·开发研究与工程应用· 文章编号：1000—3428(2013)12—0308—08 文献标识码：A 中图分类号：TP301.6

描述逻辑 SHIQ 的 ABox 一致性判定算法

彭 立，杨恒伏

(湖南第一师范学院信息科学与工程系，长沙 410205)

摘 要：为判定描述逻辑 SHIQ 的 ABox 一致性，提出一种 Tableau 算法。给定 TBox *T*、ABox *A* 和角色层次 *H*，通过预处理将 *A* 转换成标准的 ABox *A*’，按照特定的完整策略将一套 Tableau 规则应用于 *A*’，从而不断地对 *A*’进行扩展，直到将其扩展成完整的 ABox *A*”为止。*A*、*T* 和 *H* 一致，当且仅当算法能产生一个完整且无冲突的 ABox *A*”。该算法采用的阻塞机制能防止 Tableau 规则被无限次执行，避免多余的规则应用。通过证明 Tableau 规则的执行次数为有限次，确认算法的可终止性。通过证明由 *A*”能构造一个同时满足 *A*、*T* 和 *H* 的解释，确认算法的合理性。通过证明 Tableau 规则的执行不会破坏 *A*’与 *H* 的一致关系，确认算法的完备性。

关键词：描述逻辑 SHIQ；ABox 一致性判定；Tableau 算法；阻塞机制；可终止性；合理性；完备性

ABox Consistency Decision Algorithm for Description Logic SHIQ

PENG Li, YANG Heng-fu

(Department of Information Science and Engineering, Hunan First Normal University, Changsha 410205, China)

【Abstract】 In order to decide ABox consistency for Description Logic(DL) SHIQ, a Tableau algorithm is presented. Given a TBox *T*, an ABox *A* and a role hierarchy *H*, the algorithm first converts *A* into a standard ABox *A*’ by pre-disposal, and then applies a set of Tableau rules to *A*’ according to specific completion strategies, thus *A*’ is extended continually, until it is extended to a complete ABox *A*”. *A* is consistent with *T* and *H*, if and only if the algorithm can yield a complete and clash-free ABox *A*”. The blocking mechanism adopted by the algorithm can prevent Tableau rules’ unlimited execution, and avoid redundant rule application. By proving Tableau rules’ execution times is limited, the algorithm’s termination is ensured. By proving Tableau rules’ excecution is unlikely to destroy the consistency between *A*’ and *H*, the algorithm’s soundness is ensured. By proving an explanation which satisfies *A*, *T* and *H* can be constructed in terms of *A*”, the algorithm’s completeness is ensured.

【Key words】 Description Logic(DL) SHIQ; ABox consistency decision; Tableau algorithm; blocking mechanism; termination; soundness; completeness

DOI: 10.3969/j.issn.1000-3428.2013.12.066

1 概述

描述逻辑(Description Logics, DL)是一类用于知识表示和推理的逻辑，它广泛地应用于信息系统、数据库、软件工程和语义 Web 等领域[1-2]。经典的 DL 语言 ALC[3]可满足一般知识建模的需要，但在领域知识较为复杂的情况下，仍暴露出表达力不足的缺陷。为 ALC 增加传递角色[4]、角色层次[5]、反向角色[6]和限定性数量约束这些新特征后，便形成了一种表达力较强的 DL 语言 ALCHIQ$_R$+，简写为 SHIQ[7]，它能解决较为复杂的知识建模问题。

为了能让某种 DL 在实际应用中充分发挥作用，一方面需要用一个知识库来保存经该 DL 建模的应用域知识；另一方面需要提供与知识库相关的推理服务，而 ABox 一致性判定是最基本的推理服务。就 SHIQ 而言，其知识库分为 TBox、ABox 和角色层次 3 个部分，ABox 一致性判定即判断 ABox 是否与 TBox 和角色层次一致。纵观国内外的 DL 研究可以发现，要解决 ABox 一致性判定问题，主要可采用 2 种方法：一种是预完整方法[8-9]，其思想是将 ABox 转换成预完整的 ABox，从而将 ABox 一致性判定归约为概念可满足性判定。该方法较难设计，目前还没有适用于 SHI[6] 的预完整方法，以及 SHIQ 的方法。另一种是设计专门的 Tableau 算法，通过 Tableau 算法判定 ABox 的一致性。这也是目前解决 ABox 一致性判定问题普遍采用的方法。就某种 DL 而言，其 ABox 一致性判定算法通常可通过对相应

基金项目：国家自然科学基金资助项目(61073191)；湖南省教育厅科学研究基金资助项目(12C0593)；湖南第一师范学院校级课题基金资助项目(XYS10N09)

作者简介：彭 立(1974－)，男，讲师、硕士，主研方向：描述逻辑，语义 Web；杨恒伏，副教授、博士

收稿日期：2012-10-25 **修回日期：**2013-01-10 **E-mail：**goodbetter@163.com

的概念满足性判定算法进行改造而得来。

本文首先对 SHIQ 的语法和语义进行定义，然后给出一种 SHIQ 的 ABox 一致性判定 Tableau 算法，并证明该算法的合理性、可终止性和完备性。

2 描述逻辑 SHIQ

本节对 SHIQ 的语法和语义进行定义。事先规定：N_C 为概念名集合，N_R 为角色名集合，N_T 为传递角色名集合，N_P 为非传递角色名集合，O 为个体集合，$N_C\cap N_R=\varnothing$，$N_C\cap O=\varnothing$，$N_R\cap O=\varnothing$，$N_R=N_T\cup N_p$，$N_T\cap N_P=\varnothing$。

定义 1 角色集$:=N_R\cup\{R^-|R\in N_R\}$($R^-$为 R 的反向角色)。形式为 $R\sqsubseteq S$ 的公理是角色包含公理，其中 R 和 S 都是角色。角色层次是角色包含公理的有限集合。

为方便定义，特引入 *inv* 函数和$\sqsubseteq^*$标记。*inv* 函数被定义为：若 $R\in N_R$，则 $inv(R):=R^-$；若 $R=S^-$ 且 $S\in N_R$，则 $inv(R):=S$。对于角色层次 H，$\sqsubseteq^*$是 $H\cup\{inv(R)\sqsubseteq inv(S)|R\sqsubseteq S\in H\}$上$\sqsubseteq$的自反传递闭包。

定义 2 角色 R 为传递角色，当且仅当 $R\in N_T$ 或 $inv(R)\in N_T$。角色 R 为简单角色，当且仅当对于所有的 $S\sqsubseteq^* R$，S 不是传递角色。

定义 3 概念集为满足以下条件的最小集合：(1)*Nc* 中的每一个元素、$\top$和$\bot$都是概念；(2)若 C 和 D 是概念，R 是角色，则 $C\sqcap D$、$C\sqcup D$、$\neg C$、$\exists R.C$ 和$\forall R.C$ 都是概念；(3)若 n 为正整数，C 为概念，S 为简单角色，则$\leqslant nS.C$ 和$\geqslant nS.C$ 也是概念。

定义 4 形式为 $C\sqsubseteq D$ 的公理是一般概念包含公理(General Concept Inclusion, GCI)，其中，C 和 D 都是概念；TBox 是 GCI 的有限集合。

定义 5 形式为 $a:C$ 的公理是概念断言公理，形式为 $a\ R\ b$ 的公理是角色断言公理，形式为 $a\neq b$ 的公理是不等公理，其中，C 为概念；R 为角色；$a,b\in O$。ABox 是这 3 种断言公理的有限集合。

下面给出 SHIQ 的模型论语义。

定义 6 一个解释 $I=(\Delta^I,\cdot^I)$包含了一个非空集合Δ^I(解释域)和一个解释函数$\cdot^I$。解释函数将每一个概念映射为Δ^I的一个子集，将每一个角色映射为$\Delta^I\times\Delta^I$的一个子集，将每一个个体映射为Δ^I中的一个元素。让 C、D 为概念，R 为角色，n 为正整数，$P\in N_R$, $Q\in N_T$，#表示集合的势(下同)，解释函数应满足以下约束：

$\top^I=\Delta^I$ $\bot^I=\varnothing$ $(\neg C)^I=\Delta^I\backslash C^I$

$(C\sqcap D)^I=C^I\cap D^I$ $(C\sqcup D)^I=C^I\cup D^I$

$(\exists R.C)^I=\{x\in\Delta^I|$存在 $y\in\Delta^I$ 使得$(x,y)\in R^I$ 且 $y\in C^I\}$

$(\forall R.C)^I=\{x\in\Delta^I|$对于任意 $y\in\Delta^I$，若$(x,y)\in R^I$，则 $y\in C^I\}$

$(\leqslant nR.C)^I=\{x\in\Delta^I\,|\,\#\{y|(x,y)\in R^I$ 且 $y\in C^I\}\leqslant n\}$

$(\geqslant nR.C)^I=\{x\in\Delta^I\,|\,\#\{y|(x,y)\in R^I$ 且 $y\in C^I\}\geqslant n\}$

$(x,y)\in P^I$ 当且仅当$(y,x)\in(P^-)^I$

若$(x,y)\in Q^I$ 且$(y,z)\in Q^I$，则$(x,z)\in Q^I$

让 T 为 TBox，A 为 ABox，H 为角色层次，I 为解释。I 满足 H，当且仅当 I 满足所有的 $R\sqsubseteq S\in H$(使 $R^I\subseteq S^I$)；这样的 I 被称为 H 的模型。I 满足 T，当且仅当 I 满足所有的 $C\sqsubseteq D\in T$(使 $C^I\subseteq D^I$)；这样的 I 被称作 T 的模型。I 满足 A，当且仅当：(1)I 满足所有的 $a:C\in A$(使 $a^I\in C^I$)；(2)I 满足所有的 $a\ R\ b\in A$(使$(a^I,b^I)\in R^I$)；(3)I 满足所有的$a\neq b\in A$(使 $a^I\neq b^I$)，这样的 I 被称作 A 的模型。A 与 H 一致，当且仅当 A 和 H 有共同的模型。A、T 和 H 一致，当且仅当 A、T 和 H 有共同的模型。

3 SHIQ 的 ABox 一致性判定算法

对于 SHIQ，ABox 一致性判定即判断 ABox 是否与 TBox 和角色层次一致。在文献[7]提出的 SHIQ 概念可满足性判定算法的基础上，本文提出了一种适用于 SHIQ 的 ABox 一致性判定 Tableau 算法。给定 TBox T、ABox A 和角色层次 H，该算法先通过预处理将 A 转换成标准的 ABox A'，然后按照特定的完整策略将一套 Tableau 规则应用于 A'，从而不断地对 A'进行扩展，直到将它扩展成完整的 ABox A"为止。若算法能产生一个完整且无冲突的 ABox A"，则 A、T 和 H 一致，否则不一致。

3.1 预处理

定义 7 形式为$\forall x.x:C$ 的公理是通用概念公理，其中，C 为概念。解释 I 满足$\forall x.x:C$，当且仅当 $C^I=\Delta^I$[10]。

定义 8 对于一个概念，若¬运算符只出现在它所包含的概念名的前面，则该概念具有否定标准形式。所有的概念都可以通过以下规则转换成否定标准形式：

$\neg\top=\bot$ $\neg\bot=\top$ $\neg(\neg C)=C$

$\neg(C\sqcap D)=\neg C\sqcup\neg D$ $\neg(C\sqcup D)=\neg C\sqcap\neg D$

$\neg(\exists R.C)=\forall R.\neg C$ $\neg(\forall R.C)=\exists R.\neg C$

$\neg(\leqslant nR.C)=\geqslant(n+1)R.C$

$$\neg(\leqslant nR.C)=\begin{cases}\leqslant(n-1)R.C & \text{若}n>1\\ \forall R.\neg C & \text{若}n=1\end{cases}$$

给定 TBox T 和初始的 ABox A，预处理按以下步骤将 A 转换成标准的 ABox A'，A'被称为 A 的标准形式。

为每个 $C\sqsubseteq D\in T$，在 A 中添加相应的通用概念公理$\forall x.x:(\neg C\sqcup D)$。

将 A 中出现的每一个概念转换成否定标准形式。

下文中出现的 ABox 指的都是已经过了预处理的 ABox，除非它被指明是初始的 ABox。

3.2 阻塞机制

预处理结束后，算法通过执行 Tableau 规则对 ABox 进行扩展。为了防止 Tableau 规则被无限次执行，算法采用了一种阻塞机制，该机制是对 pair-wise blocking[11]的改进，它类似于 anywhere blocking[12-13]。

定义 9 对于 ABox A 中存在的个体 a 和 b，a 的概念集

$L(a)=\{C|a:C\in A\}$，a 和 b 的角色集 $L(a,b)=\{R|a\ R\ b\in A\}$。

定义 10 对于 ABox A 中存在的个体 a 和 b，a 为 b 的前驱，当且仅当存在角色 R 使得 $a\ R\ b\in A$；a 为 b 的后继，当且仅当存在角色 R 使得 $b\ R\ a\in A$；a 为 b 的邻居，当且仅当 a 为 b 的前驱或后继；a 为 b 的 R 前驱，当且仅当 $a\ inv(S)\ b\in A$ 且 $S\sqsubseteq^*R$；a 为 b 的 R 后继，当且仅当 $b\ S\ a\in A$ 且 $S\sqsubseteq^*R$；a 为 b 的 R 邻居，当且仅当 a 为 b 的 R 前驱或 R 后继。祖先关系为前驱关系的传递闭包，后代关系为后继关系的传递闭包。

定义 11 ABox 中存在的个体分为老个体和新个体 2 类。老个体是在算法执行之前就已经存在的个体。新个体是算法创建的个体。

定义 12 规定用$\prec$表示 ABox 中新个体的创建顺序。若算法在 ABox 中创建了一个新个体 b，则对于 ABox 中所有已经存在的新个体 a，有 $a\prec b$。

定义 13(阻塞) 对于 ABox 中存在的个体 a 和 b，a 被 b 直接阻塞，当且仅当：(1)a 和 b 都是新个体，$b\prec a$；(2)a 没有祖先被阻塞，b 未被阻塞；(3)a 有一个前驱 a'，b 有一个前驱 b'，使得 $L(a)=L(b)$，$L(a')=L(b')$，$L(a',a)=L(b',b)$。a 被间接阻塞，当且仅当 a 是新个体而且 a 的前驱被阻塞。一个个体被阻塞，当且仅当它被直接阻塞或被间接阻塞。

与 pair-wise blocking 不同的是，该阻塞机制允许一个新个体被在其之前创建的任意新个体直接阻塞，而不仅仅局限于其祖先，这无疑可以提高新个体被阻塞的概率，从而减少 Tableau 规则的应用次数并相应地提高算法的效率。假设 ABox $A=\{a:\geqslant 2R.C,\ a:\forall R.(\exists R.C),\ a\ R\ b_1,\ a\ R\ b_2,\ b_1:\exists R.C,\ b_2:\exists R.C\}$，其中，$a$ 为老个体；b_1 和 b_2 为新个体；而且 $b_1\prec b_2$。若采用 pair-wise blocking，显然 b_2 不可能被阻塞，而且$\exists$规则(见定义 14)可应用于 b_2，但是，若采用上面的阻塞机制，则 b_2 被 b_1 直接阻塞，而且$\exists$规则不能再应用于 b_2，这说明 Tableau 规则的应用次数减少了，算法的效率也可相应地提高。

3.3 Tableau 规则

定义 14(Tableau 规则) 让 A 为 ABox，算法采用了 9 种 Tableau 规则对 A 进行扩展。

(1)$\sqcap$规则

条件：1)$a:C\sqcap D\in A$，a 未被间接阻塞；2)$\{a:C,\ a:D\}\not\subseteq A$

动作：$A\rightarrow A\cup\{a:C,\ a:D\}$

(2)$\sqcup$规则

条件：1)$a:C\sqcup D\in A$，a 未被间接阻塞；2)$\{a:C,\ a:D\}\cap A=\varnothing$

动作：$A\rightarrow A\cup\{a:C\}$或$A\rightarrow A\cup\{a:D\}$

(3)$\exists$规则

条件：1)$a:\exists R.C\in A$，a 未被阻塞；2)a 没有 R 邻居 b 使得 $b:C\in A$

动作：$A\rightarrow A\cup\{a\ R\ b,\ b:C\}$($b$ 为 A 中不存在的个体)

(4)$\forall$规则

条件：1)$a:\forall R.C\in A$，a 未被间接阻塞；2)a 有一个 R 邻居 b 使得 $b:C\notin A$

动作：$A\rightarrow A\cup\{b:C\}$

(5)$\forall_T$规则

条件：1)$a:\forall R.C\in A$，a 未被间接阻塞；2)存在一个传递角色 S 使得 $S\sqsubseteq^*R$；3)a 有一个 S 邻居 b 使得 $b:\forall S.C\notin A$

动作：$A\rightarrow A\cup\{b:\forall S.C\}$

(6)?规则

条件：1)$a:\leqslant nR.C\in A$，a 未被间接阻塞；2)a 有一个 R 邻居 b 使得 $b:C\notin A$ 且 $b:\sim C\notin A$

动作：$A\rightarrow A\cup\{b:C\}$或 $A\rightarrow A\cup\{b:\sim C\}$($\sim C$ 为$\neg C$ 的否定标准形式，下同

(7)$\geqslant$规则

条件：1)$a:\geqslant nR.C\in A$，a 未被阻塞；2)$\#N(a,R,C)<n$

动作：$A\rightarrow A\cup\{a\ R\ b_k,\ b_k:C|1\leqslant k\leqslant n\}\cup\{b_i\neq b_j|1\leqslant i,j\leqslant n$ 且 $i\neq j\}$($b_1,b_2,\cdots,b_n$ 为 A 中不存在的个体；$N(a,R,C):=\{b|b$ 为 a 的 R 邻居且 $b:C\in A\}$，下同

(8)$\leqslant$规则

条件：1)$a:\leqslant nR.C\in A$，a 未被间接阻塞；2)$\#N(a,R,C)>n$；3)a 有 2 个 R 邻居 b_1 和 b_2 使得 $b_1:C\in A$，$b_2:C\in A$，$b_1\neq b_2\notin A$

动作：

1)若 b_1 和 b_2 都是老个体，则 $A\rightarrow A[b_1/b_2]\cup\{b_1\doteq b_2\}$；

2)若 b_1 是老个体，b_2 是新个体，b_1 为 a 的前驱，则 $A\rightarrow A[trim(b_2)][reverse(a,b_2)][b_2/b_1]$；

3)若 b_1 是老个体，b_2 是新个体，b_1 为 a 的后续，则 $A\rightarrow A[trim(b_2)][b_2/b_1]$；

4)若 b_2 是老个体，b_1 是新个体，b_2 是 a 的前驱，则 $A\rightarrow A[trim(b_1)][reverse(a,b_1)][b_1/b_2]$；

5)若 b_2 是老个体，b_1 是新个体，b_2 是 a 的后继，则 $A\rightarrow A[trim(b_1)][b_1/b_2]$；

6)若 b_1 和 b_2 都是新个体，b_1 为 a 的前驱，则 $A\rightarrow A[trim(b_2)][reverse(a,b_2)][b_2/b_1]$；

7)若 b_1 和 b_2 都是新个体，b_2 为 a 的前驱，则 $A\rightarrow A[trim(b_1)][reverse(a,b_1)][b_1/b_2]$；

8)若 b_1 和 b_2 都是新个体，且都为 a 的后继，则 $A\rightarrow A[trim(b_1)][b_1/b_2]$。

$A[][]\cdots$表示在 A 中依次执行若干个操作，每个[ ]代表一个操作。$[trim(b_1)]$表示删除所有包含了 b_1 的后代的公理；$[reverse(a,b_1)]$表示将每一条包含了 a 和 b_1 的公理 $a\ S\ b_1$ 转换成 $b_1\ inv(S)\ a$；$[b_1/b_2]$表示用 b_2 替换 b_1。

(9)$\forall_x$规则

条件：1)$\forall x.x:C\in A$；2) A 中存在一个未被间接阻塞的个体 a 使得 $a:C\notin A$；3) A 中不存在另一个体 b 使得 $a\doteq b\in A$

动作：$A\rightarrow A\cup\{a:C\}$

下面对 Tableau 规则进行一些补充说明。

⊓、⊔、∀、$\forall_T$、$\forall_x$和?规则能对 A 中个体的概念集进行扩展。若⊓规则被应用于 $a{:}C_1 \sqcap C_2 \in A$，则 $a{:}C_1$ 和 $a{:}C_2$ 被添加到 A，$L(a)$被扩展。若⊔规则被应用于 $a{:}C_1 \sqcup C_2 \in A$，则 $a{:}C_1$ 或 $a{:}C_2$ 被添加到 A，$L(a)$被扩展。若∀规则被应用于 $a{:}\forall R.C \in A$ 使得 $b{:}C$ 被添加到 A，则 $L(b)$被扩展。若$\forall_T$规则被应用于 $a{:}\forall R.C \in A$ 使得 $b{:}\forall S.C$ 被添加到 A，则 $L(b)$被扩展。若$\forall_x$规则被应用于$\forall x.x{:}C \in A$ 使得 $a{:}C$ 被添加到 A，则 $L(a)$被扩展。若?规则被应用于 $a{:}\leqslant nR.C \in A$ 使得 $b{:}C$ 或 $b{:}\sim C$ 被添加到 A，则 $L(b)$被扩展。

∃和⩾规则能在 A 中创建新个体。若∃规则被应用于 $a{:}\exists R.C \in A$，则该规则会为 a 创建一个新的 R 后继 b，并将 $a\ R\ b$ 和 $b{:}C$ 添加到 A。若⩾规则被应用于 $a{:}\geqslant nR.C \in A$，则该规则会为 a 创建 n 个 R 后继 $b_1,b_2,\cdots,b_n$，并将$\{a\ R\ b_k,\ b_k{:}C \mid 1 \leqslant k \leqslant n\} \cup \{b_i \neq b_j \mid 1 \leqslant i,j \leqslant n$ 且 $i \neq j\}$添加到 A；$b_i \neq b_j$可以防止 b_i 和 b_j 被⩽规则合并。

⩽规则能对 A 中的个体进行合并。若⩽规则被应用于 $a{:}\leqslant nR.C \in A$，则该规则会将 a 的 2 个 R 邻居 b_1 和 b_2 合并在一起，也就是根据 b_1 和 b_2 之间的关系执行相应的动作，从而让 b_1 取代 b_2 或者 b_2 取代 b_1。通过观察该规则可以发现，无论该规则执行时采用哪种动作，都不会导致这 2 种情况的出现：(1)新个体成为老个体的祖先；(2)新个体的前驱超过一个。之所以让该规则在 A 中添加 $b_1 \doteq b_2$ 形式的公理，主要是为了证明定理 1。

通过观察 Tableau 规则的第(1)个条件可以发现，当 A 中的任意个体 a 被阻塞时，∃和⩾规则不能再应用于 a，即不能再为 a 创建后继，这样就确保了 A 中的个体不会无限制地增加，算法因而也能终止；当 a 被间接阻塞时，⊓、⊔、∀、$\forall_T$、$\forall_x$、?和⩽规则不能再应用于 a，从而避免了不必要的规则应用，算法也得以优化。在这些规则中，⊓、∃、∀、$\forall_T$、$\forall_x$和⩾规则是确定规则，⊔、?和⩽规则是非确定规则，前者产生的结果是确定的而后者能以不确定的方式执行，从而产生不确定的结果。因为非确定规则的存在，算法所产生的完整的 ABox 也是不确定的，也就是说该算法是非确定性算法。

3.4 完整策略

预处理结束后，算法按照完整策略不断地将 Tableau 规则应用于 ABox，直到将它扩展成完整的 ABox 为止，这时算法停止执行。

定义 15 ABox A 是完整的 ABox，当且仅当没有 Tableau 规则可应用于 A，或者 A 中出现了冲突。

定义 16 ABox A 中出现了冲突，是指：(1)$a{:}\bot \in A$，或(2)$\{a{:}C,\ a{:}\neg C\} \in A (C \in N_C)$，或(3)$a{:}\leqslant nR.C \in A$，而且 a 有 n+1 个 R 邻居 $b_1,b_2,\cdots,b_{n+1}$，使得对于 $1 \leqslant i < j \leqslant n$+1，有 $b_i{:}C \in A$ 且 $b_i \neq b_j \in A$。

算法按照完整策略控制 Tableau 规则的执行。

策略 1 算法按以下步骤处理 ABox 中的个体。

(1)按任意顺序将所有的老个体处理一遍。

(2)按≺顺序将所有的新个体处理一遍。

(3)重复步骤(1)、步骤(2)，当没有 Tableau 规则可应用时，算法停止执行。

策略 2 在对某个个体 a 进行处理时，算法按以下步骤将 Tableau 规则应用于 a。

(1)反复应用⊓、⊔、和$\forall_x$ 规则，直到它们无法应用或 ABox 中出现冲突为止。若 ABox 中出现了冲突，则算法停止执行。

(2)反复应用∃和⩾规则，直到它们无法应用为止。

(3)反复应用∀和$\forall_T$ 规则，直到它们无法应用或 ABox 中出现冲突为止。若 ABox 中出现冲突，则算法停止执行。

(4)反复应用⩽规则，直到它无法应用或 ABox 中出现冲突为止。若 ABox 中出现了冲突，则算法停止执行。

(5)对下一个个体进行处理。

3.5 算法正确性证明

给定 TBox T、初始的 ABox A 和角色层次 H，算法能产生一个不确定的完整的 ABox A”。若能证明：算法是可以终止的而且 A 与 T 和 H 一致，当且仅当算法能产生一个完整且无冲突的 ABox A”，则说明可通过该算法判定 ABox 的一致性。

引理 1 给定 TBox T、初始的 ABox A 和角色层次 H，让 A’为 A 的标准形式。A 与 T 和 H 一致，当且仅当 A’与 H 一致。

证明：将 A 转换成 A’，就是将每一个 $C \sqsubseteq D \in T$ 转换成 A 中相应的公理$\forall x.x{:}(\neg C \sqcup D)$，并且将 A 中出现的所有概念都转换成否定标准形式。显然，一个解释能满足 $C \sqsubseteq D$，当且仅当它满足$\forall x.x{:}(\neg C \sqcup D)$，此外，对于任意概念和其否定标准形式，两者的解释必然相同。因此，若存在一个解释能同时满足 T、A 和 H，则它能同时满足 A’和 H，反之亦然。引理得证。

定理 1(合理性) 给定 TBox T、初始的 ABox A 和角色层次 H，若算法能产生一个完整且无冲突的 ABox A”，则 A、T 和 H 一致。

证明：假设算法产生了一个完整且无冲突的 ABox A”，从 A”中去除那些包含了被间接阻塞个体的公理，可得到一个新的 ABox A*，显然 A*也是完整且无冲突的 ABox。下面先为 A*和 H 定义一个解释 I。考虑到 SHIQ 不具有有限模型属性[7]，因此，当 A*中存在阻塞时，I 应被定义成无限模型，为此引入了路径[14]的概念。一条路径 p 是 A*中个体的一个序列，其形式为 $p=[(a_0,a_0'),(a_1,a_1'),\cdots,(a_n,a_n')]$。对于路径 p，定义 $tail(p) := a_n$ 且 $tail'(p) := a_n'$，并用$[p|(a_{n+1},a_{n+1}')]$表示路径$[(a_0,a_0'),(a_1,a_1'),\cdots,(a_n,a_n'),(a_{n+1},a_{n+1}')]$。$A$*的路径集 $path(A^*)$被递归地定义如下：

(1)对于 A*中的老个体 a，若 A*中不存在另一个个体 b

使得 $a \doteq b \in A^*$，则 $[(a,a)] \in path(A^*)$。

(2)对于路径 $p \in path(A^*)$和 A^*中的新个体 a、b，有：

1)若 a 是 $tail(p)$的后继，而且它没有被阻塞，则 $[p|(a,a)] \in path(A^*)$；或者

2)若 a 是 $tail(p)$的后继，而且 a 被 b 直接阻塞，则 $[p|(b,a)] \in path(A^*)$；

3)给定一条路径 $[(a_0,a_0'),(a_1,a_1'),\cdots,(a_n,a_n')]$，若 $n=0$ 或者当 $n>0$ 时，对于 $0 \leqslant i \leqslant n-1$ 有 $a_i=a_i'$，则该路径被称为简单路径。

下面为 A^*和 H 定义一个解释 I：

(1)$\Delta^I = path(A^*)$；

(2)对于 A^*中的任意个体 a：若 A^*中存在另一个个体 b 使得 $a \doteq b \in A^*$，则 $a^I=b^I$；否则 $a^I=p(p \in \Delta^I$ 为简单路径且 $tail'(p)=a)$；

(3)对于 A^*中的任意概念名 C：$C^I=\{p \in \Delta^I | C \in L(tail'(p))\}$；

(4)对于 A^*和 H 中的任意角色 R：让 $\varepsilon(R)=\{(p,[p|(a,a')]) \in \Delta^I \times \Delta^I | a'$是 $tail(p)$的 R 后继$\} \cup \{([q|(a,a')],q) \in \Delta^I \times \Delta^I | tail(q)$是 a'的 R 前驱$\} \cup \{([(a,a)],[(b,b)]) \in \Delta^I \times \Delta^I | a$ 和 b 为老个体且 b 是 a 的 R 邻居$\}$。若 R 为传递角色，则 $R^I=\varepsilon(R)^+(\varepsilon(R)^+$表示 $\varepsilon(R)$的传递闭包)；否则 $R^I=\varepsilon(R) \cup \bigcup_{P \sqsubseteq^* R \text{ 且 } P \neq R} P^I$。

下面证明 I 满足 A^*中的每一条断言公理。

对于任意 $a \neq b \in A^*$，因为 $tail'(a^I)=a$，$tail'(b^I)=b$，而且 Tableau 规则确保了 $a \neq b$，所以 $a^I \neq b^I$。对于任意 $a\ R\ b \in A^*$，因为 a 未被阻塞，所以 $tail(a^I)=tail'(a^I)=a$。b 可能是老个体也可能是新个体。若 b 为老个体，则 a 必然也为老个体，而且 $b^I=[(b,b)]$，$a^I=[(a,a)]$；若 b 为新个体，则 $b^I=[a^I|(tail(b^I),b)]$。因为 b 是 a 的 R 后继，所以不管是哪种情况，都有 $(a^I,b^I) \in R^I$。

对于 $a:C$ 形式的公理，只需证明这样一个命题：对于 $p \in \Delta^I$和概念 C，若 $C \in L(tail(p))$，则 $p \in C^I$。若该命题成立，则对于任意 $a:C \in A^*$，因为 $C \in L(a)=L(tail'(a^I))=L(tail(a^I))$，所以 $a^I \in C^I$。下面通过对概念 C 的结构进行归纳证明 $p \in C^I$。

(1)若 $C \in Nc$，因为 $C \in L(tail(p))=L(tail'(p))$，所以 $p \in C^I$。若 $C=\top$，显然 $p \in \Delta^I=\top^I$。$C=\bot$这种情况不可能出现，因为 A^*是无冲突的。

(2)若 $C=\neg D$，则 $D \in Nc$，因为 A^*中所有的概念都具有否定标准形式。A^*是无冲突的，所以 $D \notin L(tail(p))=L(tail'(p))$，$p \notin D^I$，$p \in (\neg D)^I$。

(3)若 $C=C_1 \sqcap C_2$，因为 A^*是完整且无冲突的，所以 $C_1 \in L(tail(p))$且 $C_2 \in L(tail(p))$。根据归纳假设，有 $p \in C_1{}^I$且 $p \in C_2{}^I$，所以 $p \in (C_1 \sqcap C_2)^I$。

(4)若 $C=C_1 \sqcup C_2$，因为 A^*是完整且无冲突的，所以 $C_1 \in L(tail(p))$或 $C_2 \in L(tail(p))$。根据归纳假设，有 $p \in C_1{}^I$或 $p \in C_2{}^I$，所以 $p \in (C_1 \sqcup C_2)^I$。

(5)若 $C=\forall R.D$，则只要证明：对于任意 $q \in \Delta^I$，若 $(p,q) \in R^I$，则 $q \in D^I$。要使 $(p,q) \in R^I$，只有以下 2 种可能：

1)$(p,q) \in \varepsilon(R)$。分别考虑 p 和 q 的 3 种可能关系：

①$q=[p|(tail(q),tail'(q))]$，$tail'(q)$是 $tail(p)$的 R 后继。因为 $\forall R.D \in L(tail(p))$而且 A^*是完整且无冲突的，所以 $D \in L(tail'(q))=L(tail(q))$。

②$p=[q|(tail(p),tail'(p))]$，$tail(q)$是 $tail'(p)$的 R 前驱。因为 $\forall R.D \in L(tail(p))=L(tail'(p))$而且 A^*是完整且无冲突的，所以 $D \in L(tail(q))$。

③$p=[(tail(p),tail(p))]$，$q=[(tail(q),tail(q))]$，$tail(p)$和 $tail(q)$都为老个体，且 $tail(q)$是 $tail(p)$的 R 邻居。因为 $\forall R.D \in L(tail(p))$而且 A^*是完整且无冲突的，所以 $D \in L(tail(q))$。

因此，若 $(p,q) \in \varepsilon(R)$，总有 $D \in L(tail(q))$。根据归纳假设，有 $q \in D^I$。

2)存在一个传递角色 S 使得 $\{(p,p_1),(p_1,p_2),\cdots,(p_n,q)\} \subseteq \varepsilon(S)$且 $S \sqsubseteq^* R$。p 和 p_1 的关系同样存在 3 种可能，类似地可证明 $\forall S.D \in L(tail(p_1))$，同理可得 $\forall S.D \in L(tail(p_i))(2 \leqslant i \leqslant n)$。既然 $\forall S.D \in L(tail(p_n))$且 $(p_n,q) \in \varepsilon(S)$，那么根据上述证明可知 $q \in D^I$。

(6)若 $C=\exists R.D$，则只要证明：存在 $q \in \Delta^I$ 使得 $(p,q) \in R^I$ 且 $q \in D^I$。让 $b=tail(p)$，$b'=tail'(p)$。因为 A^*是完整且无冲突的而且 b 未被阻塞，所以 b 必然有一个 R 邻居 c 使得 $D \in L(c)$。b 和 c 的关系有 2 种可能：

1)c 是 b 的 R 后继。

①若 c 是老个体，则 b 也是老个体，$p=[(b,b)]$，而且存在 $q \in \Delta^I$ 使得 $q=[(c,c)]$，$(p,q) \in R^I$。显然 $D \in L(c)=L(tail(q))$。

②若 c 为新个体，则存在 $q \in \Delta^I$ 使得 $q=[p|(tail(q),c)]$，$(p,q) \in R^I$。显然 $D \in L(c)=L(tail'(q))=L(tail(q))$。

2)c 是 b 的 R 前驱。

①若 b 为老个体，则 c 也为老个体，而且存在 $q \in \Delta^I$ 使得 $q=[(c,c)]$，$(p,q) \in R^I$。显然 $D \in L(c)=L(tail(q))$。

②若 b 为新个体且 $b=b'$，则存在 $q \in \Delta^I$ 使得 $tail(q)=c$，$p=[q|(b,b)]$，$(p,q) \in R^I$。显然 $D \in L(c)=L(tail(q))$。

③若 b 为新个体且 $b \neq b'$，则存在 $q \in \Delta^I$ 使得 $tail(q) \neq c$，$p=[q|(b,b')]$，$(p,q) \in R^I$。因为 b 直接阻塞了 b'，根据阻塞的条件可知 $L(c)=L(tail(q))$，显然 $D \in L(tail(q))$。

所以不管在哪种情况下，始终存在 $q \in \Delta^I$ 使得 $(p,q) \in R^I$ 且 $D \in L(tail(q))$。根据归纳假设，有 $q \in D^I$。

(7)若 $C=\geqslant nR.D$，让 $F(p,R,D) \coloneqq \{q \in \Delta^I | (p,q) \in R^I$ 且 $q \in D^I\}$，则只要证明 $\#F(p,R,D) \geqslant n$。让 $b=tail(p)$，$b'=tail'(p)$。关于 b，有 2 种可能：

1)b 为老个体。对于每一个 $c_i \in N(b,R,D)$(函数 N 的定义见定义 14)，都存在 $q_i \in \Delta^I$ 使得 $q_i=[(c_i,c_i)]$或 $[p|(tail(q_i),c_i)]$，$(p,q_i) \in R^I$。显然 $D \in L(c_i)=L(tail'(q_i))=L(tail(q_i))$，根据归纳假设，有 $q_i \in D^I$。此外，对于任意 $c_i,c_j \in N(b,R,D)$，若 $c_i \neq c_j$，则 $q_i \neq q_j$。

2)b 为新个体。

①若 b 只有 R 后继，则对于每一个 $c_i\in N(b,R,D)$，都存在 $q_i\in\Delta^I$ 使得 $q_i=[p|(tail(q_i),c_i)]$，$(p,q_i)\in R^I$。同理可知 $q_i\in D^I$，而且对于任意 $c_i,c_j\in N(b,R,D)$，若 $c_i\neq c_j$，则 $q_i\neq q_j$。

②若 b 有一个 R 前驱，不妨假设它为 c_1，则所有的 $c_i\in N(b,R,D)\backslash\{c_1\}$必然都为 b 的 R 后继。对于 c_1，根据 $C=\exists R.D$ 的证明可知，存在 $q_1\in\Delta^I$ 使得$(p,q_1)\in R^I$，$q_1\in D^I$，而且 $q_1=[(c_1,c_1)]$，或 $tail(q_1)=c_1$，$p=[q_1|(b,b)]$，或 $tail(q_1)\neq c_1$，$p=[q_1|(b,b')]$。对于任意 $c_i\in N(b,R,D)\backslash\{c_1\}$，根据上述证明可知，存在 $q_i\in\Delta^I$ 使得 $q_i=[p|(tail(q_i),c_i)]$，$(p,q_i)\in R^I$，$q_i\in D^I$。显然 $q_1\neq q_i$；此外，对于任意 $c_k,c_j\in N(b,R,D)\backslash\{c_1\}$，若 $c_k\neq c_j$，则 $q_i\neq q_j$。

以上证明表明：$\#F(p,R,D)\geqslant\#N(b,R,D)$。因为 A*是完整且无冲突的而且 b 未被阻塞，所以$\#N(b,R,D)\geqslant n$，$\#F(p,R,D)\geqslant n$。

(8)若 $C=\leqslant nR.D$，则只要证明$\#F(p,R,D)\leqslant n$。需注意的是，在形式为$\leqslant nR.D$ 的概念中，R 必须是简单角色，因此，$R^I=\varepsilon(R)$。让 $b=tail(p)$，$b'=tail'(p)$。考虑 3 种可能的情况：

1)b 为老个体。对于每一个 $q_i\in F(p,R,D)$，b 有且只有一个 R 邻居 c_i 使得 $q_i=[(c_i,c_i)]$或$[p|(tail(q_i),c_i)]$。因为 A*是完整且无冲突的而且 b 未被阻塞，所以 $D\in L(c_i)$或$\sim D\in L(c_i)$。$\sim D\in L(c_i)$显然不能成立，因为根据归纳假设，会导出矛盾 $q_i\in(\sim D)^I$，这意味着 $D\in L(c_i)$。对于任意 $q_i,q_j\in F(p,R,D)(q_i\neq q_j)$，若 $q_i=[(c_i,c_i)]$，$q_j=[(c_j,c_j)]$，显然 $c_i\neq c_j$，因为 $q_i\neq q_j$；若 $q_i=[(c_i,c_i)]$，$q_j=[p|(tail(q_j),c_j)]$，或 $q_j=[(c_j,c_j)]$，$q_i=[p|(tail(q_i),c_i)]$，显然 $c_i\neq c_j$，因为老个体只能出现在路径的起始处；若 $q_i=[p|(tail(q_i),c_i)]$，$q_j=[p|(tail(q_j),c_j)]$，也可得出 $c_i\neq c_j$，因为一旦 $c_i=c_j$，无论 c_i 是否被阻塞，都会导出矛盾 $q_i=q_j$。这说明 $\#F(p,R,D)\leqslant\#N(b,R,D)$。

2)b 为新个体且 $b=b'$。

①若不存在 $q_1\in F(p,R,D)$使得 $p=[q_1|(b,b)]$，则对于每一个 $q_i\in F(p,R,D)$，b 有且仅有一个 R 后继 c_i 使得 $q_i=[p|(tail(q_i),c_i]$。采用和上面相同的方法可证明$\#F(p,R,D)\leqslant\#N(b,R,D)$。

②若存在 $q_1\in F(p,R,D)$使得 $p=[q_1|(b,b)]$，则这样的 q_1 只有一个，而且 b 有且只有一个 R 前驱 $tail(q_1)$，此外，对于每一个 $q_i\in F(p,R)\backslash\{q_1\}$，$b$ 有且仅有一个 R 后继 c_i 使得 $q_i=[p|(tail(q_i),c_i]$。同理可证$\#F(p,R,D)\leqslant\#N(b,R,D)$。

3)b 为新个体且 $b\neq b'$。

①若不存在 $q_1\in F(p,R,D)$使得 $p=[q_1|(b,b')]$，则对于每一个 $q_i\in F(p,R,D)$，b 有且只有一个 R 后继 c_i 使得 $q_i=[p|(tail(q_i),c_i]$。同理可证$\#F(p,R,D)\leqslant\#N(b,R,D)$。

②若存在 $q_1\in F(p,R,D)$使得 $p=[q_1|(b,b')]$，则这样的 q_1 只有一个，而且 b'有且只有一个 R 前驱 $tail(q_1)$。因为 b'被 b 直接阻塞，所以 b 有且仅有一个前驱 c_1 使得 $L(c_1,b)=L(tail(q_1),b')$，显然 c_1 是 b 的 R 前驱。对于每一个 $q_i\in F(p,R,D)\backslash\{q_1\}$，$b$ 有且仅有一个 R 后继 c_i 使得 $q_i=[p|(tail(q_i),c_i]$。同理可证$\#F(p,R,D)\leqslant\#N(b,R,D)$。

在任何情况下，都可得出$\#F(p,R,D)\leqslant\#N(b,R,D)$。因为 A*是完整且无冲突的而且 b 未被阻塞，所以$\#N(b,R,D)\leqslant n$，$\#F(p,R,D)\leqslant n$。

现在考虑$\forall x.x:C$ 形式的公理。对于任意 $p\in\Delta^I$，A*中必然存在一个个体 a 使得 $tail'(p)=a$，而且 A*中不存在另一个个体 b 使得 $a\doteq b$。因为 A*是完整且无冲突的，所以 $a:C\in A$*。因为 $C\in L(a)=L(tail'(p))=L(tail(p))$，根据之前的证明可知 $p\in C^I$，这意味着 $C^I=\Delta^I$。至此，已证明了 I 满足 A*。

让 A'为 A 的标准形式，现在证明 I 满足 A'。对于 A'中的任意公理 Axm，Tableau 规则不会删除它，而只可能将它包含的某些个体用其他个体替代，因此，Axm 在 A*中必有一条对应的公理 Axm'。对于 Axm 中包含的任意个体 a，在 Axm'中它要么被另一个体 b 替代(A*中的公理 $a\doteq b$ 表明了 a 被 b 替代)，要么依然存在。因为 $a^I=b^I$，而且 I 满足 Axm'，所以 I 也满足 Axm。

现在证明 I 满足 H。对于任意 $R\sqsubseteq S\in H$，若$(p,q)\in R^I$，则存在 2 种可能：

(1)$(p,q)\in\varepsilon(R)$。因为 $R\sqsubseteq^*S$，所以$(p,q)\in\varepsilon(S)\subseteq S^I$。

(2)存在一个传递角色 Q，使得$\{(p,p_1),(p_1,p_2),\cdots,(p_n,q)\}\subseteq\varepsilon(Q)$且 $Q\sqsubseteq^*R$。显然$(p,q)\in Q^I$；此外，因为 $Q\sqsubseteq^*R\sqsubseteq^*S$，所以$\{(p,p_1),(p_1,p_2),\cdots,(p_n,q)\}\subseteq\varepsilon(S)$。因此，无论 S 是不是传递角色，都有$(p,q)\in S^I$。

因为 I 满足 A'和 H，根据引理 1 可知定理 1 成立。

定义 17 让 C 为具有否定标准形式的概念，A 为 ABox，H 为角色层次。$sub(C,H)$是满足以下条件的最小概念集：(1)$C\in sub(C,H)$；(2)若概念 $D\in sub(C,H)$，则~D 以及 D 所有的子概念也属于 $sub(C,H)$；(3)若$\forall R.D\in sub(C,H)$，$S\sqsubseteq^*R$，且 S 为传递角色，则$\forall S.D\in sub(C,H)$。集合 $sub(A,H):=\bigcup_{a:C\in A\text{或}\forall x.x:C\in A} sub(C,H)$。

定理 2(可终止性) 给定 TBox T、初始 ABox A 和角色层次 H，算法可以终止。

证明：算法可分为 2 个阶段：1)预处理阶段；2)Tableau 规则执行阶段。第 1 个阶段是将 A 转换成标准的 ABox A'，也就是将 T 中所有的 GCIs 转换成 A 中相应的通用概念公理，而且将 A 中出现的所有概念转换成否定标准形式。让 s_T为 T 的大小，s_A为 A 的大小，第 1 个阶段中转换操作的次数显然受 s_T+s_A的约束，因此，该阶段是可以终止的。第 2 个阶段是通过执行 Tableau 规则对 A'进行扩展，直到将 A'扩展成完整的 ABox A''为止，该阶段是否能终止取决于 Tableau 规则的执行次数是否有限。

让 $s_{A'}$为 A'的大小，容易看出 $s_{A'}=O(s_T+s_A)$。让 s_H为 H 的大小，$R_{A',H}:=\{R,inv(R)|R$ 为 A'或 H 中存在的角色$\}$，则 $\#R_{A',H}=O(s_{A'}+s_H)=O(s_T+s_A+s_H)$。对于具有否定标准形式的概念 C(假定其大小为 s_C)，文献[15]已经证明$\#sub(C,H)=O(s_C\times s_H)$，所以$\#sub(A',H)=O(s_{A'}\times s_H)$。让 $m=\#sub(A',H)$，$k=\#R_{A',H}$，$s=s_T+s_A+s_H$，则 $m=O(s^2)$，$k=O(s)$。

让 A_i 为对 A' 应用 i 次 tableau 规则后所得到的 ABox ($i\geqslant 0$)(注意：$A_0=A'$)。因为在 A_i 中新个体只能是老个体的后代而且只有一个前驱，不妨将 A_i 划分为若干个层次：老个体都位于第 0 层，新个体位于其他层；第 i+1 层($i\geqslant 0$)中的新个体是第 i 层中个体的后继。对于 A^i 中的任意个体 a 和其前驱 b，$L(a)$和 $L(b)$都是 $sub(A',H)$的子集，$L(a,b)$是 $R_{A',H}$ 的子集[15]，因此，$(L(a),L(a,b),L(b))$的取值最多只有 2^{2m+k} 种可能。用归纳法证明 A_i 的层数$\leqslant u(u=2^{2m+k}+2)$。显然，$A_0$ 的层数=1$\leqslant u$。假设 A_i 的层数$\leqslant u$。若 A_i 的层数$<u$，则 $A_{i+1}\leqslant u$，因为对 A_i 应用一次 Tableau 规则最多能导致它增加 1 个层次。若 A_i 的层数=u，对于 A_i 中第 u–1 层的任意个体 a_{u-1}，它有 u–1 个祖先 $a_0,a_1,\cdots,a_{u-2}$ 分别位于第 0 层~第 u–2 层。在 $a_0,a_1,\cdots,a_{u-1}$ 中必然存在 2 个个体 a_i 和 a_j 满足 $0<i<j\leqslant u-1$ 而且$(L(a_{i-1}),L(a_{i-1},a_i),L(a_i))=(L(a_{j-1}),L(a_{j-1},a_j),\ L(a_j))$。若 a_j 未被阻塞，则 a_j 所有的祖先(包括 a_i 在内)都不可能被阻塞，根据阻塞的定义可知 a_j 必定被 a_i 直接阻塞，矛盾，因此，a_j 一定会被阻塞，a_{u-1} 也会被阻塞，Tableau 规则不能再为 a_{u-1} 创建后继，A_{i+1} 的层数$\leqslant u$。

让 M_A 为非负整数的所有 $6u$ 元组的集合，即 $M_A=N_1\times N_2\times\cdots\times N_{6u}(N_1,N_2,\cdots,N_{6u}$ 都为非负整数集，×为笛卡尔积)。现定义一个函数 M，它将 A_i 映射为 M_A 中的一个元素。

$M(A_i)$:=$(M_{i,0},M_{i,1},\cdots,M_{i,u-1})$，其中，$M_{i,j}$:=$(M_{i,j,1},M_{i,j,2},M_{i,j,3},M_{i,j,4},M_{i,j,5},M_{i,j,6})(0\leqslant j\leqslant u-1)$。

$M_{i,j,k}(1\leqslant k\leqslant 6)$的定义如下：

(1)让 $S_{i,j,1}=\{a|$个体 a 位于 A_i 的第 j 层，A_i 中不存在另一个个体 a'使得 $a\doteq a'\in A_i\}$。$M_{i,j,1}=\#S_{i,j,1}$。

(2)让 $S_{i,j,2}=\{a{:}C|a{:}C\in A_i$，$a$ 位于 A_i 的第 j 层$\}$。$M_{i,j,2}=m\times M_{i,j,1}-\#S_{i,j,2}$。

(3)让 $S_{i,j,3}=\{a{:}\geqslant nR.C|a{:}\geqslant nR.C\in A_i$，$a$ 位于 A_i 的第 j 层$\}$，$S'_{i,j,3}=\{a{:}\geqslant nR.C|a{:}\geqslant nR.C\in A_i$，$a$ 位于 A_i 的第 j 层，a 有 n 个 R 邻居 $b_1,b_2,\cdots,b_n$，使得对于 $1\leqslant w<v\leqslant n$，有 $b_w{:}C\in A_i$，$b_w\neq b_v\in A_i\}$。$M_{i,j,3}=\#S_{i,j,3}-\#S'_{i,j,3}$。

(4)让 $S_{i,j,4}=\{a{:}\exists R.C|a{:}\exists R.C\in A_i$，$a$ 位于 A_i 的第 j 层，a 没有 R 邻居 b 使得 $b{:}C\notin A_i\}$。$M_{i,j,4}=\#S_{i,j,4}$。

(5)让 $S_{i,j,5}=\{(a{:}\forall R.C,\ b)|a{:}\forall R.C\in A_i$，$a$ 位于 A_i 的第 j 层，b 为 a 的 R 邻居，$b{:}C\notin A_i\}$。$M_{i,j,5}=\#S_{i,j,5}$。

(6)让 $S_{i,j,6}=\{(a{:}\forall R.C,\ S,\ b)|\ a{:}\forall R.C\in A_i$，$a$ 位于 A_i 的第 j 层，S 为传递角色，$S\sqsubseteq {}^*R$，b 为 a 的 S 邻居，$b{:}\forall S.C\notin A_i\}$。$M_{i,j,6}=\#S_{i,j,6}$。

将 M_A 上的字典序$\succ$定义为：对于 M_A 中的任意元素$(a_1,a_2,\cdots,a_{6u})$和$(b_1,b_2,\cdots,b_{6u})$，$(a_1,a_2,\cdots,a_{6u})\succ(b_1,b_2,\cdots,b_{6u})$当且仅当存在 $1\leqslant w\leqslant 6u$ 使得 $a_w>b_w$，而且对于任意 $1\leqslant v<w$，$a_v=b_v$。可以证明：无论采用哪种 Tableau 规则将 A_i 扩展为 A_{i+1}，都有 $M(A_i)\succ M(A_{i+1})$(因为篇幅的原因，文中没有对每一种 Tableau 规则进行分析)。$\succ$是严格全序，根据严格全序的性质可知 M 函数为单射。此外，因为$\succ$是反良基的，所以 $M(A_0)\succ M(A_1)\succ\cdots\succ M(A'')$是一条有限链，这意味着 Tableau 规则的执行次数是有限的，算法的第 2 个阶段可以终止。至此，定理 2 得证。

引理 2 让 A 和 A'为 ABox，H 为角色层次。

(1)若 A 与 H 一致，则对 A 应用一条确定规则所得到的 A'与 H 一致。

(2)若 A 与 H 一致而且有一条非确定规则可应用于 A，则该规则能以某种方式执行，从而产生一个与 H 一致的 A'。

证明：为了证明(1)，下面分别对每一条确定规则进行分析。假定 $I=(\Delta^I,\cdot^I)$满足 A 和 H。

若$\sqcap$规则被应用于 $a{:}C\sqcap D\in A$，则 $A'=A\cup\{a{:}C,a{:}D\}$。因为 $a^I\in(C\sqcap D)^I$，所以 $a^I\in C^I$ 且 $a^I\in D^I$。I 满足 $a{:}C$ 和 $a{:}D$，因而也满足 A'。

若$\forall$规则被应用于 $a{:}\forall R.C\in A$，则 a 有一个 R 邻居 b 使得 $A'=A\cup\{b{:}C\}$。显然$(a^I,b^I)\in R^I$。因为 $a^I\in(\forall R.C)^I$，所以 $b^I\in C^I$。I 满足 $b{:}C$，因而也满足 A'。

若$\forall_T$ 规则被应用于 $a{:}\forall R.C$，则存在一个传递角色 S 使得 $S\sqsubseteq {}^*R$，且 a 有一个 S 邻居 b 使得 $A'=A\cup\{b{:}\forall S.C\}$。显然$(a^I,b^I)\in S^I$。假设 $b^I\notin(\forall S.C)^I$，则必然存在 $x\in\Delta^I$ 使得$(b^I,x)\in S^I$ 且 $x\notin C^I$。因为 S 为传递角色，所以$(a^I,x)\in S^I\subseteq R^I$。因为 $a^I\in(\forall R.C)^I$，所以 $x\in C^I$，产生矛盾。这说明假设不成立，$b^I\in(\forall S.C)^I$。I 满足 $b{:}\forall S.C$，因而也满足 A'。

若$\forall_x$ 规则被应用于$\forall x.x{:}C\in A$，则 A 中存在一个个体 a 使得 $A'=A\cup\{a{:}C\}$。因为 $C^I=\Delta^I$，所以 $a^I\in C^I$。I 满足 $a{:}C$，因而也满足 A'。

若$\exists$规则被应用于 $a{:}\exists R.C\in A$，则该规则会创建一个新个体 b，使得 $A'=A\cup\{a\ R\ b,\ b{:}C\}$。因为 $a^I\in(\exists R.C)^I$，所以必然存在 $x\in\Delta^I$ 使得$(a^I,x)\in R^I$ 且 $x\in C^I$。定义一个新的解释函数 I'，并规定：对于 A 中所有的个体 a 都有 $a^{I'}=a^I$；对于新创建的个体 b 有 $b^{I'}=x$；对于 A 中所有的概念 C 都有 $C^{I'}=C^I$；对于 A 和 H 中所有的角色 R 都有 $R^{I'}=R^I$。$I'=(\Delta^I,\cdot^{I'})$显然满足 A'和 H。

若$\geqslant$规则被应用于 $a{:}\geqslant nR.C\in A$，则该规则会创建 n 个新个体 $b_1,b_2,\cdots,b_n$，使得 $A'=A\cup\{a\ R\ b_k,\ b_k{:}C|1\leqslant k\leqslant n\}\cup\{b_i\neq b_j|1\leqslant i,j\leqslant n$ 且 $i\neq j\}$。因为 $a^I\in(\geqslant nR.C)^I$，所以必然存在 n 个不同的 $x_i\in\Delta^I(1\leqslant i\leqslant n)$使得$(a^I,x_i)\in R^I$。定义一个新的解释函数 I'，并规定：对于 A 中所有的个体 a 都有 $a^{I'}=a^I$；对于所有新创建的个体 b_i 都有 $b_i^{I'}=x_i$；对于 A 中所有的概念 C 都有 $C^{I'}=C^I$；对于 A 和 H 中所有的角色 R 都有 $R^{I'}=R^I$。$I'=(\Delta^{I'},\cdot^{I'})$显然满足 A'和 H。

为了证明(2)，分别对每一条非确定规则进行分析。假定 $I=(\Delta^I,\cdot^I)$满足 A 和 H。

若$\sqcup$规则可应用于 $a{:}C\sqcup D\in A$，则该规则能产生的结果为：$A'=A\cup\{a{:}C\}$或 $A'=A\cup\{a{:}D\}$。因为 $a^I\in(C\sqcup D)^I$，所以

$a^I \in C^I$ 或 $a^I \in D^I$。若 $a^I \in C^I$，则让 $A'=A\cup\{a:C\}$，否则让 $A'=A\cup\{a:D\}$。I 显然满足 A'。

若?规则可应用于 $a:\leqslant nR.C\in A$，则 a 有一个 R 邻居 b 使得 $b:C\notin A$ 且 $b:\sim C\notin A$。因为 I 满足 A，所以 $b^I\in C^I$ 或 $b^I\in(\sim C)^I$。若 $b^I\in C^I$，则让 $A'=A\cup\{b:C\}$，否则让 $A'=A\cup\{b:\sim C\}$。I 显然满足 A'。

若≤规则可应用于 $a:\leqslant nR.C\in A$，则 $\#N(a,R,C)>n$。因为 $a^I\in(\leqslant nR.C)^I$，而且对于任意 $b_i\in N(a,R,C)$，都有 $(a^I,b_i^I)\in R^I$，所以必然存在 $b_1,b_2\in N(a,R,C)$ 满足 $b_1\neq b_2$ 且 $b_1^I=b_2^I$。因为 I 满足 A，所以 $b_1\neq b_2\notin A$。在执行该规则时，可以选择 b_1 和 b_2 作为合并的对象，并得到相应的 A'。I 显然满足 A'。至此，引理 2 得证。

定理 3(完备性) 给定 TBox T、初始的 ABox A 和角色层次 H，若 A 与 T 和 H 一致，则算法能产生一个完整且无冲突的 ABox A”。

证明：根据定理 2 和引理 1、引理 2 可知，算法必定能产生一个与 H 一致的完整的 ABox A”。既然 A”与 H 一致，那么 A”中显然不可能存在冲突。定理 3 得证。

定理 4 Tableau 算法是判定 SHIQ 的 ABox 一致性的非确定性算法。

证明：根据定理 1~定理 3 可知该定理成立。

4　结束语

本文给出了一种 SHIQ 的 ABox 一致性判定 Tableau 算法，并证明了该算法的正确性。先通过预处理将 ABox 转换成标准的 ABox，然后按照特定的完整策略将一套 Tableau 规则应用于 ABox，直到将它扩展成完整的 ABox 为止。算法通过一种阻塞机制来确保终止，该机制能避免多余的规则应用并提高算法的效率。在证明算法的合理性和完备性时，本文并未通过构建 Tableau 进行证明，而是采用了一种更为直接、简洁的方法。因为算法所采用的 Tableau 规则中包含了非确定规则，所以该算法是一种非确定性算法。如果为⊔、?和≤规则去除非确定性，即可将该算法改造成确定性算法，这将是下一阶段的工作。

参考文献

[1] 石　莲，孙吉贵. 描述逻辑综述[J]. 计算机科学，2006, 33(1): 194-197.

[2] 袁金平，鲍爱华，姚　莉. 语义 Web 技术及其逻辑基础[J]. 计算机工程，2008, 34(24): 194-196.

[3] 段跃兴. ALC 中的 Tableau 算法及其性质[J]. 计算机应用与软件，2010, 27(10): 272-274.

[4] Sattler U. A Concept Language Extended with Different Kinds of Transitive Roles[EB/OL]. (2012-07-21). http://lat.inf.tu-dresden.de/research/papers/1996/Sattler-KI-96.ps.gz.

[5] Horrocks I, Gough G. Description Logics with Transitive Roles[EB/OL]. (2012-07-21). http://dl.kr.org/dl97/proceedings/horrocks.ps.gz.

[6] Horrocks I, Sattler U. A Description Logic with Transitive and Inverse Roles and Role Hierarchies[J]. Logic and Computation, 1999, 9(3): 385-410.

[7] Horrocks I, Sattler U, Tobies S. A Description Logic with Transitive and Converse Roles, Role Hierarchies and Qualifying Number Restrictions[EB/OL]. (2012-07-21). http://dl.acm.org/citation.cfm?id=891255.

[8] Tessaris S, Gough G. ABox Reasoning with Transitive Roles and Axioms[C]//Proc. of International Workshop on Description Logics. Linköping, Sweden: [s. n.], 1999: 31-44.

[9] Tessaris S, Horrocks I. Abox Satisfiability Reduced to Terminological Reasoning in Expressive Description Logics[C]//Proc. of the 9th International Conference on Logic for Programming, Artificial Intelligence, and Reasoning. London, UK: Springer-Verlag, 2002: 435-449.

[10] Haarslev V, Moller R. Expressive ABox Reasoning with Number Restrictions, Role Hierarchies, and Transitively Closed Roles[EB/OL]. (2012-07-21). http://dl.acm.org/citation.cfm?id=896437.

[11] Horrocks I, Sattler U, Tobies S. Practical Reasoning for Very Expressive Description Logics[J]. Logic Journal of the IGPL, 2000, 8(3): 239-264.

[12] Motik B, Shearer R, Horrocks I. Hypertableau Reasoning for Description Logics[J]. Artificial Intelligence Research, 2009, 36(1): 165-228.

[13] Glimm B, Horrocks I, Motik B. Optimized Description Logic Reasoning via Core Blocking[C]//Proc. of the 5th International Conference on Automated Reasoning. Heidelberg, Germany: Springer-Verlag, 2010: 457-471.

[14] Horrocks I, Sattler U. A Tableau Decision Procedure for SHOIQ[J]. Automated Reasoning, 2007, 39(3): 249-276.

[15] Tobies S. Complexity Results and Practical Algorithms for Logics in Knowledge Representation[EB/OL]. (2012-07-21). http://lat.inf.tu-dresden.de/research/phd/Tobies-PhD-2001.pdf.

编辑　顾逸斐