Journal of Southeast University (English Edition) Vol. 24, No. 3, pp. 361 – 364 Sept. 2008 ISSN 1003—7985

Clause-based enhancing mode for tableau algorithm for ALCN

Gu Huamao[1,2,3] Shi Jinqin[3] Gao Ji[1]

([1] Institute of Artificial Intelligence, Zhejiang University, Hangzhou 310027, China)

([2] School of Computer and Information Engineering, Zhejiang Gongshang University, Hangzhou 310018, China)

([3] Food Sensory Science Laboratory, Zhejiang Gongshang University, Hangzhou 310018, China)

Abstract: As the tableau algorithm would produce a lot of description overlaps when judging the satisfiabilities of concepts (thus wasting much space), a clause-based enhancing mode designed for the language ALCN is proposed. This enhancing mode constructs a disjunctive normal form on concept expressions and keeps only one conjunctive clause, and then substitutes the obtained succinctest conjunctive clause for sub-concepts set in the labeling of nodes of a completion tree constructed by the tableau algorithm (such a process may be repeated as many times as needed). Due to the avoidance of tremendous descriptions redundancies caused by applying $\cap$- and $\cup$-rules of the ordinary tableau algorithm, this mode greatly improves the spatial performance as a result. An example is given to demonstrate the application of this enhancing mode and its reduction in the cost of space. Results show that the improvement is very outstanding.

Key words: tableau algorithm; enhancing mode; clause; satisfiability

In description logics (DLs), one of the significant fundamentals for ontology in the semantic web, reasoning is definitely a key issue. Among reasoning, the satisfiability of concepts is an elementary question. So far, the reasoning regarding satisfiablity always focuses around the tableau algorithms, by either adding new operators to fit new DL systems[1-5], or integrating DL with other fields, while the reasoning framework is still the tableau algorithm[6-9]. All these researches did not touch the essence of the tableau algorithm. Therefore, strictly speaking, there is no innovation in principle and no improvement in performance.

The tableau algorithm decides the satisfiability of a concept by unfolding it to form a series of sub-concept sets and gain the minimum requirements for the satisfiability of that concept. However, it is full of many overlapping descriptions among these sub-concept sets. For example, if node label $L(x)$ contains $A \cap B$, then there should be $A \in L(x)$ and $B \in L(x)$ according to the tableau algorithm. And these are apparently repeated descriptions of A and B in the same label set.

The enhancing mode presented in this paper aims at solving the description overlaps in the same node labeling. It substitutes the succinctest conjunctive clause of concepts for a sub-concept set in the labeling of nodes of the completion tree, and greatly improves the spacial performance as a result.

Received 2008-04-15.

Biographies: Gu Huamao (1975—), male, graduate; Gao Ji (corresponding author), male, professor, gaoji@ mail. hz. zj. cn.

Foundation items: The National Natural Science Foundation of China (No. 60775029), the Science and Technology Program of Zhejiang Province (No. 2007C33072).

Citation: Gu Huamao, Shi Jinqin, Gao Ji. Clause-based enhancing mode for tableau algorithm for ALCN[J]. Journal of Southeast University (English Edition), 2008, 24(3): 361 – 364.

Because the research on this enhancing mode is still in its preliminary stage, we employ the basic but pragmatic DL language ALCN as the carrier in order to demonstrate how this enhancing mode is realized and why it is more efficient in saving space.

1 Syntax and Semantics of ALCN

ALCN is the language system that extends AL[10] by adding negation and number restrictions. To make this paper self-contained, we give a minimalist survival guide to ALCN before going ahead.

Definition 1 Let NC be the set of an ALCN-concept names, NR be the set of ALCN role names, then,

1) Each concept name $C \in$ NC is an ALCN-concept;

2) If C, D are ALCN-concepts, $R \in$ NR is an ALCN role name, then $(C \cup D)$, $(C \cap D)$, $(\neg C)$, $(\forall R.C)$, $(\exists R.C)$, $(\geqslant nR)$, $(\leqslant nR)$ are all ALCN-concepts.

Besides, the universal concept top (denoted $\top$) and the incoherent concept bottom (denoted $\perp$) are often predefined. If $A \in$ NC and A cannot be described with other concepts, then A is called a primitive concept.

Definition 2 ALCN-interpretation $I = (\Delta^I, \cdot^I)$ consists of non-empty set Δ^I (domain), and interpretation function $\cdot^I$. Function $\cdot^I$ maps each concept A to subset $A^I \subseteq \Delta^I$, and maps each role R to $R^I \subseteq \Delta^I \times \Delta^I$. Interpretation function $\cdot^I$ also satisfies:

$\top^I = \Delta^I$; $\perp^I = \varnothing$; $(\neg C)^I = \Delta^I \setminus C^I$; $(C \cap D)^I = C^I \cap D^I$; $(C \cup D)^I = C^I \cup D^I$; $(\forall R.C)^I = \{x \in \Delta^I \mid \text{for all } y \in \Delta^I: \langle x, y\rangle \in R^I \text{ implies } y \in C^I\}$; $(\exists R.C)^I = \{x \in \Delta^I \mid \text{there is } y \in \Delta^I: \langle x, y\rangle \in R^I, \text{and } y \in C^I\}$; $(\geqslant nR)^I \mid = \{x \in \Delta^I \| \{\langle x, y\rangle \in R^I\} \mid \geqslant n\}$; $(\leqslant nR)^I = \{x \in \Delta^I \| \{\langle x, y\rangle \in R^I\} \mid \leqslant n\}$

If there exists an interpretation I satisfying $C^I \neq \varnothing$, then C is satisfiable, and I is called a model of C. Interpretation I is the model of subsumption axiom $C \sqsubseteq D$ iff $C^I \subseteq D^I$; I is the model of the equivalence axiom $C \equiv D$ iff $C^I = D^I$. Interpretation I is the model of the concept assertion $C(x)$ iff $x \in C^I$; I is the model of the role assertion $R(x, y)$ iff $\langle x, y\rangle \in R^I$.

2 Relevant Definitions

We suppose that concepts in this paper are in form of a negation normal form (NNF), i. e., negation occurs only in front of concept names.

Definition 3 Suppose that A is an ALCN-primitive concept; C, D are ALCN-concepts; R is a role; then A, $\neg A$, $\exists R.C$, $\forall R.C$, $\top$, $\perp$ are called literals. Moreover, $\exists R.C$, $\forall R.D$ are called $\exists / \forall$ role literals. In this paper, we represent them with capital letters, such as X, Y, Z.

Definition 4 Clauses are concept descriptions satisfying: A single literal is a clause; a concept description consisting of two or more literals connected by $\cap$ is a clause. In this paper, clauses are usually represented by lowercase letters, such as x, y. The occurrence of a literal X in a clause x is denoted as $X \in x$. A concept description that is a single clause or consists of two or more clauses connected by $\cup$ is called a disjunctive normal form(DNF). Any ALCN-concept D can be transformed to the form of DNF. We represent the DNF of D by DNF(D). That a clause x occurs in D is denoted as $x \in D$. Besides, for simplicity of representation, we denote a certain clause in the DNF of concept D by OneOf(DNF(D)). And if clause x contains the same literals, we use the function Trim(x) to remove the repeated literals and just keep one for the same literals.

Definition 5 For simplicity, we define three functions as follows: 1) CS(x, y) = 1 if it has $X \in y$ for each literal $X \in x$, otherwise 0; 2) DS(D, y) = 1 if there is $x \in D$ with CS(x, y) = 1, otherwise 0; 3) TheoneOf(D, y) = x if there is $x \in D$ with CS(x, y) = 1, otherwise $\perp$. Where x, y are clauses; D is a DNF of the concept.

Notice: If function TheOneOf finds more than one clause, it returns to the first one in alphabetical order.

Definition 6 Suppose that D is an ALCN-concept in NNF. RSet(D) denotes the roles occurring in D; Sub(D) = $\{x \mid x$ is a sub-concept of $D\}$; SubCL(D) = $\{x \mid$ each literal in clause x is a sub-concept of $D\}$, then we define an ALCN E-tableau ET of D as a triple (Σ, L, ε):

Σ: a group of individuals;

L: $\Sigma \rightarrow 2^{\text{subCL}(D)}$ maps each individual in Σ to a clause;

ε: RSet(D) $\rightarrow 2^{\Sigma \times \Sigma}$ maps each role in RSet(D) to a set of individual couples.

Besides, there should be one $s \in \Sigma$ such that DS(DNF(D), $L(s)$) = 1. For any $s, t \in \Sigma$, $C \in$ Sub(D), $R \in$ RSet(D), ET also meets the following properties:

Property 1 If $X \in L(s)$, then $\neg X \notin L(s)$;

Property 2 If literal($\forall R.\ C$) $\in L(s)$ and $\langle s, t\rangle \in \varepsilon(R)$, then DS(DNF($C$), $L(t)$) = 1;

Property 3 If literal($\exists R.\ C$) $\in L(s)$, then there exists certain $t \in \Sigma$ such that $\langle s, t\rangle \in \varepsilon(R)$, and DS(DNF($C$), $L(t)$) = 1;

Property 4 If $(\leqslant nR) \in L(s)$, then $|\{t \in \Sigma \mid \langle s, t\rangle \in \varepsilon(R)\}| \leqslant n$;

Property 5 If $(\geqslant nR) \in L(s)$, then $|\{t \in \Sigma \mid \langle s, t\rangle \in \varepsilon(R)\}| \geqslant n$.

Lemma 1 An ALCN-concept D is satisfiable iff there exists an ALCN E-tableau ET of D.

Proof "if" direction: If ET = (Σ, L, ε) is an ALCN E-tableau of D, and DS(DNF(D), $L(s_0)$) = 1, then a model of D, $I = (\Delta^I, \cdot^I)$ can be defined as

$\Delta^I = \Sigma$; $R^I = \varepsilon(R)$; $A^I = \{s \mid A \in L(s)\}$, A is a primitive concept in Sub(D)

By induction, we can prove if literal $X \in L(s)$, then $s \in X^I$. Obviously $s \in L(s)^I$ according to the definition of the model on $\cap$. Moreover, if DS(DNF(E), $L(s)$) = 1, then $s \in E^I$, according to the definition of the model on $\cup$ and the definition of function DS.

1) If X is a primitive concept, then $s \in X^I$, according to the definition of the model;

2) If X is the negation of primitive concept X', then $X' \notin L(s)$ by property 1 of E-tableau, so $s \in X^I$;

3) If $X = \exists S.\ C$, then there is $t \in \Sigma$ such that $\langle s, t\rangle \in \varepsilon(S)$, and DS(DNF($C$), $L(t)$) = 1. So $\langle s, t\rangle \in S^I$ by definition. And it holds that $t \in C^I$ by induction. So, $s \in (\exists S.\ C)^I$;

4) If $X = \forall S.\ C$, and $\langle s, t\rangle \in S^I$, then $\langle s, t\rangle \in \varepsilon(S)$ and DS(DNF(C), $L(t)$) = 1. It holds that $s \in (\forall S.\ C)^I$ by induction.

5) Properties 4 and 5 in definition 6 can guarantee the correctness of the interpretation of number restrictions.

Because DS(DNF(D), $L(s_0)$) = 1, $s_0 \in D^I$; therefore I is a model of D.

"only if" direction: If $I = (\Delta^I, \cdot^I)$ is a model of D, then an ALCN E-tableau of D, ET = (Σ, L, ε) can be defined as $\Sigma = \Delta^I$; $\varepsilon(R) = R^I$; $L(s) = X_1 \cap X_2 \cap \ldots \cap X_n$; literal $X_i \in$ Sub(D) and $s \in X_i{}^I$ for $1 \leqslant i \leqslant n$.

Now let's prove ET is an E-tableau of D.

1) Property 1 is apparently satisfied by semantics;

2) If $(\forall R.\ C) \in L(s)$ and $\langle s, t\rangle \in \varepsilon(R)$, then $t \in C^I$ by semantics, that is, there exists a certain clause C' of C satisfying $t \in C'^I$. According to the definition of $L(t)$, it obviously holds that DS(DNF(C), $L(t)$) = 1. So property 2 is satisfied. In the same way, property 3 is satisfied.

3) The semantics of the number restrictions can guarantee the satisfiabilities of properties 4 and 5.

3 Building ALCN E-Tableau

Just like the ordinary tableau algorithm, the process of building an E-tableau is also a process of building a completion tree. Each node x in the completion tree is labeled with a clause $L(x) \in$ SubCL(D). Each directed edge $\langle x, y\rangle$ is labeled with a role R occurring in Sub(D), say, $L(\langle x, y\rangle) = R$.

If nodes x and y are connected by edge $\langle x, y\rangle$ and $L(\langle x, y\rangle) = R$, then we say y is an R-successor of x, while x is called the predecessor of y. If the labeling of node x in completion tree T contains literals $\perp$ or contains literals $A \in L(x)$ and $\neg A \in L(x)$ at the same time, or contains $(\leqslant nS)$ and has more than n S-successor, then we say tree T contains a clash.

For a concept D, the building of tree T starts with a single node x_0, and $L(x_0)$ = OneOf(DNF(D)). Node x_0 is called the root of T. And then apply the rules in Tab. 1 to tree T repeatedly until no rules can be applicable.

Tab. 1 Enhanced tableau extending rules for ALCN

Rules	Operations
$\exists$-rule	If $(\exists S.\ C) \in L(x)$, and x has no S-successor y with DS(DNF(C), $L(y)$) = 1, then create a new node y with $L(\langle s, t\rangle) = S$, and $L(y)$ = OneOf(DNF(C)).
$\forall$-rule	If $(\forall S.\ C) \in L(x)$, and x has an S-successor y with DS(DNF(C), $L(y)$) $\neq 1$, then $L(y)$ = Trim($L(y) \cap$ OneOf(DNF(C))).
$\geqslant$-rule	If $(\geqslant nS) \in L(x)$ and x has no n S-successors $y_1, \ldots, y_n$ with $y_i \neq y_j$ for $1 \leqslant i \leqslant j \leqslant n$, then create n new nodes $y_1, \ldots, y_n$, with $L(\langle x, y_i\rangle) = S$, $L(y_i) = \{A\}$, and $y_i \neq y_j$ for $1 \leqslant i \leqslant j \leqslant n$, A is a concept name not occurring in D.
$\leqslant$-rule	If $(\leqslant nS) \in L(x)$, and x has more than n S-successors, then if x has two S-successors y and z, without $y \neq z$, let $L(z)$ = Trim($L(z) \cap L(y)$) and $L(\langle x, y\rangle) = \varnothing$, $L(y) = \varnothing$.

If a completion tree contains nodes having clashes, or there are no rules applicable, we say this tree is complete. If we can build a complete, clash-free completion tree for D, then D is satisfiable, otherwise, unsatisfiable.

Suppose that input concept $C_0 = \neg A \cap \forall S.(\forall S.(\neg D)) \cap \exists S.(\exists S.(A \cap B) \cap \exists S.(C \cup D) \cap \leqslant 1S)$, where A, B, C, D are primitive concepts, S is the role name. Fig. 1 gives the completion tree of C_0. And the following are explanations on the orders of applying extending rules:

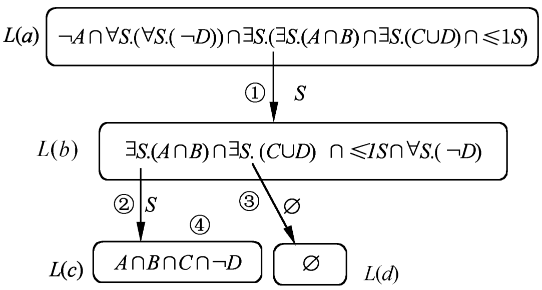

Fig. 1 Completion tree built according to C_0

1) Apply $\exists$-rule to literal $\exists S.(\exists S.(A \cap B) \cap \exists S.(C \cup D) \cap \leqslant 1S)$ in node a, and create node b with $L(b) = \exists S.(A \cap B) \cap \exists S.(C \cup D) \cap \leqslant 1S$. Then apply $\forall$-rule to literal $\forall S.(\forall S.(\neg D))$ in node a by adding $\forall S.(\neg D)$ to $L(b)$;

2) Apply $\exists$-rule to literal $\exists S.(A \cap B)$ in node b, and create node c with $L(c) = A \cap B$. Then apply $\forall$-rule to $\forall S.(\neg D)$ in node b by adding $\neg D$ to $L(c)$;

3) Apply $\exists$-rule to literal $\exists S.(C \cup D)$ in node b, and create node d with $L(d) = C$. Then apply $\forall$-rule to literal $\forall S.(\neg D)$ in node b by adding $\neg D$ to $L(d)$;

4) Apply $\leqslant$-rule to literal $\leqslant 1S$ in node b, and merge the label of d to node c and at the same time set both the label of d and the edge $\langle b, d \rangle$ to $\varnothing$.

Now, the building of completion tree is over, as there is no rule applicable and no clash found. Therefore, C_0 is satisfiable. However, choosing different clauses may cause a clash. For example, if we let $L(d) = D$ at step 3), then a clash occurs.

4 Performance Analysis

The main contribution of the enhancing mode is that it abandons the $\cap$- and $\cup$-rules and takes a clause for the labeling of nodes in the completion tree, thereby avoiding lots of overlapping in concept descriptions.

Taking C_0 in section 3 for example, the completion tree built by an ordinary tableau is shown in Fig. 2. By comparing Fig. 2 with Fig. 1, we know that the labeling of the enhanced tableau just corresponds to the terminal expressions of that of the ordinary tableau. Therefore, the space saved by adopting the enhancing mode is very remarkable.

5 Discussion and Conclusion

From the performance analysis, we conclude that spacial performance improvement contributes to discarding $\cap$- and $\cup$-rules, and to adopting clause labeling which directly decomposes concepts in one step. While the uncertainties caused by $\cup$-rules is transferred to the choosing of clauses, which is embodied by the function of OneOf(DNF(C)). After all, the different choices are likely to affect the final shape of the completion tree, and thus to affect the clash situation of the tree. Besides, the completion trees built by the ordinary tableau and the enhanced one are still corresponding with each other in structure, because the enhancing mode just omits those intermediate concepts created by $\cap$- and $\cup$-rules.

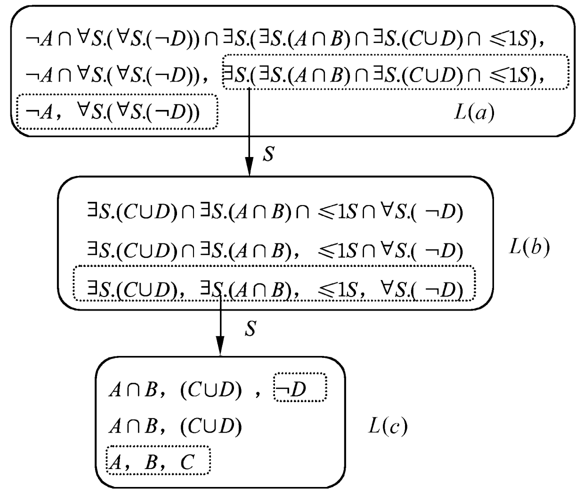

Fig. 2 Completion tree of C_0 built by the ordinary tableau, the descriptions in dash box correspond to labeling created by the enhanced tableau

Though this paper just applies this enhancing mode on a tableau based on ALCN, we believe that such a mode is also feasible for other description logic languages, such as ALCN, SHIQ, etc. Further research works are underway.

References

[1] Baader F, Calvanese D, McGuinness D, et al. *The description logic handbook: theory, implementation, and applications* [M]. Cambridge: Cambridge University Press, 2003.

[2] Baader F, Sattler U. An overview of tableau algorithms for description logics[J]. *Studia Logica*, 2001, **69**(1): 5 − 40.

[3] Horrocks I, Cough G. Description logics with transitive roles [C]//*Proceedings of the International Workshop on Description Logics*. Paris, France, 1997: 25 − 28.

[4] Horrocks I, Sattler U. Ontology reasoning in the SHOQ(D) description logic[C]//*Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence*. Seattle, Washington, USA, 2001: 199 − 204.

[5] Horrocks I, Sattler U. A tableau decision procedure for SHOIQ[J]. *Journal of Automated Reasoning*, 2007, **39**(3): 249 − 276.

[6] Dong M K, Jiang Y C, Shi Z Z. A description logic with default reasoning[J]. *Chinese Journal of Computers*, 2003, **26**(6): 729 − 736. (in Chinese)

[7] Kang Dazhou, Xu Baowen, Lu Jianjiang, et al. Description logics for fuzzy ontologies on semantic web[J]. *Journal of Southeast University: English Edition*, 2006, **22**(3): 343 − 347.

[8] Stoilos G, Stamou G, Pan J Z, et al. Reasoning with very expressive fuzzy description logics[J]. *Journal of Artificial Intelligence Research*, 2007, **30**(8): 273 − 320.

[9] Straccia U. Reasoning within fuzzy description logics[J]. *Journal of Artificial Intelligence Research*, 2001, **14**(1): 137 − 166.

[10] Schmidt-Schau M, Smolka G. Attributive concept descriptions with complements[J]. *Artificial Intelligence*, 1991, **48**(1): 1 − 26.

基于子句的 ALCN 语言 tableau 算法增强方式

古华茂[1,2,3]　石锦芹[3]　高　济[1]

([1] 浙江大学人工智能研究所,杭州 310027)
([2] 浙江工商大学计算机与信息工程学院,杭州 310018)
([3] 浙江工商大学食品感官科学实验室,杭州 310018)

摘要: 由于目前 tableau 算法在判断概念可满足性时会产生大量的描述重复(因而浪费了很多空间),针对描述逻辑语言 ALCN 提出了一种基于子句重构的 tableau 增强方式. 该增强方式对概念描述先构建一个析取范式,并只保留其中的一个合取子句,然后用这个获得的最简的概念合取子句去取代传统 tableau 算法构建的完整树的结点标记中的子概念集(这个过程根据需要可重复多次). 由于避免了传统 tableau 算法因∩-,∪-规则所带来的大量的概念描述重复,这种增强方式在空间性能上有了非常大的改善. 通过例子示范了这种增强方式的运用及其在空间性能上的改善程度. 结果表明,这种改善是相当明显的.

关键词: tableau 算法;增强方式;子句;可满足性

中图分类号: TP301