An ABox abduction algorithm for the description logic ALCI

Yanwei Ma, Tianlong Gu, Binbin Xu, Liang Chang

Guangxi Key Laboratory of Trusted Software, Guilin University of Electronic Technology, Guilin 541004, China

liberty_myw520@163.com, cctlgu@guet.edu.cn, changl@guet.edu.cn

Abstract. ABox abduction is the foundation of abductive reasoning in description logics. It finds the minimal sets of ABox axioms which could be added to a background knowledge base to enforce the entailment of certain ABox assertions. In this paper, an abductive reasoning algorithm for the description logic ALCI is presented. The algorithm is an extension of an existing ABox abduction algorithm for the description logic ALC, with the feature that it is based on the Tableau of ALCI directly and do not need to use arguments and Skolem terms. It firstly transforms the ABox abduction problem into the consistency problem of knowledge base; then traditional Tableau construction process for ALCI is expanded to deal with this problem; finally the solution of the abduction problem is constructed by a process of backtracking.

Keywords: abductive reasoning, ABox abduction problem, Tableau, consistency of knowledge base

1 Introduction

In recent decades abduction has gained considerable attention in fields such as logic, artificial intelligence and computer science. It has been widely recognized that the style of reasoning, usually illustrated as an inference from puzzling observations to explanatory hypotheses, is in fact inherent in a vast majority of problem-solving and knowledge acquisition tasks. In the face of such a widespread and diverse application interest, researchers proposed many algorithms for abduction problem in the formalisms of propositional logic, first-order logic and modal logics [1,2].

In recent years, abductive reasoning in description logics become an important topic, since description logics is the logic foundation of the Web Ontology Language OWL and is playing an important role in the semantic Web. In paper [3], based on a combination of the tableau-based reasoning mechanism and the resolution-based reasoning mechanism of both the first-order logic and the modal logic, Klarman et.al. proposed an abduction algorithm for the description logic ALC. In paper [4], Elsenbroich et al. presented several application scenarios in which various forms of abduction would be useful; they also developed some formal apparatus needed to employ abductive inference in expressive description logics. In paper [5], Davenport et al.

developed a cognitive model and presented an algorithm for abductive reasoning over instances of a triples ontology. In paper [6], Jianfeng Du et al. proposed a new ABox abduction method based on reducing the original problem to an abduction problem in the logic programming.

In this paper, an abductive reasoning algorithm for the description logic ALCI is presented. This algorithm is an extension of the algorithm proposed by Klarman et.al. for the description logic ALC [3]. Besides the mechanisms for dealing with inverse roles occurring in ALCI, compared with Klarman et.al.'s work, a feature of the algorithm presented here is that the abduction process is based on the Tableau of ALCI and do not need to use a large number of arguments and Skolem terms.

2 Abduction Problem in the Description Logic ALCI

Due to the space limitation, here we omit the introduction of the description logic ALCI. The syntax and semantics of ALCI can be found in [7].

Definition 1 Let $\boldsymbol{K}$=<$\boldsymbol{T}$, $\boldsymbol{A}$> be a knowledge base of ALCI, and let Q be a concept assertion. If $\boldsymbol{K}\nvDash Q$, then we call the pair <$\boldsymbol{K}$, Q> as an ABox abduction problem of ALCI.

Definition 2 Let <$\boldsymbol{K}$, Q> be an ABox abduction problem of ALCI. If there exists an ABox E such that not only <$\boldsymbol{T}$, $\boldsymbol{A}\cup E$>$\vDash Q$ but also the following conditions holds, then we say that E is a solution for the ABox abduction problem:

(1) E is consistent with the problem, i.e., <$\boldsymbol{T}$, $\boldsymbol{A}\cup E$>$\nvDash\bot$;

(2) E is relevant with the problem, i.e., $E\nvDash Q$;

(3) E is minimal, i.e., if there is some ABox B with $K\cup B\vDash Q$, $K\cup B\nvDash\bot$, $B\nvDash Q$ and $A\vDash B$, then it must be $B\vDash A$.

3 Abductive Reasoning Algorithm for the Problem

Let <$\boldsymbol{K}$, Q> be an ABox abduction problem, where $\boldsymbol{K}$=<$\boldsymbol{T}$, $\boldsymbol{A}$>, Q is an ABox concept assertion, and $\boldsymbol{K}\nvDash Q$. Intuition of the algorithm is that the ABox abduction problem will be transformed into the consistency problem of knowledge base, then traditional Tableau construction process can be extended correspondingly to deal with the abduction problem. More precisely, since $\boldsymbol{K}\nvDash Q$, the knowledge base $\boldsymbol{K'}$=<$\boldsymbol{T}$, $\boldsymbol{A}\cup\{\neg Q\}$> must be consistent. The order of the algorithm is to find a set E of ABox assertions such that the knowledge base $\boldsymbol{K''}$=<$\boldsymbol{T}$, $\boldsymbol{A}\cup\{\neg Q\}\cup E$> is inconsistent, and the set E satisfies all the three conditions listed in Def. 2.

For the ease of presentation, we firstly transform the TBox into a set $\boldsymbol{T'}$ of concepts according to the following three steps:

(1) transform every concept definition $C\equiv D$ into two GCIs $C\sqsubseteq D$ and $D\sqsubseteq C$;

(2) transform every GCI $C\sqsubseteq D$ into $\top\sqsubseteq\neg C\sqcup D$ if C is not the Top concept $\top$;

(3) abbreviate the GCI $\top\sqsubseteq\neg C\sqcup D$ as $\neg C\sqcup D$.

For every concept occurring in the knowledge base $\boldsymbol{K}'=<\boldsymbol{T}', \boldsymbol{A}\cup\{\neg Q\}>$, transform it into a normal form according to the following steps:

(1) transform it into an NNF (i.e., negation normal form) such that negation signs occur only in front of concept names;

(2) transform the NNF further into a conjunction normal form by the function $\tau_{\sqcap}$:

① $\tau_{\sqcap}(\sqcup_{1\leq i\leq n}C_i) = \sqcup_{1\leq i\leq n}C_i$, where $C_i\in\{C,\exists r.C,\forall r.C\}$;

② $\tau_{\sqcap}(B\sqcup(C\sqcap D)\sqcup E) = \tau_{\sqcap}(B\sqcup C\sqcup E) \sqcap \tau_{\sqcap}(B\sqcup D\sqcup E)$;

③ $\tau_{\sqcap}(\exists r.C) = \exists r.\tau_{\sqcap}(C)$, and $\tau_{\sqcap}(\forall r.C) = \forall r.\tau_{\sqcap}(C)$;

④ $\tau_{\sqcap}(\exists r^{-}.C) = \exists r^{-}.\tau_{\sqcap}(C)$, and $\tau_{\sqcap}(\forall r^{-}.C) = \forall r^{-}.\tau_{\sqcap}(C)$.

(3) eliminate redundancy by the function τ_{eli}:

① $\tau_{eli}(\forall r.\top)=\top$, $\tau_{eli}(C\sqcap\top)=C$, $\tau_{eli}(C\sqcap\neg C)=\bot$;

② $\tau_{eli}(\exists r.\bot)=\bot$, $\tau_{eli}(C\sqcap\bot)=\bot$, $\tau_{eli}(C\sqcap C)=C$;

③ $\tau_{eli}(\forall r.C\sqcap\forall r.D)=\forall r.(C\sqcap D)$, $\tau_{eli}(\exists r.C\sqcap\exists r.(C\sqcap D))=\exists r.(C\sqcap D)$;

④ $\tau_{eli}(\exists r.C\sqcap\forall r.\bot)=\bot$, $\tau_{eli}(\exists r.\top\sqcap\exists r.C)=\exists r.C$.

(4) for each resulted conjunction normal form $(B_1\sqcup B_2\sqcup\cdots\sqcup B_n)\sqcap(C_1\sqcup C_2\sqcup\cdots\sqcup C_n)\sqcap\cdots(D_1\sqcup D_2\sqcup\cdots\sqcup D_n)$ which is contained in $\boldsymbol{T}'$, split it into the disjunction terms of $B_1\sqcup B_2\sqcup\cdots\sqcup B_n$, $C_1\sqcup C_2\sqcup\cdots\sqcup C_n$, $\cdots$, and $D_1\sqcup D_2\sqcup\cdots\sqcup D_n$.

Let $\boldsymbol{K}''=<\boldsymbol{T}'', \boldsymbol{A}''\cup\{\neg Q''\}>$ be the resulted knowledge base. Let $J=\boldsymbol{T}''\cup\boldsymbol{A}''\cup\{\neg Q''\}$, and let $\boldsymbol{N_C}^{\mathbf{K}}$ be all the individuals occurring in $\boldsymbol{K}''$.

Now, we will construct a Tableau for J. Starting from the initial node $\{\neg Q''\}$, the Tableau is constructed by using rules introduced in the following paragraphs. During the construction process, we will use four set $\text{Role}_{\exists}$, $\text{Role}_{\forall}$, $\text{Role}_{\exists^-}$ and $\text{Role}_{\forall^-}$ which are initialized as empty sets to record the relationship between individuals.

All the rules used for constructing the Tableau are divided into two groups: node evolution rules, and branch expansion rules.

There are six node evolution rules.

(1)$\sqcap$-rule: for any assertion of the form $(C_1\sqcap C_2\cdots\sqcap C_n)(x)$ in the current leaf, generate n Tableau and the assertion in current leaf changes to $C_1(x)$, $C_2(x)$,$\cdots$, $C_n(x)$ respectively;

(2)$\sqcup$-rule: for any assertion of the form $(C_1\sqcup C_2\cdots\sqcup C_n)(x)$ in the current leaf, attach n new successor node $\{C_1(x)\}$, $\{C_2(x)\}$,$\cdots$, $\{C_n(x)\}$;

(3)$\exists$-rule: for any assertion of the form $\exists r.C(x)$ in the current leaf, generate a new individual b, put $r(x, b)$ in the set $\text{Role}_{\exists}$, and replace $\exists r.C(x)$ with $C(b)$;

(4)$\exists^-$-rule: for any assertion of the form $\exists r^-.C(x)$ in the current leaf, generate a new individual c, put $r(c, x)$ in the set $\text{Role}_{\exists^-}$, and replace $\exists r^-.C(x)$ with $C(c)$;

(5)$\forall$-rule: for any assertion of the form $\forall r.D(x)$ in the current leaf, generate a new individual argument y, put $r(x, y)$ in the set $\text{Role}_{\forall}$, and replace $\forall r.D(x)$ with $D(y)$;

(6)$\forall^-$-rule: for any assertion of the form $\forall r^-.D(x)$ in the current leaf, generate a new individual argument z, put $r(z, x)$ in the set $\mathrm{Role}_{\forall}^-$, and replace $\forall r^-.D(x)$ with $D(z)$.

Expanding a branch needs to select a concept from J according to the assertion in the current leaf node, e.g. $C(z)$ is the assertion in the current leaf node. In order to investigate whether z has a relationship in $\mathrm{Role}_{\exists}$、$\mathrm{Role}_{\forall}$、$\mathrm{Role}_{\exists}^-$ or $\mathrm{Role}_{\forall}^-$, we use a transition function ***Translate*** to map $C(z)$ into the initial value of $C(z)$. The recursive definition of transition function ***Translate*** is as follows:

If the function is ***Translate***($C(z)$), then Retrieving $r(y, z)$ in $\mathrm{Role}_{\exists}$ and $\mathrm{Role}_{\forall}$ and Retrieving $r(z, y)$ in $\mathrm{Role}_{\exists}^-$ and $\mathrm{Role}_{\forall}^-$:

① $r(y, z)$ is in $\mathrm{Role}_{\exists}$, execute ***Translate***($\exists r.C(y)$);

② $r(z, y)$ is in $\mathrm{Role}_{\exists}^-$, execute ***Translate***($\exists r^-.C(y)$);

③ $r(y, z)$ is in $\mathrm{Role}_{\forall}$, execute ***Translate***($\forall r.C(y)$);

④ $r(z, y)$ is in $\mathrm{Role}_{\forall}^-$, execute ***Translate***($\forall r^-.C(y)$);

⑤ or else ***Translate***($C(z)$)=$C(z)$.

For example, suppose $\mathrm{Role}_{\exists}=\{r_3(y, z)\}$, $\mathrm{Role}_{\forall}=\{r_1(a, x)\}$, $\mathrm{Role}_{\exists}^-=\{r_2(y, x)\}$ and $a\in \boldsymbol{N}_{\mathrm{C}}^{\mathbf{K}}$. By the procedure ***Translate***($C(z)$) we get the assertion $\exists r_3.C(y)$ since $r_3(y, z) \in\mathrm{Role}_{\exists}$, and then will execute ***Translate***($\exists r_3.C(y)$); since $r_2(y, x)\in\mathrm{Role}_{\exists}^-$, we will get the assertion $\forall r_2^-.\exists r_3.C(x)$ and then will execute ***Translate***($\exists r_2^-.\exists r_3.C(x)$); since $r_1(a,x)\in\mathrm{Role}_{\forall}$, then we will get the assertion $\forall r_1.\exists r_2^-.\exists r_3.C(a)$ and therefore will execute ***Translate***($\forall r_1.\exists r_2^-.\exists r_3.C(a)$); finally we have ***Translate***($\forall r_1.\exists r_2^-.\exists r_3.C(a)$) $=\forall r_1.\exists r_2^-.\exists r_3.C(a)$ and the procedure terminates.

Expanding a branch needs to select a concept from J according to the assertion in the current leaf node. The concept selected should satisfy the following conditions:

(1)Condition 1: when the assertion in the current leaf is $C(x)$, if $D(y)$=***Trans late***($C(x)$), $\neg D$ must be a part of the concept selected from J;

(2)Condition 2: when expanding a branch, if $E(x)$ is any assertion in the node of the current branch, and $F(y)$=***Translate***($E(x)$), then F should not be a part of the concept selected from J.

Let $E(x)$ be the assertion in the current leaf, and $C_1\sqcup C_2\cdots\sqcup C_n$ be the concept selected from J. In that case, attach n new successor node $\{C_1(x)\}$, $\{C_2(x)\}$,$\cdots$,$\{C_n(x)\}$ to $E(x)$ when we expand the current branch.

If on a branch in a Tableau appears both $C(x)$=***Translate***($D(y)$) and $\neg C(x)$ =***Translate***($E(z)$), then the branch is called closed. A Tableau is called closed if every branch of it is closed.

A Tableau T is called saturated if it no rule can be applied to it anymore.

Finally, given an abduction problem <$\boldsymbol{K}$, Q> of ALCI, the complete procedure of the algorithm is as follows:

(1) Construct a knowledge base $\boldsymbol{K}'$=<$\boldsymbol{T}$, $\boldsymbol{A}\cup\{\neg Q\}$> and transform it into the knowledge base $\boldsymbol{K}''$=<$\boldsymbol{T}''$, $\boldsymbol{A}''\cup\{\neg Q''\}$>, get the set $J=\boldsymbol{T}''\cup\boldsymbol{A}''\cup\{\neg Q''\}$;

(2) Construct the Tableau of J, get the assertions that can make the unclosed branches close for every unclosed Tableau modal, and put these assertions in set E; for example, if $C(x_n)$ is the last node of an unclosed branch in the Tableau, then put $\neg$***Translate***$(C(x_n))$ into E;

(3) Check every element of E according to Def. 2, remove the elements that don't satisfy the requirement of consistency, relevance or minimality from the set E;

(4) The result set E is a solution for <$\boldsymbol{K}$, Q>.

At the end of this section, we illustrate the procedure of the algorithm with a small example. Consider an ALCI abduction problem <$\boldsymbol{K}$, Q>, where Q=*HAPPY*(*John*) and $\boldsymbol{K}$ is shown in Table 1.

Table 1. The knowledge base $\boldsymbol{K}$

TBox	ABox
$NIHILIST\sqcap\forall loves^{-}.\exists owns.(DOG\sqcap BOOK))\sqsubseteq HAPPY$	*NIHILIST*(*John*)
$OPTIMIST\sqsubseteq\forall loves^{-}.\exists owns.DOG$	*DOG*(*Snoopy*)

After the pretreatment for $\boldsymbol{T}\cup\boldsymbol{A}\cup\neg Q$, we will get the set $J=\{\neg NIHILIST\sqcup\exists loves^{-}.\forall owns.(\neg DOG\sqcup\neg BOOK)\sqcup HAPPY$, $\neg OPTIMIST\sqcup\forall loves^{-}.\exists owns.DOG$, $NIHILIST(John)$, $DOG(Snoopy)$, $\neg HAPPY(John)\}$. The procedure for constructing the Tableau is illustrate in Fig. 1. According to Fig. 1 and the transition function ***Translate***, we will get $E=\{NIHILIST(john)$, $OPTIMIST(john)$, $\exists owns.BOOK(john)\}$. According to Definition 2, we get a solution $\{NIHILIST(john)$, $OPTIMIST(john)$, $\forall loves^{-}.\exists owns.BOOK(john)\}$.

4 Conclusion

In this paper, the ABox abduction problem is transformed into the consistency problem of knowledge base, and the traditional Tableau construction process of the description logic ALCI is extended and altered to deal with this problem. As a result, the ABox abduction problem can be solved directly in description logics and do not need to use arguments and Skolem terms of first order logics. One of our future work is to improve the algorithm to deal with role assertions. Another work is to expend the algorithm for more expressive description logics.

Acknowledgements

This work is supported by the National Natural Science Foundation of China (Nos. 60903079, 60963010), the Natural Science Foundation of Guangxi Province (No.2012GXNSFBA053169), and the Innovation Project of Guangxi Graduate Education (No. 2011105950812M21).

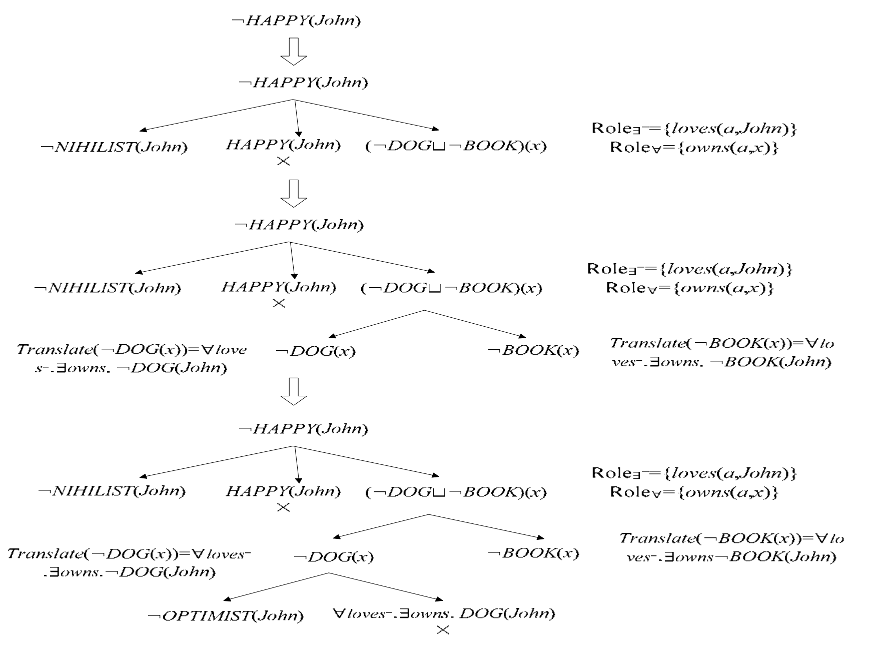

Fig. 1. the procedure of constructing Tableau

References

1. Marquis, P.: Extending abduction from propositional to first-order logic. In: Proceedings of the First International Workshop on Fundamentals of Artificial Intelligence Research. Berlin: Springer-Verlag, 1991: 141-155
2. Mayer, M.C., Pirri, F.: First order abduction via tableau and sequent calculi. *Logic Journal of the IGPL*, 1993, 1(1): 99-117
3. Klarman, S., Eendriss, Schlobach, et al.: ABox abduction in the description logic ALC. *Journal of Automated Reasoning*, 2011, 46(1): 43-80
4. Elsenvroich, C., Kutz, O., Sattler, U., et al.: A case for abductive reasoning over ontologies. In: Proceedings of the 9th OWL: Experiences and Directions. Amsterdam: IOS Press, 2006: 56-75
5. Davenport, D., Hill, R.: Fast abductive reasoning over ontologies. In: Proceedings of the American Association for Artificial Intelligence 2006 Fall Symposium. Menlo Park: AAAI Press, 2006: 84-97
6. Jianfeng Du, Guilin Qi, Yidong Shen, et al.: Towards Practical ABox Abduction in Large OWL DL Ontologies. In: Proc. of the 25th American Association for Artificial Intelligence Conference. Menlo Park: AAAI Press, 2011: 1160-1165
7. Baader, F., Nutt, W., Horrocks, L., et al.: Handbook of Description Logic, Cambridge: Cambridge University Press, 2007
8. Baader, F., Sattler, U.: An Overview of Tableau Algorithms for Description Logics. *Studia Logica*, 2001, 69: 5-40
9. Hahnle, R.: Tableaux and Related Methods. Handbook of Automated Reasoning, Amsterdam: Elsevier, 2001