,parrot2}
SWAN$^{\mathcal{I}_1}$={swan}, SPARROW$^{\mathcal{I}_1}$={sparrow}
BIRD$^{\mathcal{I}_1}$={parrot1,parrot2,sparrow,swan}

$\Delta^{\mathcal{I}_2}$={parrot,goat,butterfly}, PARROT$^{\mathcal{I}_2}$={parrot}
GOAT$^{\mathcal{I}_2}$={goat}, FLYING_ANIMAL$^{\mathcal{I}_2}$={parrot,butterfly}
¬SPEAKING_ANIMAL$^{\mathcal{I}_2}$={goat},

r_{12}={(parrot1, parrot), (parrot2, parrot)}

FIG. 4.2. *The distributed interpretation of the DDT*

between ontologies used by different agents are defined using the so-called bridge rules of distributed description logic. As mentioned in Section 2, however, DDL model is not sufficient for modeling communication among multiple agents with heterogeneous ontologies because the default information among these ontologies is not considered. In this situation, query based on multi-agent systems will be possible to get partially correct results. To construct a sufficiently completely model, default information should be considered and added into the DDL model with multiple ontologies. In the following, we discuss the problem of default extension to DDL.

Our default extension approach is operated on a distributed terminological knowledge base. A distributed terminological knowledge base originally embraces only some strict information (i. e., the information having been expressed explicitly in distributed terminological knowledge base). Default information is used for getting complete and correct information from multiple distributed ontologies. We should consider a way to explicitly include and express the default information in a distributed terminological knowledge base for reasoning based on these distributed ontologies. To be able to include default information in distributed knowledge base, we firstly introduce the notation description of a default rule.

DEFINITION 4.1. *A default rule is of the form* $P(x) : J_1(x), J_2(x), \cdots, J_n(x)/C(x)$*, where* P, C *and* J_i *are concept names* ($1 \leq i \leq n$)*, and* x *is a variable.* $P(x)$ *is called the prerequisite of the default, all of* $J_i(x)$ *are called the justifications of the default, and* $C(x)$ *is called the consequent of the default. The meaning of default rule* $P(x) : J_1(x), J_2(x), \cdots, J_n(x)/C(x)$ *can be expressed as follows:*

If there exists an interpretation $\mathcal{I}$ *such that* $\mathcal{I}$ *satisfies* $P(x)$ *and doesn't satisfy every* $J_i(x)$ ($1 \leq i \leq n$)*, then* $\mathcal{I}$ *satisfies* $C(x)$*. Otherwise, if* $\mathcal{I}$ *satisfies every* $J_i(x)$ ($1 \leq i \leq n$)*, then* $\mathcal{I}$ *satisfies* $C(x)$*.*

For example, to state that a person can speak except if s/he is a dummy, we can use the default rule

$$\text{Person(x):Dummy(x)/CanSpeak(x).}$$

If there is an individual named John in a domain of individuals, then the closed default rule is

$$\text{Person(John):Dummy(John)/CanSpeak(John).}$$

To deal with strict taxonomies information as well as default information in distributed knowledge base, the definition of distributed knowledge base should be extended for including a set of default rules. We call the distributed terminological knowledge base with explicit default information default distributed terminological knowledge base, which is denoted as DDT.

DEFINITION 4.2. *A default distributed terminological knowledge base DDT=(DT,D), where DT is the DTB of distributed description logic, and D is a set of default rules.*

An example of a DDT is shown in figure 4.1. The DT of the DDT is based on two local terminological knowledge bases, named T_1 and T_2 respectively. The DT and D of the DDT are shown in Figure 4.1. Figure 4.2 provides a distributed interpretation of the DDK.

The satisfaction problem of DDT should be discussed for queries based on it. The satisfaction symbol is denoted as $\models_{dd}$. The kind of satisfiability of these elements in DDT means that they should satisfy not only DT, but also the set D of default rules. So we call satisfiability of elements in DDT default satisfiability. Default satisfiability serves as a complement of satisfiability definition in a distributed terminological knowledge base with default rules. In queries based on DDT, the definition will be used to detect satisfiability of a concept or assertion.

DEFINITION 4.3. *A distributed interpretation $\mathcal{J}$ dd-satisfies (written $\models_{dd}$) the elements of $DDT = (DT, D)$, according to following clauses: For every default rule δ in D, δ=$P(x) : J_1(x), J_2(x), \ldots, J_n(x)/C(x)$, for every $i, j \in$ I*

— $\mathcal{J} \models_{dd} DDT$, if $\mathcal{J} \models_d DT$ and $\mathcal{J} \models_d \delta$
— $\mathcal{J} \models_{dd} DT$, if $\mathcal{J} \models_d DT$ and $\mathcal{J} \models_d \delta$
— $\mathcal{J} \models_d \delta$, if $\mathcal{J} \models_d P \sqsubseteq C$ implies $\mathcal{J} \not\models_d J_k \sqsubseteq \neg C$ for all k $(1 \leq k \leq n)$
— $\mathcal{J} \models_d P \sqsubseteq C$, if $i \neq j$, such that $\mathcal{J} \models_d i : P \sqsubseteq C$ or $\mathcal{J} \models_d i : P \xrightarrow{\sqsubseteq} j : C$ or $\mathcal{J} \models_d j : C \xrightarrow{\exists} i : P$
— $DDT \models_{dd} DT$, if for all distributed interpretation $\mathcal{J}$, $\mathcal{J} \models_{dd} DDT$ implies $\mathcal{J} \models_{dd} DT$

In a distributed knowledge base, default information may have been used during reasoning, but a DDT is not really helpful for reasoning with default information in distributed knowledge. Some additional information with respect to default rules should be included explicitly into DT. A closed default rule of the form $P(x) : J_1(x), J_2(x), \cdots, J_n(x)/C(x)$ can be divided into two parts: $P(x) \rightarrow C(x)$ and $J_i(x) \rightarrow C(x)$, $(1 \leq i \leq n)$. We call the first part fulfilled rule, and the second exceptional rules. A rule of the form $A(x) \rightarrow B(x)$ means for every (distributed) interpretation $\mathcal{I}$, $x \in A^{\mathcal{I}}$, then $x \in B^{\mathcal{I}}$, i. e. $A \sqsubseteq B$, where A and B are concept names, and x denotes an individual.

DEFINITION 4.4. *An extended distributed knowledge base EDDT is constructed based on a DDT=(DT,D), according to the following clauses: For every default rule δ in D,δ=$P(x) : J_1(x), J_2(x), \ldots, J_n(x)/C(x)$,*

1) Dividing into two parts which embrace fulfilled rules and exceptional rules, respectively. The fulfilled rule denotes that it holds in most cases until the exception facts appear, while the exceptional rules denote some exceptional facts.

2) Adding $P \sqsubseteq C$ and $J_i \sqsubseteq C$ into DT $(1 \leq i \leq n)$, which are the assertions corresponding to fulfilled rule and exceptional rules, respectively

3) Setting the priorities of different rules for selecting appropriate rules during reasoning. The assertions corresponding to exceptional rules have the highest priority, while original strict information has normal priority. The assertions corresponding to fulfilled rules are given the lowest priority.

In the course of constructing an EDDT, default information has been added into distributed knowledge base for default reasoning, because these default information may have been used during reasoning. Exceptional information has been assigned the highest priority to avoid conflicting with some strict information, while fulfilled rules would be used only in the situation that no other strict information can be used, its priority is least. A simplified view of the EDDT based on the DDT and its interpretation (shown in figure 4.1 and 4.2) can be found in figure 4.3. The default rule $BIRD(x) : PARROT(x)/SPEAKING\_ANIMAL(x)$ is divided into one fulfilled rule and one exceptional rule, the fulfilled rule $BIRD \sqsubseteq \neg SPEAKING\_ANIMAL$ and the exceptional rule $PARROT \sqsubseteq SPEAKING\_ANIMAL$ has been added into EDDT. In fact, an EDDT can be recognized as a collection of integrated ontologies with default information expressed explicitly. Default reasoning can be performed based on an EDDT. In the following section, we will focus on how the default reasoning based on EDDT will be realized. Meanwhile, an adapted algorithm will be discussed for checking default satisfiability of complex concepts and subsumption assertions.

5. Reasoning with Default Information. Reasoning with default information provides agents using different ontologies with stronger query capability. In our opinion, a query based on DDT can boil down to checking default satisfiability of complex concept in accord with the query. Based on description logics, satisfiability of a complex concept is decided in polynomial time according to Tableau algorithm for $\mathcal{ALCN}$ [10, 16]. An important result of DDL is the ability to transform a distributed knowledge base into a global one. So the existing description logic reasoners can be applied for deriving new knowledge. This would allow us to transfer theoretical results and reasoning techniques from the extensive current DL literatures. In our reasoning approach with default information, the result will be used. The reasoning problem of distributed terminological

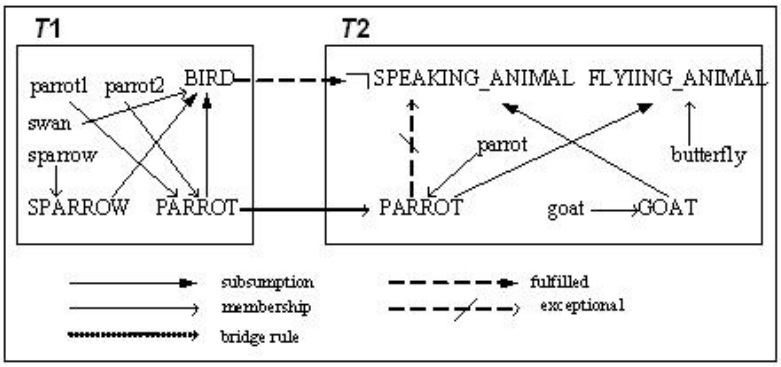

FIG. 4.3. *An example of EDDT*

knowledge base of a DDL will be transformed to the reasoning problem of terminological knowledge base of a global DL corresponding to the DDL. So in our opinion, detecting default satisfiability of a DDL is just detecting the default satisfiability of the global DL in accord with the DDL. A default extension to Tableau algorithm for $\mathcal{ALCN}$ DL can be used for detecting default satisfiability of $\mathcal{ALCN}$ concepts based on an EDDT.

DEFINITION 5.1. *A constraint set S consists of constraints of the form C(x), p(x,y), where C and p are concept name and role name, respectively. Both x and y are variables.*

An $\mathcal{I}$-assignment maps a variable x into a element of $\Delta^{\mathcal{I}}$. If $x^{\mathcal{I}} \in C^{\mathcal{I}}$, the $\mathcal{I}$-assignment satisfies $C(x)$. If $(x^{\mathcal{I}}, y^{\mathcal{I}}) \in p^{\mathcal{I}}$, the $\mathcal{I}$-assignment satisfies $p(x, y)$. If the $\mathcal{I}$-assignment satisfies every element in constraint set S, it sati sfies S. If there exist an interpretation $\mathcal{I}$ and an $\mathcal{I}$-assignment such that the $\mathcal{I}$-assignment satisfies the constraint set S, S is satisfiable. S is satisfiable iff all the constraints in S are satisfiable.

It will be convenient to assume that all concept descriptions in EDDT are in *negation normal form* (NNF). Using de-Morgan's rules and the usual rules for quantifiers, any $\mathcal{ALCN}$ concept description can be transformed into an equivalent description in NNF in linear time. For example, the assertion description $SPARROW \sqsubseteq BIRD$ can be transformed the form $\neg SPARROW \sqcup BIRD$. To check satisfiability of concept C, our extended algorithm starts with constraint set $S = \{C(x)\}$, and applies transformation rules in an extended distributed knowledge base. The concept C is satisfiable iff the constraint set S is unsatisfiable. In applying transformation rules, if there exist all obvious conflicts (clashes) in S, S is unsatisfiable, which means the concept C is satisfiable. Otherwise, S is unsatisfiable. The transformation rules are derived from concepts and assertions in EDDT. If the constraint set S before the action is satisfiable, S after the action is also satisfiable. The transformation rules of default extension to satisfiability algorithm are shown as Figure 5.1.

When the adapted algorithm is used for detecting default satisfiability of $\mathcal{ALCN}$ concepts, every action must preserve satisfiability. Because if an action don't preserve satisfiability, we cannot ensure the condition that if the constraint set before the action is satisfiable then the set after the action is satisfiable. In the extension algorithm, we must prove the actions preserve satisfiability.

THEOREM 5.2. *The action of the applied transformation rules preserves satisfiability.*

Proof. Because a DDL can be regarded as a global DL, for simplification, we use interpretation $\mathcal{I}$ of the global DL for distributed interpretation $\mathcal{J}$ of the DDL.

In the extension algorithm, every step may involve the actions of some transformation rules that are applied. so we must prove all of these actions in these steps preserve satisfiability. Because the actions in the second step are originally derived from the classical Tableau algorithm, we have known they preserve satisfiability [10]. The remainder of the proof will only consider the actions in the first step and the third step.

1) In the first step, the action condition is that for any default rule of the form

$$\mathrm{P}(x)\colon \mathrm{J}_1(x), \mathrm{J}_2(x), \ldots, \mathrm{J}_n(x)/\mathrm{C}(x)$$

in set of default rules, there exists $\mathrm{J}_i(x)$ is contained in S. If the constraint set S before the action is satisfiable, then there exists an interpretation $\mathcal{I}$ such that $\mathcal{I}$ satisfies all of elements of S. Because $\{\mathrm{J}_i(x)\} \subseteq \mathrm{S}$, then $\mathcal{I}$ satisfies $\mathrm{J}_i(x)$ $(1 \leqslant i \leqslant \mathrm{n})$. Furthermore, according to the Definition 4.1, we know $\mathcal{I}$ satisfies $\neg\mathrm{C}(x)$ after the action. From the above, we know that $\mathcal{I}$ satisfies both $\neg\mathrm{C}(x)$ and S, i. e., $\mathcal{I}$ satisfies $\mathrm{S} \cup \{\neg\mathrm{C}(x)\}$ after the action.

Exceptional rules:(Used for Step 1)
Condition:
For any default rule of the form P(x):J$_1$(x), J$_2$(x),...,J$_n$(x)/C(x), there exists J$_i$(x) (1$\leqslant i \leqslant$n) is contained in S,but S doesn't contain $\neg$C(x).
Action:
S=S$\cup${$\neg$C(x)}
Strict rules:(Used for Step 2)
$\sqcap-$rule:
Condition:
{(C$\sqcap$D)(x)}$\subseteq$S, but S doesn't contain both C(x) and D(x).
Action:
S=S$\cup${C(x), D(x)}
$\sqcup-$rule:
Condition:
{(C$\sqcup$D)(x)}$\subseteq$S but {C(x),D(x)} $\cap$ S=$\emptyset$.
Action:
S=S$\cup${C(x)} or S=S$\cup${D(x)}
$\exists-$rule:
Condition:
{($\exists$R.C)(x)} $\subseteq$S, but there is no individual name y such that S contains C(x) and R(x, y).
Action:
S=S$\cup${C(y), R(x, y)}
$\forall-$rule:
Condition:
{($\forall$R.C)(x), R(x, y)} $\subseteq$S, but S doesn't contain C(y).
Action:
S=S$\cup${C(y)}
$\geq$n-rule:
Condition:
($\geq$nR)(x) S, there doesn't exist individual names y$_1$, y$_2$,$\cdots$,y$_n$ such that R(x,y$_i$) and y$_i\neq$y$_j$ are in S, (1$\leq$i$\leq$j$\leq$n).
Action:
S=S$\cup$R(x,y$_i$)$\cup$y$_i\neq$y$_j$, (1$\leq$i$\leq$j$\leq$n), where y$_1$,$\cdots$, y$_n$ are distinct individual names not occurring in S.
$\leq$n-rule:
Condition:
distinct individual names y$_1$,$\cdots$,y$_{n+1}$ are contained in S such that ($\leq$nR)(x) and R(x,y$_1$),$\cdots$, R(x,y$_{n+1}$) are in S, and y$_i\neq$y$_j$ is not in S for some i,j,
1$\leq$i$\leq$j$\leq$n+1.
Action:
for each pair y$_i$ and y$_j$, such that 1$\leq$i$\leq$j$\leq$n+1 and yi$\neq$yj is not in S, the S$_{i,j}$:=[y$_i$/y$_j$]S is obtained from S by replacing each occurrence of y$_i$ by y$_j$.
Fulfilled rule: (Used for Step 3)
Condition:
no other transformation rules is applicable, and for any default rule of the form P(x):J$_1$(x), J$_2$(x), ..., J$_n$(x)/C(x), {P(x)}$\subseteq$S, but all of the J$_i$(x) (1$\leqslant i \leqslant$n) and C(x) are not contained in S.
Action:
S=S$\cup${C(x)}

FIG. 5.1. *The adapted Tableau rules used for detecting default satisfiability of $\mathcal{ALCN}$ concepts*

2) In the third step, the action condition is that {P(x)}$\subseteq$S, S doesn't contain all of the J$_i$(x) (1$\leqslant i \leqslant$n) and no other transformation rules can be applied. If the constraint set S before the action is satisfiable, then there exists an interpretation $\mathcal{I}$ such that $\mathcal{I}$ satisfies all of elements of S. Because {P(x)}$\subseteq$S, then $\mathcal{I}$ satisfies P(x). Furthermore, we know that $\mathcal{I}$ doesn't satisfy any J$_i$(x) (1$\leqslant i \leqslant$n), otherwise, there would exist other exceptional rules which can be applied. Because $\mathcal{I}$ satisfies P(x) before the action. So from Definition 4.1, we get $\mathcal{I}$ satisfies C(x). Because $\mathcal{I}$ satisfies both S and C(x), we get $\mathcal{I}$ satisfies S$\cup${C(x)}.

From above proofs, we can conclude that every action in the applied transform rules, in the extension algorithm, preserves satisfiability. □

As mentioned in Definition 4.4, an EDDT embraces three types of transformation rules: strict information, fulfilled information and exceptional information. These different types of information are given different levels of priority. Here, we use the symbol SR to denote the set of strict facts in an EDDT, FR to denote the set of fulfilled information and ER to denote the set of exceptional information. Then, based on the EDDT shown in Figure 4.3, we will get the descriptions of its sets of different types of information in NNF, where

Algorithm: checking default satisfiability of C based on the EDDT

```
Require: An EDDT which embraces SR, FR and ER.
Ensure: the descriptions of SR, FR and ER in NNF.
1. S0=C(x), i=1;
2. apply strict rules and transform S0 into Si;
3. for each r∈ER do                    //Step 1
4.    if Si meets the condition of r
5.       apply r to Si and result of action: Si+1←Si;
6.       i=i+1;
7.       if there exist clashes in Si
8.          return "C is satisfiable";
9.       end if
10.   end if
11. end for
12. for each r∈SR do                   // Step 2
13.   if Si meets the condition of r and Si isn't labeled "Clash"
14.      apply r to Si and result of action: Si+1←Si;
15.      i=i+1;
16.      if there exist clashes in Si
17.         Si is labeled "Clash";
18.      end if
19.   end if
20. end for
21. for each r∈FR do                   // Step 3
22.   if Si meets the condition of r and Si isn't labeled "Clash"
23.      apply r to Si and result of action: Si+1←Si;
24.      i=i+1;
25.      if there exist clashes in Si
26.         Si is labeled "Clash";
27.      end if
28.   end if
29. end for
30. if the leaf nodes of all possible branches in the constructed tree-like model are labeled "Clash"
31.    return "C is satisfiable";
32. else return "C is unsatisfiable";
33. end if
```

$$SR = \{\neg PARROT \sqcup BIRD, \neg SPARROW \sqcup BIRD,$$
$$\neg PARROT \sqcup FLYING\_ANIMAL, \neg GOAT \sqcup \neg SPEAKING\_ANIMAL\}$$
$$FR = \{\neg BIRD \sqcup \neg SPEAKING\_ANIMAL\}$$
$$ER = \{\neg PARROT \sqcup SPEAKING\_ANIMAL\}.$$

The subsumption assertions to be checked should be transformed into their negation description in NNF according to the theorem [10]: $A \sqsubseteq B$ is satisfiable iff $A \sqcap \neg B$ is unsatisfiable, where A and B are concept descriptions, respectively. For example, the subsumption assertion $SPARROW \sqsubseteq \neg SPEAKING\_ANIMAL$ will be transformed into the concept description with negation $SPARROW \sqcap SEAKING\_ANIMAL$. In the following, we will describe particularly the extension algorithm for checking default satisfiability of a given concept. The default extension algorithm can be divided into three steps. In the first step, we apply exceptional rules to constraint set because they have the highest priority. If exceptional rules can be used for the detected concept, strict rules will not be used. Otherwise, if no exceptional rules can be used, the strict rules can be applied to constraint set (step 2). The reason why we do like this is to avoid conflicting with some strict information. Another reason is to save reasoning time. In step three, only in the situation that no other strict information can be used, could fulfilled rules be used. The default extension algorithm either stops because all actions fail with obvious conflicts, or it stops without further used rules.

The following example shown in Figure 5.2 demonstrates the algorithm with a tree-like diagram. We want to know whether the subsumption assertion $SPARROW \sqsubseteq \neg SPEAKING\_ANIMAL$ is satisfiable in the EDDT shown in Figure 4.3. That is to say, we should detect that the concept $SPARROW \sqcap SPEAKING\_ANIMAL$ is unsatisfiable. The concept is firstly transformed into constrain set S_0. Considering the default rule $BIRD(x)$:

$PARROT(x)/SPEAKING\_ANIMAL(x)$, we know that $PARROT(x)$ isn't contained in S_0, Then, in the first step, the exceptional rule $\neg PARROT(x) \sqcup SPEAKING\_ANIMAL(x)$ can not be applied to S_0. In the following steps, we apply strict rules, the reasoning continues until it stops with obvious conflicts. Finally, the leaf node of every branch in this tree-like diagram is notated using "Clash" tag. So we know the constraint $SPARROW \sqcap SPEAKING\_ANIMAL$ are not satisfiable. That is to say, the subsumption assertion $SPARROW \sqsubseteq \neg SPEAKING\_ANIMAL$ is satisfiable.

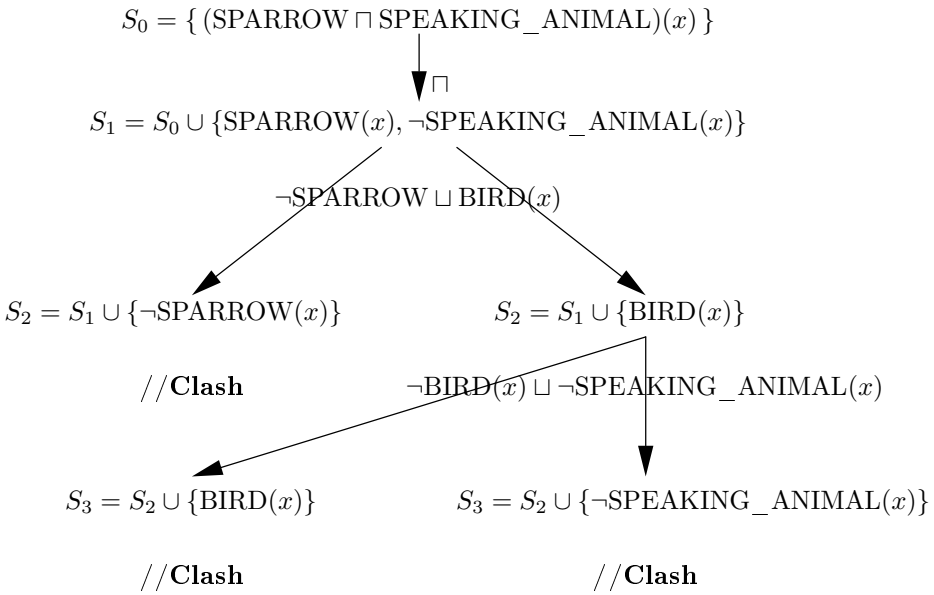

FIG. 5.2. *Detecting default satisfiability of complex concept*

Please note that the extension algorithm can tackle both general subsumption assertions and assertions about exceptional facts. In another example shown in Figure 5.3, we want to check whether the subsumption assertion $PARROT \sqsubseteq SPEAKING\_ANIMAL$ is satisfiable, that is to say, we check the default satisfiability of the concept $PARROT(x) \sqcap \neg SPEAKING\_ANIMAL(x)$, which transformed into a constrain set. In the first step, when the exceptional rule $\neg PARROT(x) \sqcup SPEAKING\_ANIMAL(x)$ is applied to constraint set, the complete conflicts occur. So we know the concept $PARROT(x) \sqcap \neg SPEAKING\_ANIMAL(x)$ is not satisfiable, which means that the subsumption assertion $PARROT \sqsubseteq SPEAKING\_ANIMAL$ is satisfiable. Then reasoning process stops without applying other transformation rules. This can be served as an example of reasoning for an exceptional fact.

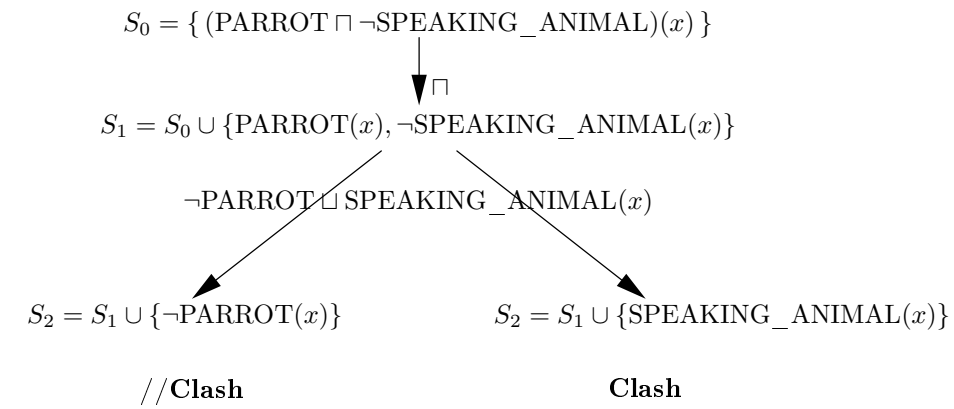

FIG. 5.3. *An example of detecting exceptional fact*

In the following, we give a brief of discussion of complexity issues about the default satisfiability algorithm.

THEOREM 5.3. *Default satisfiability of $\mathcal{ALCN}$-concept descriptions is PSPACE-complete.*

Proof. From [16], we know that satisfiability of $\mathcal{ALCN}$-concept descriptions is PSPACE-complete. As mentioned above, our default satisfiability algorithm for $\mathcal{ALCN}$-concept descriptions can be divided into three steps. In fact, every step is just the satisfiability algorithm for $\mathcal{ALCN}$. Then the sequence of the three steps is also essentially the satisfiability algorithm for $\mathcal{ALCN}$. So we get the conclusion that default satisfiability of $\mathcal{ALCN}$-concept descriptions is PSPACE-complete. □

6. Related work and Discussions. In the description logics community, a number of approaches to extend description logics with default reasoning have been proposed. Baader and Hollunder [17] investigated the problems about open default in detail and defined a preference relation. The approach is not restricted to simple normal default. Two kinds of default rules were introduced by Straccia [18]. The first kind is similar to the fufilled rules in our approach. The second kind of rules allows for expressing default information of fillers of roles. Lambrix [19] presented a default extension to description logics for use in an intelligent search engine, Dwebic. Besides the standard inferences, Lambrix added a new kind of inference to description logic framework to describe whether an individual belongs to a concept from a knowledge base. Calvanese [20] proposed a formal framework to specify the mapping between the global and the local ontologies. Maedche [21] also proposed a framework for managing and integrating multiple distributed ontologies. Stuckenschmidt [6] exploited partial shared ontologies in multi-agent communication using an approximation approach of rewriting concepts. However, default information was not considered in these different frameworks and systems. An important feature of our formal framework distinguished from other work is that our default extension approach is based on DDL. To our best knowledge, little work has been done to pay attention to default extension to DDL for communication among agents.

There is an alternative proposal for dealing with the problem of the example shown in Figure 2.1. For example, if the term SPARROW instead of BIRD in ontology 1 is mapped into the term

NON_SPEAKING_ANIMAL

in ontology 2, and the term PARROT in ontology 1 is not mapped into the term NON_SPEAKING_ANIMAL, then there is no default information to be considered. It seems that we have avoided the problem of default information between the two ontologies using the inter-ontology mapping. However, in fact, this approach is exhausted and unscalable. If there are a lot of terms belonging to the subclasses of BIRD to be added into ontology 1, we have to map every one of these added terms into NON_SPEAKING_ANIMAL in ontology 2. In the situation, we will find the alternative approach is much exhausted and unscalable. In contrast to the alternative approach, our default extension approach to DDL considers the inter-ontology mapping efforts and the scalability of ontologies used by different agents as key features.

Regarding to the complexity issue of the proposed default satisfiability algorithm, we will find that the algorithm increase no more complexity than satisfiability algorithm for $\mathcal{ALCN}$. It means that we can perform reasoning with strict information as well as default information in the same time and space complexity. The future work includes a flexible mechanism for parsing exchanged messages among agents. ACLs are used to construct and parse exchanged messages required by both participants. Then, concepts defined in DAML+OIL ontology language can be readily combined with the mechanism, thus increasing the flexibility of messages, and hence accessibility and interoperability of services within open environments.

7. Conclusion. In this paper, an approach is proposed to enables agents using different ontologies on the Web to exchange semantic information solely relying on internally provided mapping between the ontologies. Because of the semantic heterogeneity among these ontolgies, it is difficult for an agent to understand the terminology of another agent. To get complete and correct semantic information from multiple ontologies used by different agents, default information among these ontologies should be considered. Our approach is based on default extension to DDL. The distributed terminological knowledge base is originally used to present strict information. To perform default reasoning based on DDL, strict as well as default information is taken into account. Then, all of default information above is added into an extended default distributed terminological knowledge base (EDDT), which is constructed from a default distributed terminological knowledge base (DDT). The default Tableau algorithm is used on EDDT where different rules have different priority: exceptional rules have the highest priority, and fulfilled rules the least. Reasoning with default information provides agents using different ontologies with stronger query capability. In our opinion, a query based on DDT can boil down to checking default satisfiability of complex concept in accord with the query.

Our approach enables agents using different ontologies on the Web to exchange semantic information solely relying on internally provided mapping between the ontologies. But so far, our approach is considered as a basic mechanism for facilitating agent communication. To apply it in practice, there is still a lot of work to be done [23]. For example, more sophisticated agent communication protocols, similar to KQML [22] and FIPA [24], have to be developed for getting complete and correct information through agents. Using the communication protocols, concepts defined in DAML+OIL ontology language can be readily combined with the mechanism, thus increasing the flexibility of messages, and hence accessibility and interoperability of services within open environments.

Acknowledgments. This research is partially supported by the National Grand Fundamental Research Program of China under Grant No.TG1999035805, 2002CB312005, the Chinese National "863" High-Tech Program under Grant No.2001AA113010.

REFERENCES

[1] R. T. Payne, et al, *Communicating Agents in Open Multi Agent Systems*, In First GSFC/JPL Workshop on Radical Agent Concepts (WRAC'02), 2002.
[2] V. Damjanovic, et al, *Web Agent's Enclaves—A New Opportunity for the Semantic Web Services*, In Proceedings of WWW2004, 2004.
[3] T. Berners-Lee, et al, *The Semantic Web*, Scientific American, 284(5), (2001), pp. 34–43.
[4] T. Berners-Lee, *The Semantic Web—XML2000*, 2000. `http://www.w3.org/2000/Talks/1206-xml2k-tbl/`
[5] J. Hendler, *Agents and the Semantic Web*, IEEE Intelligent Systems, 16(2), (2001), pp.30–37.
[6] H. Stuckenschimdt, *Exploiting Partially Shared Ontologies for Multi-Agent Communication*, In Proceedings of CIA'02, 2002.
[7] M. Uschold, Barriers to Effective Agent Communications, In Proceedings of Ontologies in Agent Systems (OAS'01), 2001.
[8] M. Klein, *Combining and relating ontologies: an analysis of problems and solutions*, In Proceedings of IJCAI'01 Workshop: Ontologies and information sharing, Seattle, USA, 2001.
[9] H. Wache, et al, *Ontology-Based Integration of Information-A Survey of Existing Approaches*, In Proceedings of IJCAI'01 Workshop: Ontologies and Information Sharing. USA, (2001) pp. 108–117.
[10] F. Baader, et al, *The Description Logic Handbook: Theory, Implementation and Applications*, Cambridge University Press, Cambridge, 2003.
[11] F. Baader, et al, *Description logics as ontology languages for the semantic web*, Lecture Notes in Artificial Intelligence, Springer-verlag, 2003.
[12] I. Horrocks, et al, *Reviewing the design of DAML+OIL: language for the Semantic Web*, In Proceedings of AAAI'02, Canada, (2002), pp. 792–797.
[13] W3C, *OWL Web Ontology Language Reference*, 2004. `http://www.w3.org/TR/owl-ref/`
[14] A. Borgida, et al, *Distributed Description Logics: Directed Domain Correspondences in Federated Information Sources*, Journal of Data Semantics, l,1, Springer Verlag, (2003) pp. 153–184.
[15] D. Connolly. et al, *DAML+OIL (March 2001) Reference Description*, 2001. `http://www.w3.org/TR/daml+oil-reference`.
[16] F. Baader and U. Sattler, *An Overview of Tableau Algorithms for Description Logics*, In Proceeding of Tableaux2000, 2000.
[17] F. Baader and B. Hollunder, *Embedding Defaults into Terminological Representation Systems*, Journal of Automated Reasoning, 14, (1995) pp. 149–180.
[18] U. Straccia, *Default Inheritance Reasoning in Hybrid KL-ONE-style Logics*, In Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI'93), (1993) pp. 676–681.
[19] P. Lambrix, et al, *A Default Extension to Description Logics for use in an Intelligent Search Engine*, In proceedings of 31st Annual Hawaii International Conference on System Sciences, 5, USA, (1998) pp. 28–35.
[20] D. Calvanese, et al, *A Framework for ontology Integration*, In I. Cruz, S. Decker, J. Euzenat, and D. McGuinness, (eds.), The Emerging Semantic Web _ Selected Papers from the First Semantic Web Working Symposium, IOS Press, (2002) pp. 201–211.
[21] A. Maedche, et al, *Managing multiple and distributed ontologies on the Semantic Web*, In The VLDB Journal-Digital Object Identifier (DOI), 12, (2003) pp. 286–302.
[22] T. Finin, et al, *KQML as an agent communication language*, In Proceeding of the third Conference on Information and Knowledge Management (CIKM'94), ACM press, 1994.
[23] L. Gasser, MAS infrastructure definitions, needs and prospects, In Proceedings of ICMAS2000, 2000.
[24] FIPA, *Foundation for Intelligent Physical Agents*, 2004. `http://www.fipa.org/`

Edited by: Shahram Rahimi, Raheel Ahmad
Received: October 10, 2005
Accepted: March 19, 2006