Journal of Southeast University (English Edition) Vol. 25, No. 4, pp. 477 – 481 Dec. 2009 ISSN 1003—7985

Updating ontologies and general rules

Zhang Xiaowang[1, 2] Xiao Guohui[1] Lin Zuoquan[1]

([1]Department of Information Science, Peking University, Beijing 100871, China)

([2]School of Mathematical Sciences, Anhui University, Hefei 230039, China)

Abstract: Inconsistencies or conflicts appearing in the integration of ontologies and general rules are handled by applying prioritizing and updating. First, a prioritized knowledge base is obtained by weighting information weight. Then, based on the idea "abandoning the old for the new", the weight of each rule is greater than that of the information in ontologies. If ontologies conflict with general rules, then a new knowledge-base without any inconsistency or conflict is obtained by using rules with big weight updating information in ontologies with small weight. Thus, current logic programming solvers and description logic reasoners are employed to implement the reasoning services, such as querying etc. Updating based on prioritizing is more suitable for handling inconsistencies than other approaches to introducing non-standard semantics if knowledge bases are dynamically evolving. Moreover, a consistent knowledge base can be always maintained in the dynamical environment by updating outdated information with new information based on weighting. Finally, this approach to dealing with inconsistencies is feasibly exemplified.

Key words: semantic web; rule; inconsistency handling; updating

The semantic web is an extension of the current web by introducing standards and technologies that help machines to understand the information on the web so that they can support richer discovery, data integration, navigation, and automation of tasks[1]. The semantic web is conceived in hierarchical layers, where the ontology layer is in the form of the web ontology language (OWL) and the rule layer offers sophisticated representation and reasoning capabilities. However, the ontology layer of the semantic web is quite developed, and the rule layer is far less developed[2]. Based on the input from the semantic web rules community, the semantic web architecture has been recently reconsidered by Berners-Lee et al[1]. Ontologies and rules are now sitting side by side between RDF(s) and a unifying logic layer. The problem of adding rules to ontologies is currently a hot research topic in the semantic web community due to the interest of the semantic web applications towards the integration of rule-based systems with ontologies[3]. Mei et al. [4] presented an approach to integrating DLs with more general rules called $\mathrm{ALC}_{\mathrm{P}}^{\mathrm{U}}$. The $\mathrm{ALC}_{\mathrm{P}}^{\mathrm{U}}$ extends a DL-based knowledgebase, consisting of a TBox $\mathscr{T}$ of subsumptions and an ABox $\mathscr{A}$ of assertions with a PBox P of general rules, i. e., Datalog $\neg$ rules permitting default negation for atoms in the body, in a homogeneous manner. That is to say, a PBox of general rules sharing predicates with the DL concepts and DL roles is added in order to introduce the nonmonotonic reasoning mechanism to ontologies in $\mathrm{ALC}_{\mathrm{P}}^{\mathrm{U}}$. For its open answer set semantics, extended Herbrand structures are used to interpret DL concepts and DL roles, while open answer sets hold for general rules. The primary characteristics of $\mathrm{ALC}_{\mathrm{P}}^{\mathrm{U}}$ are the unifying predicates of DL and the concepts or roles of DLs. It facilitates reasoning in DLs by many inference engines in LP. Unfortunately, an $\mathrm{ALC}_{\mathrm{P}}^{\mathrm{U}}$KB does not do anything when facing conflicting or contradictory information. For instance, given $\mathrm{ALC}_{\mathrm{P}}^{\mathrm{U}}$ KB $\mathscr{K}=(\mathscr{T}, \mathscr{A}, P)$, where TBox $\mathscr{T}=\{\text{Bird} \sqsubseteq \text{Fly}, \text{Penguin} \sqsubseteq \text{Bird}\}$, ABox $\mathscr{A}=\{\text{Bird}(\text{Tweety}), \text{Penguin}(\text{Tweety})\}$, PBox $P=\{\text{Fly}(x) \leftarrow \text{Bird}(x), \text{not Penguin}(x), \neg\text{Fly}(x) \leftarrow \text{Penguin}(x)\}$, the answer set S of $\mathscr{K}$ is $\{\text{Bird}(\text{Tweety}), \neg\text{Fly}(\text{Tweety}), \text{Fly}(\text{Tweety})\}$. Clearly, S is inconsistent. Practically, it is not reasonable that S entails anything by the trivial inference. The main reason of the trivial reasoning is the existence of inconsistent information $\neg$ Fly (Tweety) and Fly (Tweety).

Not surprisingly, the study of dealing with inconsistent ontologies with rules becomes more and more important. Roughly, there are two fundamentally different approaches to handling inconsistencies in both DLs and LP. The first approach, called the paraconsistent approach, applies a nonstandard reasoning method to obtain meaningful answers[5-7]. The other approach is repairing inconsistencies in order to obtain a consistent ontology by applying techniques such as belief revision, prioritizing and updating etc[8-9]. Techniques of prioritizing and updating are widely applied in dealing with inconsistencies in answer set programming[10].

In this paper, according to the second approach, we employ prioritizing and updating to handle inconsistencies or conflicts occurring from the integration of ontologies and general rules in $\mathrm{ALC}_{\mathrm{P}}^{\mathrm{U}}$. The main innovations and contributions of this work can be summarized in the following aspects.

1) We define a prioritized dl-program based on the prioritizing technique, which is presented to treat ontologies as rules with weight from the view logic programming during the process of reasoning.

2) We introduce a prioritized $\mathrm{ALC}_{\mathrm{P}}^{\mathrm{U}}$ based on the prioritized dl-program to provide a strict partial ordering between two dl-rules to weight different values to different dl-rules. Thus those conflicts occurring between ontologies and rules can be handled in the prioritized $\mathrm{ALC}_{\mathrm{P}}^{\mathrm{U}}$.

3) We employ the updating technique to handle inconsistencies between ontologies and general rules in $\mathrm{ALC}_{\mathrm{P}}^{\mathrm{U}}$. The main principle of updating for ontologies and rules is "abandoning the old for the new". That is to say, the latest infor-

Received 2009-06-30.

Biographies: Zhang Xiaowang (1980—), male, graduate; Lin Zuoquan (corresponding author), male, doctor, professor, lzq@is.pku.edu.cn.

Foundation item: The National Natural Science Foundation of China (No. 60973003).

Citation: Zhang Xiaowang, Xiao Guohui, Lin Zuoquan. Updating ontologies and general rules[J]. Journal of Southeast University (English Edition), 2009, 25(4): 477 – 481.

mation would be accepted and other outdated information should be discarded based on the partial ordering among three units, namely, an ABox, a TBox and a PBox.

4) We apply the updating technique to deal with inconsistencies occurring among an ABox, a TBox and a PBox by updating an ABox with a PBox and updating a TBox with a PBox, based on facts that ① a PBox is on the top layer of an ABox and a TBox; and ② the knowledge of a PBox is newer than that of an ABox or a TBox based on the idea "abandoning the old for the new".

1 Preliminaries

$\mathrm{ALC}_{\mathrm{P}}^{\mathrm{U}}$ presented by Mei et al[4]. is an approach to integrating ontologies with general rules, which enables the newly envisioned unifying logic on top of ontologies and rules in the semantic web. For comprehensive background reading, please refer to Ref. [4].

Let I be a finite set of named individuals, and $V=\{x, y, z, \ldots\}$ a countable set of variables. A (Datalog) term is a named individual or a variable. Let N_{C} be a set of concept names and N_{R} a set of role names. The set Σ_R of ALC-roles is N_{R}. The set Σ_C of ALC-concepts is the smallest set such that 1) The top concept $\top$ and the bottom concept $\perp$ are ALC-concepts; 2) Every concept name in N_C is an ALC-concept; and 3) If C, C_1, C_2 are ALC-concepts and R is an ALC-role, then $\neg C$, $C_1 \sqcap C_2$, $C_1 \sqcup C_2$, $\exists R.C$, $\forall R.C$ are also ALC-concepts.

We say that a concept is in the negation normal form (NNF) if classical negation "$\neg$" occurs only in front of atomic concepts. A dl-literal is the form L or $\neg L$, where L is an assertion in DLs. Lit denotes a set of all dl-literals. A rule is a dl-rule if the rule has the form of $L_0 \leftarrow L_1, \ldots, L_m$, not $L_{m+1}, \ldots,$ not L_n where $L_i (0 \leqslant i \leqslant n)$ is a dl-literal in DLs. We refer to the L_0 as the head of r, denoted by Head(r), while the conjunction $L_1, \ldots, L_m$, not $L_{m+1}, \ldots,$ not L_n is called the body of r, denoted by Body$(r)=\{L_1, \ldots, L_m, \text{not } L_{m+1}, \ldots, \text{not } L_n\}$. Body$^+(r)=\{L_1, \ldots, L_m\}$ and Body$^-(r)=\{L_{m+1}, \ldots, L_n\}$. A set of dl-rules is called a dl-program, denoted by Π. Given a concept C, clos(C) is the smallest set that contains C and is closed under subconcepts and negation (in NNF). For a set of concepts Σ, clos$(\Sigma)=\bigcup_{C\in\Sigma}$clos$(C)$. An $\mathrm{ALC}_{\mathrm{P}}^{\mathrm{U}}$ KB has the form $\mathscr{K}=(\mathscr{T}, \mathscr{A}, P)$, where 1) TBox $\mathscr{T}$: Subsumptions are $C_1 \sqsubseteq C_2$ with $C_1, C_2 \in \Sigma_C$; 2) ABox $\mathscr{A}$: Assertions are $C(a)$ or $R(a, b)$ with $C \in \Sigma_C$, $R \in \Sigma_R$, and $a, b \in I$; and 3) PBox $\mathscr{P}$: dl-rules are $r, L_0(u) \leftarrow L_1(v_1), \ldots, L_m(v_m)$, not $L_{m+1}(v_{m+1}), \ldots,$ not $L_n(v_n)$ with $L_i \in \Sigma_C \cup \Sigma_R$, and $\boldsymbol{u}$, $\boldsymbol{v}_i$ are vectors of terms in $I \cup V$, for each $0 \leqslant i \leqslant m \leqslant n$ (Each vector has length 1 or 2 since concepts from Σ_C become unary predicates and roles from Σ_R become binary predicates).

Syntactically, dl-rule variables in PBox P are merely required to satisfy the most general Datalog safeness condition[2, 4].

Given a function-free first-order language $\mathscr{L}$, an $\mathscr{L}$-structure is a pair $\mathscr{J}=\langle U, I\rangle$, where the universe $U=\langle D, \sigma\rangle$ consists of a non-empty domain D and a function σ: $I\cup D' \rightarrow D$ which assigns a domain value to each individual, and $\sigma(d)=d$ for all $d \in D'$, given $I \cap D \neq \varnothing$.

Elements of D' are called unnamed individuals. We remark that the corresponding definitions in Ref. [2] are not clear, where σ: $I \cup D$ and any $d \in D$ is defined as an unnamed individual if there is no $i \in I$ such that $\sigma(i)=d$. Using σ, we formalize UNA, PNA, and SNA as follows: in the case that σ is injective, the UNA applies; in the case that D' is empty, the PNA applies; the SNA is exactly the combination of UNA and PNA.

Let I be an $\mathscr{L}$-interpretation over D, which assigns a relation $p^I \subseteq D^n$ to each n-ary predicate symbol p(here $n>1$). Answer set semantics is usually defined in terms of a Herbrand structure that has a fixed universe, namely the Herbrand universe $H=(I, \text{id})$, where id: $I \rightarrow I$ is the identity function. Obviously, by I and id, the SNA is applied here. An extended Herbrand structure $\mathscr{J}=\langle(D, \text{id}), I\rangle$ is defined for a set of named individuals I, a set of concepts Σ_C and a set of roles Σ_R, where 1) id: $I \cup D' \rightarrow D$ and id$(d)=d$ for all $d \in I \cup D'$, given $I \cup D \neq \varnothing$; 2) I: $\Sigma_C \rightarrow 2^D$ for concepts and I: $\Sigma_R \rightarrow 2^{D\times D}$ for roles such that for concepts $C, C_1, C_2 \in \Sigma_C$ and roles $R \in \Sigma_R$, the following are satisfied: $\top^I=D$, $\perp^I=\varnothing$, $(\neg C)^I=D\backslash C^I$, $(C_1 \sqcap C_2)^I=C_1^I \cap C_2^I$, $(C_1 \sqcup C_2)^I=C_1^I \cup C_2^I$, $(\exists R.C)^I=\{e_1 \in D \mid \exists e_2.(e_1, e_2) \in R^I \text{ and } e_2 \in C^I\}$, $(\forall R.C)^I=\{e_1 \in D \mid \forall e_2.(e_1, e_2) \in R^I \text{ implies } e_2 \in C^I\}$.

An associated valuation v^I of interpretation I over D is a mapping, s.t., $v^I(C(d))=$ true, if $d \in C^I$, where $C \in \Sigma_C$ and $d \in D$; $v^I(R(d_1, d_2))=$ true, if$(d_1, d_2) \in R^I$, where $R \in \Sigma_R$ and $d_1, d_2 \in D$.

An extended Herbrand structure $\mathscr{J}$ satisfies a TBox $\mathscr{T}$ if $C_1^I \subseteq C_2^I$ for all $C_1 \sqsubseteq C_2$ in $\mathscr{T}$, where $C_1, C_2 \in \Sigma_C$. Such a structure $\mathscr{J}$ is called a model of $\mathscr{T}$, written as $\mathscr{J} \models \mathscr{T}$. An extended Herbrand structure $\mathscr{J}$ satisfies an ABox $\mathscr{A}$, if id$(a)=a \in C^I$ and (id(a_1), id$(a_2))=(a_1, a_2) \in R^I$ for all $C(a)$ and $R(a_1, a_2)$ in $\mathscr{A}$, where $C \in \Sigma_C$, $R \in \Sigma_R$ and $a, a_1, a_2 \in I$. Such a structure $\mathscr{J}$ is called a model of $\mathscr{A}$, written as $\mathscr{J} \models \mathscr{A}$. P_{g} is the result of grounding P. $\lambda(P_{\mathrm{g}})$ is the least extended Herbrand models of P_{g} in which none of the dl-rules contains "not". $\Gamma(P_{\mathrm{g}}, S)$ is a set of dl-rules after Gelfond-Lifschitz transformation[11] for P_{g} with general dl-rules. S is an open answer set of P_{g} if $S=\lambda(\Gamma(P_{\mathrm{g}}, S))$. An extended Herbrand structure $\mathscr{J}$ is a model of P, written by $\mathscr{J} \models P$, if the set $S=\{C(d) \mid v^I(C(d))=\text{true}\} \cup \{R(d_1, d_2) \mid v^I(R(d_1, d_2))=\text{true}\}$ is an answer set of P_{g}. An extended Herbrand structure $\mathscr{J}$ satisfies an $\mathrm{ALC}_{\mathrm{P}}^{\mathrm{U}}$ KB $\mathscr{K}=(\mathscr{T}, \mathscr{A}, P)$ if $\mathscr{J}$ is a model of $\mathscr{T}$, $\mathscr{A}$ and P. Such a structure $\mathscr{J}$ is called a model of $\mathscr{K}$, written by $\mathscr{J} \models \mathscr{K}$. A KB $\mathscr{K}$ is satisfiable if there is a model of $\mathscr{K}$.

2 Prioritized $\mathrm{ALC}_{\mathrm{P}}^{\mathrm{U}}$

In this section, we transform ontologies into $\mathrm{ALC}_{\mathrm{P}}^{\mathrm{U}}$ and assign different weights to different dl-rules. The main idea of the introduction of a prioritized $\mathrm{ALC}_{\mathrm{P}}^{\mathrm{U}}$ is employing the logic programming reasoners, most of which are efficient, to implement the reasoning with ontologies.

It is known that each subsumption in the TBox $\mathscr{T}$ can be rewritten as a dl-rule[11]. Let $\mathscr{K}'$ be$(\varnothing, \mathscr{A}, P^{\mathscr{T}})$, where $P^{\mathscr{T}}=P \cup P_{\mathscr{T}}$, and $P_{\mathscr{T}}=\{C_2(x) \leftarrow C_1(x) \mid C_1 \sqsubseteq C_2 \in \mathscr{T}\}$. Clearly, $P_{\mathscr{T}}$, P and $P^{\mathscr{T}}$ are $\mathrm{ALC}_{\mathrm{P}}^{\mathrm{U}}$ KBs. Moreover, by appending the rules: $C_1 \sqcap C_2(x) \to C_1(x)$, $C_2(x)$; $C_1 \sqcup C_2(x) \to C_1(x)$; $C_1 \sqcup C_2(x) \to C_2(x)$; $\exists R.\ C(x) \to C(y)$, $R(x, y)$; $C \sqcup \neg\, C(x) \to \top(x)$; $\forall R.\ \neg\, C \sqcup \exists R.\ C(x) \to \top(x)$, for each computable ALC-concept, we obtain a KB $\mathscr{K}''(\varnothing, \mathscr{A}, P^{\mathscr{T}\ \mathscr{E}})$[11], which is used to evaluate extensions of complex ALC-concepts.

In this paper, we mainly consider the updating $\mathscr{K}'$ since the $\mathrm{ALC}_\mathrm{P}^\mathrm{U}$ KB $\mathscr{K} = (\mathscr{T}\ \mathscr{A}, P)$ is equivalent to its updating $\mathscr{K}' = (\varnothing, \mathscr{A}, P^{\mathscr{T}})$ as well as $\mathscr{K}''(\varnothing, \mathscr{A}, P^{\mathscr{T}\ \mathscr{E}})$[4]. In order to simplify our work, we suppose that each ABox of $\mathrm{ALC}_\mathrm{P}^\mathrm{U}$ KB is coherent with respect to its TBox. In this work, for convenience, we assume that all assertions in ALC-ontologies have been transformed into dl-rules. In the following, we introduce prioritized $\mathrm{ALC}_\mathrm{P}^\mathrm{U}$ with adding the partial order to $\mathrm{ALC}_\mathrm{P}^\mathrm{U}$ by employing prioritizing in logic programming[10].

Definition 1 A prioritized $\mathrm{ALC}_\mathrm{P}^\mathrm{U}$ dl-program P is a triple $(\Pi, N, <)$, where Π is an $\mathrm{ALC}_\mathrm{P}^\mathrm{U}$ dl-program; N is a naming function mapping each dl-rule in Π to a name; $<$ is a strict partial ordering on names. We use $P(<)$ to denote the set of $<$-relations of P.

Definition 1 shows that a prioritized $\mathrm{ALC}_\mathrm{P}^\mathrm{U}$ dl-program P is an $\mathrm{ALC}_\mathrm{P}^\mathrm{U}$ KB with a partial ordering. In particular, if $N(r) < N(r')$ holds in P, dl-rule r would be preferred to apply over dl-rule r' during the evaluation of P.

Based on defeating rules in LP, we characterize whether a dl-rule is defeated in the following definition.

Definition 2 A dl-rule r is defeated by Π if and only if Π has an answer set and, for any answer set S of Π, $\mathrm{Body}(r) \cap S \neq 0$.

The ground instantiation fills the concepts or roles by individuals in order to make $\mathrm{ALC}_\mathrm{P}^\mathrm{U}$ only contain assertions.

Definition 3 A prioritized $\mathrm{ALC}_\mathrm{P}^\mathrm{U}$ dl-program $P = (\Pi_g, N_g, <_g)$ is the ground instantiation of $P = (\Pi, N, <)$ if 1) Π_g is the ground instantiation of Π; 2) $<_g$ is a strict partial ordering and $N_g(r_1') < N_g(r_2') \in P'(<_g)$ if and only if there exist dl-rules r_1 and r_2 in Π such that r_1' and r_2' are ground instances of r_1 and r_2, respectively, and $N(r_1) < N(r_2) \in P(<)$.

Notice that in the process of ground instantiation, we should maintain the partial order between two assertions. In the following, we will only consider a ground prioritized $\mathrm{ALC}_\mathrm{P}^\mathrm{U}$KB without explicit declaration.

Definition 4 A set of dl-rules is a reduct of P with respect to $<$, denoted by $P^{<}$, if and only if there exists a sequence of set $\Pi_i (i = 0, 1, \ldots)$ such that

1) $\Pi_0 = \Pi$;

2) $\Pi_i = \Pi_{i-1} - \{r_1, r_2, \ldots \mid$ ① There exists $r \in \Pi_{i-1}$ such that for every $j (j = 1, 2, \ldots)$, $N(r) < N(r_j) \in P(<)$ and $r_1, r_2, \ldots$ are defeated by $\Pi_{i-1} - \{r_1, r_2, \ldots\}$; ② There are no dl-rules $r', r'' \ldots \in \Pi_{i-1}$ such that $N(r_j) < N(r')$, $N(r_j) < N(r'')$, … for some $j (j = 1, 2, \ldots)$ and r', r'', … are defeated by $\Pi_{i-1} - \{r', r'', \ldots\}\}$;

3) $P^{<} = \cap_{i=0}^{\infty} \Pi_i$.

Definition 5 For any subset S of Lit, which is the set of all ground dl-literals in the language of P, S is an answer set of P if and only if S is an answer set for some reduct $P^{<}$ of P.

We say that a prioritized $\mathrm{ALC}_\mathrm{P}^\mathrm{U}$ dl-program is well defined if it has a consistent answer set. An $\mathrm{ALC}_\mathrm{P}^\mathrm{U}$ KB is consistent if all its answer sets are consistent. Based on the above basic definitions, we introduce a prioritized $\mathrm{ALC}_\mathrm{P}^\mathrm{U}$ KB.

Definition 6 An $\mathrm{ALC}_\mathrm{P}^\mathrm{U}$ KB $\mathscr{K} = (\mathscr{T}\ \mathscr{A}, P)$ is prioritized, denoted by $(\mathscr{T}\ \mathscr{A}, P, <)$, if $P^{\mathscr{T}}$ is a prioritized $\mathrm{ALC}_\mathrm{P}^\mathrm{U}$ dl-program.

3 Updating Prioritized $\mathrm{ALC}_\mathrm{P}^\mathrm{U}$

In this section, we employ updating presented by Zhang[10] for prioritized $\mathrm{ALC}_\mathrm{P}^\mathrm{U}$ to handle inconsistencies. In our view points, rules are newer than ontologies since rules are on the top of ontologies. Based on an important idea of updating that "abandoning the old for the new", rules are sounder and more reasonable than ontologies. In this paper, we develop a mechanism of updating following from this idea. In the updating mechanism, we update ABoxes with PBoxes and update TBoxes with PBoxes since PBoxes are more credible than both ABoxes and TBoxes. In the following, for convenient discussing, we mainly consider a prioritized $\mathrm{ALC}_\mathrm{P}^\mathrm{U}$ KB $\mathscr{K} = (\mathscr{T}\ \mathscr{A}, P, <)$, where $\mathscr{A}$ is coherent with respect to $\mathscr{T}$ and P is consistent.

3.1 PBoxes updating ABoxes

We denote $\mathscr{L}_\mathrm{new}$ which satisfies that for each concept or role symbol P in $\mathscr{L}$, there is a corresponding concept or role symbol New-P in $\mathscr{L}_\mathrm{new}$ with the same arity of P. $\mathrm{Lit}_\mathrm{new}$ is used to denote the set of all ground dl-literals of $\mathscr{L}_\mathrm{new}$, that is, $\mathrm{Lit}_\mathrm{new} = \mathrm{Lit} \cup \{\text{New-}L \mid L \in \mathrm{Lit}\}$, where New-$L$ denotes to corresponding dl-literal L in $\mathscr{L}$.

Definition 7 The prioritized $\mathrm{ALC}_\mathrm{P}^\mathrm{U}$ of $\mathscr{L}_\mathrm{new}$ is a specification of updating ABox $\mathscr{A}$ with PBox P, denoted by Update$(\mathscr{A}, P) = (P^*, N, <)$ as

1) P^* is composed of the following dl-rules: ① Initial knowledge dl-rules, for each L in $\mathscr{A}$, there is a dl-rule $L \leftarrow$; ② Inertia dl-rules, for each concept or role symbol P in $\mathscr{L}$, there are two dl-rules, New-$P(x) \leftarrow P(x)$, not $\neg$ New-$P(x)$ and $\neg$ New-$P(x) \leftarrow \neg\, P(x)$, not New-$P(x)$; ③ Update dl-rules, for each dl-rule $L_0 \leftarrow L_1, \ldots, L_m$, not $L_{m+1}, \ldots$, not L_n there is a dl-rule, New-$L_0 \leftarrow$New-$L_1, \ldots$, New-L_m, not New-$L_{m+1}, \ldots$, not New-L_n;

2) Naming function N assigns a unique name for each dl-rule in Π^*;

3) For any inertia dl-rule r and update dl-rule r', $N(r) < N(r')$.

Following from definition 7, when conflicts occur between inertia and update rules, inertia rules should defeat the corresponding update rules. Otherwise, the preference ordering does not play any role in the evaluation of Update $(\mathscr{A}, P)$.

Definition 8 A set $\mathscr{A}^*$ of ground dl-literals is called a possible resulting ABox with respect to Update $(\mathscr{A}, P)$, if and only if $\mathscr{A}^*$ satisfies the following conditions: ① If Update$(\mathscr{A}, P)$ has a consistent answer set, written by S, then $\mathscr{A}^* = \{L \mid \text{New-}L \in S\}$; and ② If Update $(\mathscr{A}, P)$ does not have a consistent answer set, then $\mathscr{A}^* = \mathscr{A}$. We use Res (Update$(\mathscr{A}, P)$) denote the set of all resulting KBs of Update$(\mathscr{A}, P)$.

The following property presents a close relationship between the result after updating and the answer set of its corresponding prioritized $\mathrm{ALC}_{\mathrm{P}}^{\mathrm{U}}$KB.

Theorem 1 Let $\mathcal{K} = (\mathcal{T}, \mathcal{A}, P, <)$ be a prioritized $\mathrm{ALC}_{\mathrm{P}}^{\mathrm{U}}$ KB and Update$(\mathcal{A}, P)$ be a well defined update specification. Then $\mathcal{A}^*$ results with respect to Update$(\mathcal{A}, P)$ if and only if $\mathcal{A}^*$ is an answer set of prioritized $\mathrm{ALC}_{\mathrm{P}}^{\mathrm{U}}$ $P = (P \cup \{L \leftarrow \mathrm{not}\bar{L} \mid L \in \mathcal{A}\}, N, <)$, where $N(r) < N(r')$ for each dl-rule r: $L \leftarrow L$, not $\bar{L}$ with $L \in \mathcal{A}$ and each dl-rule r' in P where $\bar{L}$ stands for the complement of dl-literal L.

Definition 9 $\mathcal{A}$ satisfies a dl-rule r if and only if facts $L'_1, \ldots, L'_m$ are in $\mathcal{A}$ and facts $L'_{m+1}, \ldots, L'_n$ are not in $\mathcal{A}$. Then fact L'_0 is in $\mathcal{A}$ for each ground instance r' of r: $L'_0 \leftarrow L'_1, \ldots, L'_m$, not $L'_{m+1}, \ldots$, not L'_n. $\mathcal{A}$ satisfies P if $\mathcal{A}$ satisfies each dl-rule in P.

We will show that every possible result of a KB after updating it with Π satisfies Π. That is, we can indeed obtain a consistent KB after updating.

Theorem 2 Let $\mathcal{K} = (\mathcal{T}, \mathcal{A}, P, <)$ be a prioritized $\mathrm{ALC}_{\mathrm{P}}^{\mathrm{U}}$ KB and $\mathcal{A}^*$ be a resulting ABox with respect to Update$(\mathcal{A}, P)$. If the update specification Update $(\mathcal{A}, P)$ is well defined, then $\mathcal{A}^*$ satisfies P.

We say a set S is coherent with an $\mathrm{ALC}_{\mathrm{P}}^{\mathrm{U}}$ KB Π if for any answer set S^* of $e(\Pi, S)$ of Π with respect to S, $S \cup S^*$ is consistent, where $e(\Pi, S)$ is viewed as a simplified $\mathrm{ALC}_{\mathrm{P}}^{\mathrm{U}}$ of Π providing that all ground dl-literals in S are true. ABox $\mathcal{A}$ is coherent with PBox P in a similar way. Clearly, $\mathcal{A}$ is coherent with P if and only if $P \cup \{L \leftarrow \mid L \in \mathcal{A}\}$ is well defined.

3.2 PBoxes updating TBoxes

In the following, we consider the case in which $P_{\mathcal{T}}$ is updated by P. First, we groundly instantiate the TBox. Then, we update the ground instantiation of TBox with PBox by applying the analogical mechanism of PBoxes updating ABoxes. Let $S_{P_{\mathcal{T}}}$ be an answer set of $P_{\mathcal{T}}$ and $S_{(P_{\mathcal{T}}, P)} \in$ Res (Update$(S_{P_{\mathcal{T}}}, P)$), which is denoted to a set of all the resulting KBs of Update $(S_{P_{\mathcal{T}}}, P)$.

Definition 10 An $\mathrm{ALC}_{\mathrm{P}}^{\mathrm{U}}$KB $\Pi_{(P_{\mathcal{T}}, P)}$ is a transformed $\mathrm{ALC}_{\mathrm{P}}^{\mathrm{U}}$ from $P_{\mathcal{T}}$ with respect to P, if $\Pi_{(P_{\mathcal{T}}, P)}$ is a maximal subset of the ground instantiation of $P_{\mathcal{T}}$ such that $S_{P_{\mathcal{T}}}$ is coherent with $\Pi_{(P_{\mathcal{T}}, P)}$.

We call $P_{\mathcal{T}}$ coherent with P if for any answer set S of $P_{\mathcal{T}}$, S is coherent with P. Next, we define the update operator called P-Update for a prioritized $\mathrm{ALC}_{\mathrm{P}}^{\mathrm{U}}$KB.

Definition 11 A specification of updating $P_{\mathcal{T}}$ with P is specified as the prioritized $\mathrm{ALC}_{\mathrm{P}}^{\mathrm{U}}$ P-Update $(P_{\mathcal{T}}, P) = (\Pi_{(P_{\mathcal{T}}, P)} \cup P, N, <)$ where there is a preference relation $N(r) < N(r')$ for each dl-rule r in P and each dl-rule r' in $\Pi_{(P_{\mathcal{T}}, P)}$.

Definition 12 An $\mathrm{ALC}_{\mathrm{P}}^{\mathrm{U}}$KB $P_{\mathcal{T}}^*$ is a possible resulting $\mathrm{ALC}_{\mathrm{P}}^{\mathrm{U}}$ of P-Update$(P_{\mathcal{T}}, P)$ after updating $P_{\mathcal{T}}$ with P if $P_{\mathcal{T}}^*$ is a reduct of the ground instantiation of P-Update$(P_{\mathcal{T}}, P)$.

In the following theorem, we present some good properties of the answer sets of a resulting $\mathrm{ALC}_{\mathrm{P}}^{\mathrm{U}}$ of P-Update$(P_{\mathcal{T}}, P)$.

Theorem 3 Let $\mathcal{K} = (\mathcal{T}, \mathcal{A}, P, <)$ be a prioritized $\mathrm{ALC}_{\mathrm{P}}^{\mathrm{U}}$ KB, P-Update$(P_{\mathcal{T}}, P)$ be an update specification and $P_{\mathcal{T}}^*$ be a resulting $\mathrm{ALC}_{\mathrm{P}}^{\mathrm{U}}$ of P-Update$(P_{\mathcal{T}}, P)$. The following properties hold:

1) If Head$(P_{\mathcal{T}}) \cap$ Body$(P) = \emptyset$, then each answer set of $P_{\mathcal{T}}^*$ is also a result with respect to $S_{(P_{\mathcal{T}}, P)}$, where $S_{P_{\mathcal{T}}} = \{L \mid L \leftarrow \in P_{\mathcal{T}}\}$.

2) For each answer set S^* of $P_{\mathcal{T}}^*$, we have $S \subseteq S^*$, where S is an answer set of P.

3) $P_{\mathcal{T}}^*$ does not include any ground dl-rules, which are ground instances of some dl-rules in $P_{\mathcal{T}}$ of the form $L \leftarrow \ldots$, not L', $\ldots$, where L' is included in every answer set P.

4 Reasoning with Updating

Let $\mathcal{K} = (\mathcal{T}, \mathcal{A}, P, <)$ be a prioritized $\mathrm{ALC}_{\mathrm{P}}^{\mathrm{U}}$ KB and Q a query $\{p_1, \ldots, p_n\}$. A conjunctive query(CQ) Q over an $\mathrm{ALC}_{\mathrm{P}}^{\mathrm{U}}$ KB $\mathcal{K}$ is of the form being $\{p_1(\omega_1), \ldots, p_n(\omega_n)\}$, where p_i is either a concept or a role in DLs, and $\boldsymbol{\omega}_i$ is a (unary, binary) vector of terms, for each $1 \leqslant i \leqslant n$. A union of conjunctive queries(UCQ) Q' over an $\mathrm{ALC}_{\mathrm{P}}^{\mathrm{U}}$ KB $\mathcal{K}$ is of the form being $Q_1(\omega_1) \vee \ldots \vee Q_n(\omega_m)$, where Q_i is a CQ for each $1 \leqslant i \leqslant m$. Q is said to be a query, whether Q is a CQ or a UCQ.

Given a query Q and an extended Herbrand structure $\mathcal{J} = \langle (D, \mathrm{id}), I \rangle$, the variable substitution with respect to Q and $\mathcal{J}$ is $\theta = \{x_1/t_1, \ldots, x_n/t_n\}$, which substitutes each variable $x_i \in V$ appearing in Q with a/an (un)named individual $t_i \in D$ and $1 \leqslant i \leqslant n$. As for a CQ Q, a structure $\mathcal{J}$ is a model of Q, denoted by $\mathcal{J} \models Q$, if there is a variable substitution θ with respect to Q and $\mathcal{J}$ such that $\mathcal{J} \models p_i(\boldsymbol{\omega}_i)\theta$, i.e., applying θ to variables in p_i, where $1 \leqslant i \leqslant n$. For a UCQ Q', $\mathcal{J}$ is a model of Q', denoted by $\mathcal{J} \models Q'$, if $\mathcal{J} \models Q_j$ in Q', where $1 \leqslant j \leqslant m$. Given an $\mathrm{ALC}_{\mathrm{P}}^{\mathrm{U}}$ KB $\mathcal{K}$ and a query Q, $\mathcal{K}$ entails Q, denoted by $\mathcal{K} \models Q$, if $\mathcal{J} \models Q$ for each model $\mathcal{J}$ of $\mathcal{K}$.

Given an $\mathrm{ALC}_{\mathrm{P}}^{\mathrm{U}}$ KB $\mathcal{K}$, an interpretation I of $\mathcal{K}$ satisfies a CQ Q if and only if I can be extended to the variables in Q in such a way that $\mathcal{J}$ satisfies every term in Q. A boolean CQ is "true" with respect to $\mathcal{K}$, written by $\mathcal{K} \models Q$ if and only if every interpretation that satisfies $\mathcal{K}$ also satisfies Q. In fact, a boolean CQ is true with respect to a KB if and only if it is a logical consequence of the KB. In short, the inference problem is to decide whether $\mathcal{K} \models Q$.

We consider $P = (\emptyset, \mathcal{A}, P_{\mathcal{T}} \cup P \cup \{Q \leftarrow p_1, \ldots, p_n\}, <)$ is the corresponding dl-program P of $\mathcal{K} \cup \{Q \leftarrow p_1, \ldots, p_n\}$ and Q is a special concept or role symbol which does not occur but signs the answer to the query Q.

The query entailment problem of Q is transformed into the problem of the existence of Q in the answer set S of P.

Since the answer sets are consistent and equivalent, the following theorem holds.

Theorem 4 If an ABox $\mathcal{A}$ and a TBox $\mathcal{T}$ are coherent with a PBox P, then we obtain $(\emptyset, \mathcal{A}^*, P_{\mathcal{T}}^* \cup P, <) \models Q$ if and only if $\mathcal{K} \models Q$.

Theorem 4 easily follows from the definition of updating. When $\mathcal{A}$ or $\mathcal{T}$ is not coherent with P, there exists a consistent answer set of $(\emptyset, \mathcal{A}^*, P_{\mathcal{T}}^* \cup P, <)$ while all answer sets of$(\emptyset, \mathcal{A}, P^{\mathcal{K}}, <)$ are inconsistent. It can be easily concluded that updating ABoxes and TBoxes by PBoxes can eliminate contradictory or conflicting the information between ontologies and general rules.

Next, we reconsider the beginning example. We have $P^{\mathcal{T}} = \{\text{Fly}\leftarrow\text{Bird}, \text{Bird}\leftarrow\text{Penguin}\}$. Since there exists only one individual Tweety occurring in $\mathscr{K}$, we mainly discuss the set {Tweety}. After groundly instantiating, we obtain that {Fly(Tweety) ←Bird(Tweety), Bird(Tweety) ←Penguin (Tweety)}; {Bird(Tweety), Penguin(Tweety)}; {Fly (Tweety) ← Bird(Tweety), not Penguin(Tweety), Fly (Tweety) ← Penguin(Tweety)}.

Supposed that the weight of members of the same units (ABox, TBox, PBox) is weighted with the same value. After employing the operator *P*-Update, we obtain the answer set of resulting $S^* = \{\text{Bird(Tweety)}, \text{Penguin(Tweety)}, \neg\text{Fly(Tweety)}\}$. It shows that S^* is consistent and the exception is considered in the reasoning. We analogously discuss more general examples by employing our updating. In short, our approach is indeed employed to deal with inconsistencies or conflicts.

5 Conclusion and Future Work

In this paper, we present an approach to dealing with inconsistencies or conflicts occurring in the integration of ontologies and general rules. A mechanism of updating based on the idea "abandoning the old for the new" is introduced. Though there are many different mechanisms of updating, the technical principle of different mechanisms of updating is uniform. In fact, we provide the main technical principle of updating for ontologies and general rules in this paper. In the future work, we mainly aim at studying updating for a general ontology, whose ABox is unknown to be consistent with respect to its TBox, and general rules.

References

[1] Berners-Lee T, Hendler J, Lassila O. The semantic web [J]. *Scientific American*, 2001, **284**(5): 35-43.

[2] Eiter T, Ianni G, Lukasiewicz T, et al. Combining answer set programming with description logics for the semantic web [J]. *Artif Intell*, 2008, **172**(12/13): 1495-1539.

[3] Eiter T, Ianni G, Polleres A, et al. Reasoning with rules and ontologies [C]//*Proc of the 2nd International Summer School Reasoning Web*. Berlin: Springer-Verlag, 2006, **4126**: 93-127.

[4] Mei J, Lin Z, Boley H. $\mathrm{ALC}_{\mathrm{P}}^{\mathrm{U}}$: an integration of description logic and general rules [C]//*Proc of the 1st International Conference on Web Reasoning and Rule Systems*. Berlin: Springer-Verlag, 2007, **4524**: 163-177.

[5] Ma Y, Hitzler P, Lin Z. Algorithms for paraconsistent reasoning with OWL [C]//*Proc of the 4th European Semantic Web Conference*. Berlin: Springer-Verlag, 2007, **4519**: 399-413.

[6] Zhang X, Lin Z. Paraconsistent reasoning with quasi-classical semantic in ALC [C]// *Proc of the 2nd International Conference on Web Reasoning and Rule Systems*. Berlin: Springer-Verlag, 2008, **5341**: 222-229.

[7] Zhang X, Xiao G, Lin Z. A tableau algorithm for handling inconsistency in OWL [C]//*Proc of the 6th European Semantic Web Conference*. Berlin: Springer-Verlag, 2007, **5554**: 399-413.

[8] Huang Z, van Harmelen F, ten Teije A. Reasoning with inconsistent ontologies [C]//*Proc of the 19th International Joint Conference on Artificial Intelligence*. Denver: Professional Book Center, 2005: 454-459.

[9] Liu H, Lutz C, Milicic M, et al. Updating description logic ABoxes [C]//*Proc of the 10th International Conference on Principles of Knowledge Representation and Reasoning*. AAAI Press, 2006: 46-56.

[10] Zhang Y. Logic program-based updates [J]. *ACM Trans Comput Log*, 2006, **7**(3): 421-472.

[11] Gelfond M, Lifschitz V. The stable model semantics for logic programming [C]// *Proc of the 5th International Conference and Symposium*. Massachusetts: MIT Press, 1988: 1070-1080.

更新本体与一般规则

张小旺[1,2] 肖国辉[1] 林作铨[1]

([1] 北京大学信息科学系，北京 100871)
([2] 安徽大学数学科学学院，合肥 230039)

摘要：为了消除出现在本体与一般规则整合过程中产生的不一致或冲突，引入优先和更新技术. 首先，通过给信息赋予权值的方法得到一个带偏序的知识库. 再根据"废旧迎新"原则，设定所有规则的权值大于本体中信息的权值. 当本体与一致规则发生不一致或冲突时，权值大的规则将更新替换本体中权值小的信息，从而得到一个没有冲突的知识库. 然后，调用现有逻辑程序求解器与描述逻辑推理机来实现查询等推理任务. 与采用非标准的语义表示信息来处理不一致方法相比，基于偏序的更新技术更适合于动态进化的知识库. 同时，通过赋值可以使新来的信息替换已经过时的信息,从而使得知识库在动态进化过程中保持一致性. 最后，通过实例说明该处理不一致方法是切实可行的.

关键词：语义网；规则；不一致性处理；更新

中图分类号：TP311