On Reasoning by Cases in Clausal Default Logic*

XU Dao-yun[1], DING De-cheng[1], ZHANG Ming-yi[2]

[1](*Department of Mathematics, Nanjing University, Nanjing* 210093, *China*);

[2](*Guizhou Academy of Sciences, Guiyang* 550001, *China*)

E-mail: dcding@nju.edu.cn

http://www.zj.js.cn

Received September 12, 1999; accepted February 20, 2001

Abstract: In this paper, a kind of tree method is proposed to investigate Roos' extensions about the reasoning by cases in default logic, discuss deeply the computation of Roos' extensions and analyze the relationship between Roos' extensions and Reiter's extensions. The algorithm decomposing the smallest set of literals from a set of clauses is presented to compute Roos' extensions. The method is useful for computing Roos' extensions and analyzing the complexity of reasoning by cases in default logic.

Key words: default logic; extension; reasoning by cases

Reiter's Default Logic[1] is one of the most popular formalism for describing nonmonotonic reasoning in artificial intelligence. One important defect of Reiter's default logic is, however, inability to reason by cases. We consider the following examples.

Example 1. Let $W=\{\alpha\vee\beta\}$, $D=\left\{\frac{\alpha:\gamma}{\gamma},\frac{\beta:\gamma}{\gamma}\right\}$. Intuitively, the consequence γ should be deducible. But, it is impossible in Reiter's default theory (D,W) because no default rules in D are applicable. But we can derive γ through reasoning by cases.

Example 2. Let $W=\{bird, penguin\vee ostrich\}$

$$D=\left\{\frac{bird:excep\_bird,can\_fly}{can\_fly},\frac{penguin:\neg excep\_penguin,\neg can\_fly}{\neg can\_fly},\frac{penguin:\neg excep\_penguni,excep\_bird}{excep\_bird},\right.$$

$$\left.\frac{ostrich:\neg\ excep\_ostrich,\neg can\_fly}{\neg can\_fly},\frac{ostrich:\neg excep\_ostrich,excep\_bird}{excep\_bird}\right\}.$$

In the default logic, the consequence "can_fly" is only one deducible consequence. But it is not the result that we hope to get. We convince that both "$excep\_bird$" and "$\neg can\_fly$" should be derivable through reasoning by cases. Therefore, the rule "$\frac{bird:\neg excep\_bird,can\_fly}{can\_fly}$" is no longer applicable.

We note that through reasoning by cases a default rule will be applied in Example 1, and the application of a default rule may be blocked in Example 2. These are intuitively more plausible.

To overcome the defect in Reiter's defaul: logic, Roos[2] modified the definition of Reiter's extension, and dis-

---

* **XU Dao-yun** was born in 1959. He is a Ph.D. canadidate at the Department of Mathematis, Nanjing University and an associate professor of the Department of Computer Science, Guizhou University. His research interests are complexity in logic, default logic and parameterized complexity. **DING De-cheng** was born in 1943. He is a professor and doctoral supervisor of the Department of Mathematics, Nanjing University. His current research areas include unsolvable degree, nonmonotonic reasoning and complexity in logic. **ZHANG Ming-yi** was born in 1943. He is a professor of Guizhou Academy of Sciences. His current research areas include computer science and artifical intelligence.

cussed the relationship between Reiter's extensions and Roos' extensions.

Zhang[3] investigated how to compute Roos' extensions of a default theory in similar way to compute Reiter's extensions, and presented some similar results.

In this paper, we will introduce a kind of tree method to deal with Roos' extensions so that we can deeply discuss the computation of Roos' extensions and analyze the relationship between the two kinds of extensions. In the definition of Roos' extensions, Roos considered mainly the deductive closure of a set of literals. We combine the method about computing Reiter's extensions with the idea of Roos to compute Roos' extensions, and presented algorithms about decomposing the smallest set of literals from a set of clauses by introducing three algorithms about resolution. The method is useful for computing Roos' extensions and analyzing the complexity of reasoning by cases in default logic.

1 Preliminaries

We assume conventionally that all formulas and default theories are closed except for specification.

Definition 1.1.[1](Reiter-extension)

Let $\Delta=(D,W)$ be a default theory. For any set of formulas S, let $\Gamma(S)$ be the smallest set satisfying the following three conditions:

(D_1) $W\subseteq\Gamma(S)$

(D_2) $Th(\Gamma(S))=\Gamma(S)$

(D_3) if $\frac{\alpha:\beta_1,\ldots,\beta_m}{\gamma}\in D$, $\alpha\in\Gamma(S)$, and $\neg\beta_1,\ldots,\beta_m\notin S$, then $\gamma\in\Gamma(S)$.

A set of formulas E is a Reiter-extension of the default theory (D,W) if and only if $E=\Gamma(E)$.

Theorem 1.2.[1] Let E be a set of formulas, and let $\Delta=(D,W)$ be a default theory. Define $E_0=W$ and for $i\geqslant 0$; $E_{i+1}=Th(E_i)\cup\left\{\gamma\left|\frac{\alpha:\beta_1,\ldots,\beta_m}{\gamma}\in D,\alpha\in E_i,\neg\beta_1,\ldots,\neg\beta_m\notin E\right.\right\}$. Then E is a Reiter-extension if and only if $E=U_{0\leqslant i<\infty}E_i$.

Definition 1.3.[2](Roos-extension)

Let $\Delta=(D,W)$ be a default theory. For any set of formulas S, let $\Gamma(S)=\{T_1,\ldots,T_n\}$, $T\in\Gamma(S)$ if and only if T is the smallest set satisfying the following three conditions:

(D'_1) $W\subseteq T$

(D'_2) T is equal to the deductive closure of the set of literals that T contains, i.e. $\alpha\in T$ if and only if there exists a subset of literals $T'\subseteq T$ such that $T'\vdash\alpha$.

(D'_3) if $\frac{\alpha:\beta_1,\ldots,\beta_m}{\gamma}\in D$, $\alpha\in T$ and $\neg\beta_1,\ldots,\beta_m\notin S$, then $\gamma\in T$.

A set of formulas T is a Roos-extension of the default theory (D,W) if and only if $E\in\Gamma(E)$.

Notice that the difference of the two kinds of definitions about extension is the conditions (D_2) and (D'_2). In Definition 1.3, $\Gamma(S)$ will exactly consist of one set T if the condition (D'_2) is replaced by the condition (D_2) in Definition 1.1.

Remark 1.4. (1) Let $W=\{\alpha\vee\neg\beta\}$ and $D=\left\{\frac{\alpha:\delta}{\delta},\frac{\neg\beta:\delta}{\delta},\frac{\gamma:\neg\delta}{\neg\delta}\right\}$, then for the default theory $\Delta=(D,W)$, $E=Th(\{\alpha\vee\neg\beta\})$ is a Reiter-extension, and both $E_1=Th(\{\alpha,\delta\})$ and $E_2=Th(\{\neg\beta,\delta\})$ are Roos-extensions. Clearly, $E\subseteq E_1$ and $E\subseteq E_2$. Additionally, we note that the above default theory (D,W) is a normal default theory and E_1,E_2 are two different Roos-extensions. But $E_1\cup E_2$ is consistent, i.e. the orthogonality of Roos-extensions is false in a normal default theory.

(2) Let $W=\varnothing$ and $D=\left\{\frac{:\neg\alpha,\neg\beta}{\alpha\vee\beta}\right\}$, then the default theory $\Delta=(D,W)$ has a Reiter-extension

$E=Th(\{\alpha\vee\beta\})$, but it has no Roos-extensions.

(3) (Example 1 continued) The default theory $\Delta=(D,W)$ has a Reiter-extension $E=Th(\{\alpha\vee\beta\})$ without $\gamma\in E$, and the default theory (D,W) has two Roos extensions $E_1=Th(\{\alpha,\gamma\})$ and $E_2=Th(\{\beta,\gamma\})$ with $\gamma\in E_1$ and $\gamma\in E_2$.

(4) (Example 2 continued) $E=Th(\{bird,penguin\vee ostrich,can\_fly\})$ is a Reiter-extension and

$$E_1=Th(\{bird,penguin,can\_fly\}),$$

$$E_2=Th(\{bird,penguin,\neg can\_fly,excep\_bird\})$$

$$E_3=Th(\{bird,ostrich,can\_fly\}),$$

$$E_4=Th(\{bird,ostrich,\neg can\_fly,excep\_bird\})$$

are Roos-extensions, and $E_1\cup E_2$ and $E_3\cup E_4$ are inconsistent.

Definition 1.5. [4](Clausal default theory)

(1) A default $d=\left\{\frac{\alpha:\beta_1,\ldots,\beta_m}{\gamma}\right\}$ is called clauseal if

(a) α is a conjunction of clauses.

(b) each $\beta_j(1\leqslant j\leqslant m)$ is the negation of a clause.

(2) A default theory $\Delta=(D,W)$ is clausal if W consists of clauses and every default in D is clausal.

From now on, a default theory $\Delta=(D,W)$ means that it is clausal.

2 Discussions about Roos-extension

In this section, we will present some basic definitions and notations, and investigate an characterization about Roos-extension.

Definition 2.1. [2](1) Given a formula α, define $SL(\alpha)=\{L_1,\ldots,L_n\}$: $L\in SL(\alpha)$ is the smallest set of literals satisfying the following conditions:

(1.1) L is consistent

(1.2) $L\subseteq Literals(\alpha)$

(1.3) $L\vdash\alpha$

(where $Literals(\alpha)$ is a set of literals occurring in the formula α)

(2) Given a set of clauses S, define $SL(S)=\{L_1,\ldots,L_n\}$: $L\in SL(S)$ is the smallest set of literals satisfying the following conditions:

(1.1) L is consistent

(1.2) $L\subseteq Literals(\alpha)$

(1.3) $L\vdash S$

When $SL(S)$ has exactly one element L, we also write $SL(S)=L$.

Definition 2.2. [3] Let $\Delta=(D,W)$ be a default theory and E a set of clauses. Define $E_0(\Delta)\in SL(W)$ and for $i\geqslant 0$, $E_{i+1}(\Delta)=Th(L)\cup\left\{\gamma\,\middle|\,\frac{\alpha:\beta_1,\ldots,\beta_m}{\gamma}\in D,\alpha\in E_i(\Delta)\text{ and }\neg\beta_1,\ldots,\neg\beta_m\notin E\right\}$ for some $L\in SL(E_i(\Delta))$. Finally, let $M^r=U_{0\leqslant i<\infty}E_i(\Delta)$.

We note that for a different L at stage $i(i\geqslant 0)$, we get a different $E_{i+1}(\Delta)$. Therefore, $M^r=U_{0\leqslant i<\infty}E_i(\Delta)$ is relevant to the choice of the different L at each stage.

Let $\Theta(E,\Delta)=\{M_1^r,\ldots,M_k^r\}$, $M^r\in\Theta(E,\Delta)$ if and only if $M^r=U_{0\leqslant i<\infty}E_i(\Delta)$.

Notations: Let D be a set of defaults and $\delta=\frac{\alpha:\beta_1,\ldots,\beta_m}{\gamma}$ a default in D. Denote

$$Pre(\delta)=\{\alpha\},Pre(D)=U_{\delta\in D}Pre(\delta),$$

$$Just(\delta)=\{\beta_1,\ldots,\beta_m\},\ Just(D)=U_{\delta\in D}Just(\delta)$$

$$Con(\delta)=\{\gamma\}\quad\text{and}\quad Con(D)=U_{\delta\in D}Con(\delta).$$

Definition 2.3. [2] Let $\Delta=(D,W)$ be a default theory, E a set of formulas and M^r in $\Theta(E,\Delta)$. Define

$$GD(M^r,\Delta,E)=\left\{\frac{\alpha:\beta_1,\ldots,\beta_m}{\gamma}\in D\mid \alpha\in M^r \text{ and } \neg\beta_1,\ldots,\neg\beta_m\notin E\right\}.$$

In particular, $GD(E,\Delta,E)$ is denoted by $GD(E,\Delta)$.

We say that $GD(M^r,\Delta,E)$ is the set of default generators of M^r with respect to the set of formulas E.

Theorem 2.4. [3] Let $\Delta=(D,W)$ be a default theory and E a set of formulas. Then E is a Roos-extension of (D,W) if and only if $E\in\{Th(L)\}_{L\in SL(W\cup Con(GD(E,\Delta)))}$.

Definition 2.5. [2] Let $\Delta=(D,W)$ be a default theory and D' a subset of D. Define

$$D_0(D',\Delta,L)=\left\{\frac{\alpha:\beta_1,\ldots,\beta_m}{\gamma}\subset D'\mid L\vdash\alpha\right\} \text{ for some } L\in SL(W) \text{ and for } i\geqslant 0$$

$$D_{i+1}(D',\Delta,L)=\left\{\frac{\alpha:\beta_1,\ldots,\beta_m}{\gamma}\in D'\mid L\vdash\alpha\right\} \text{ for some } L\in SL(W\cup Con(D_i(D',\Delta,L)))$$

As in Definition 2.2, we get the different $D_i(D',\Delta,L_i)$ by taking the different L at stage $i(i\geqslant 0)$. Therefore, $M^d=U_{0\leqslant i<\infty}D_i(D',\Delta,L_i)$ is relevant to the choice of L_i at stage i.

Let $\Lambda(D',\Delta)=\{M_1^d,\ldots,M_k^d\}$. $M^d\in\Lambda(D',\Delta)$ if and only if $M^d=Y_{0\leqslant i\leqslant\infty}D_i(D',\Delta,L_i)$.

Definition 2.6. Let $\Delta=(D,W)$ be a default theory. For any set of formulas E, define inductively a tree $T_r(E,\Delta)$ as follows:

(1) The root of $T_r(E,\Delta)$ is labeled with an empty set, denoted by $Root(T_r(E,\Delta))=T_r(E,\Delta,0)$ and called the 0-level nodes of $T_r(E,\Delta)$.

(2) Assume $SL(W)=\{L_1,\ldots,L_n\}$. Then for each j, L_j is a child node of $T_r(E,\Delta,0)$, denoted by $Children(T_r(E,\Delta,0))=\{L_1,\ldots,L_n\}=T_r(E,\Delta,1)$. The elements of $T_r(E,\Delta,1)$ are called the 1-level nodes of $T_r(E,\Delta)$.

(3) For any $i\geqslant 1$ and each $L\in T_r(E,\Delta,i)$, define $E_{i-1}(\Delta)=Th(L)\cup\left\{\gamma\mid\frac{\alpha:\beta_1,\ldots,\beta_m}{\gamma}\in D, L\vdash\alpha \text{ and } \neg\beta_1,\ldots,\neg\beta_m\notin E\right\}$ and $SL(E_{i+1}(\Delta))=\{L'_1,\ldots,L'_{n'}\}$. Then for each $j(1\leqslant j\leqslant n')$, L'_j is a child node of L, denoted by $Children(L)=\{L'_1,\ldots,L'_n\}$ and $T_r(E,\Delta,i+1)=U_{L\in T_r(E,\Delta,i)}Children(L)$. The elements of $T_r(E,\Delta,i+1)$ are called the $(i+1)$-level nodes of $T_r(E,\Delta)$.

Definition 2.7. Let $\Delta=(D,W)$ be a default theory and E a set of formulas, and let B be a branch from the root to some leaf in $T_r(E,\Delta)$. Define $E_B(E,\Delta,i)$ as a node at i-th level on the branch B.

Clearly, (1) For each branch B of $T_r(E,\Delta)$, B corresponds to a sequence of set of formulas:

$$E_B(E,\Delta,0)=\varnothing,\ E_B(E,\Delta,1),\ldots,E_B(E,\Delta,i),\ldots$$

where $E_B(E,\Delta,i+1)\in SL(E_B(E,\Delta,i))$ for $i\geqslant 1$.

(2) Each M^r in $\Theta(E,\Delta)$ corresponds to a branch B of $T_r(E,\Delta)$. Conversely, a branch B of $T_r(E,\Delta)$ can be transformed into a M^r in $\Theta(E,\Delta)$ in Definition 2.2.

Theorem 2.8. Let $\Delta=(D,W)$ be a default theory and E a set of formulas. E is a Roos-extension of (D,W) if and only if there exists a branch B of $T_r(E,\Delta)$ such that

$$E=U_{0\leqslant i<\infty}E_B(E,\Delta,i+1).$$

Proof. Since E is a Roos-extension if and only if E is in $\Theta(E,\Delta)$.

Definition 2.9. [3] Let D' be a set of defaults and L a set of literals. D' is L-compatible if there is no $\frac{\alpha:\beta_1,\ldots,\beta_m}{\gamma}\in D'$ such that $L\vdash\neg\beta_i$ for some $i(1\leqslant i\leqslant n)$. D' is maximally L-compatible if there is no D'', $D'\subset D''$, such that D'' is L-compatible.

Definition 2.10. For a default theory $\Delta=(D,W)$ and D' a subset of D. Define inductively a tree $T_d=(D',\Delta)$ as follows:

(1) The root of $T_d(D',\Delta)$ is labeled with an empty set, denoted by $Root(T_d(D',\Delta))=T_d(D',\Delta,0)$, and

called 0-level node of $T_d(D',\Delta)$.

(2) Assume $SL(W)=\{L_1,\ldots,L_n\}$. Then for each j, $D_0(D',\Delta,L_j)=\left\{\frac{\alpha:\beta_1,\ldots,\beta_m}{\gamma}\in D' \mid L_j\vdash\alpha\right\}$ is a child node of $T_d(D',\Delta,0)$, denoted by $Children(T_d(D',\Delta,0))=\{D_0(D',\Delta,L_1),\ldots,D_0(D',\Delta,L_n)\}=T_d(D',\Delta,1)$. The elements of $T_d(D',\Delta,1)$ are called the 1-level nodes of $T_d(D',\Delta)$.

For any $i\geqslant 1$ and each $D''\in T_d(D',\Delta,i)$, let $SL(W\cup Con(D''))=\{L'_1,\ldots,L'_{n'}\}$. Then for each $j(1\leqslant j\leqslant n')$, define $D_{i+1}(D',\Delta,L'_j)=\left\{\frac{\alpha:\beta_1,\ldots,\beta_m}{\gamma}\in D' \mid L'_j\vdash\alpha\right\}$ for $L'_j\in SL(W\cup Con(D''))$ as a child node of D'', and denote $Children(D'')=\{D_{i+1}(D',\Delta,L'_1),\ldots,D_{i+1}(D',\Delta,L'_{n'})\}$ and $T_d(D',\Delta,i+1)=\bigcup_{D''\in T_d(D',\Delta,i)} Children(D'')$. The elements of $T_d(D',\Delta,i+1)$ are called the $(i+1)$-level nodes of $T_d(D',\Delta)$.

Definition 2.11. Let $\Delta=(D,W)$ be a default theory and D' a subset of D, and let B be a branch from the root to some leaf in $T_d(D',\Delta)$. Define $D_B(D',\Delta,i)$ as a node at the i-level on the branch B.

Clearly, (1) For each branch B of $T_d(D',\Delta)$, B corresponds to a sequence of set of defaults:

$$D_B(D',\Delta,0)=\varnothing,\ D_B(D',\Delta,1),\ldots,D_B(D',\Delta,i),\ldots$$

where $D_B(D',\Delta,i+1)=\left\{\frac{\alpha:\beta_1,\ldots,\beta_m}{\gamma}\in D' \mid L\vdash\alpha\right\}$ for some $L\in SL(W\cup Con(D_B(D',\Delta,i)))$.

(2) For each branch B of $T_d(D',\Delta)$, B corresponds to a sequence of set of literals

$$L_B(D',\Delta,0),\ L_B(D',\Delta,1),\ldots,L_B(D',\Delta,i),\ldots$$

such that for $i\geqslant 0$

$$L_B(D',\Delta,i)\in SL(W\cup Con(D_B(D',\Delta,i)))$$

and

$$D_B(D',\Delta,i+1)=\left\{\frac{\alpha:\beta_1,\ldots,\beta_m}{\gamma}\in D' \mid L_B(D',\Delta,i)\vdash\alpha\right\}$$

(3) Similar to $T_e(E,\Delta)$ and $\Theta(E,\Delta)$, $D^*\in\Lambda(D',\Delta)$ corresponds to a branch B of $T_d(D',\Delta)$. Conversely, a branch B of $T_d(D',\Delta)$ can be transformed into a D^* in $\Lambda(D',\Delta)$ in Definition 2.5.

Therefore, it follows that

Theorem 2.12. Let $\Delta=(D,W)$ be a default theory. (D,W) has a Roos-extension E if and only if there exists a set of defaults $D^*\subseteq D$ such that $T_d(D^*,\Delta)$ has a branch B satisfying the following conditions:

(1) $D^*=U_{0\leqslant i<\infty}D_B(D^*,\Delta,i)$

(2) D^* is L_{D^*}-compatible, where $L_{D^*}=U_{0\leqslant i<\infty}L_B(D^*,\Delta,i)$

(3) For any $\frac{\alpha:\beta_1,\ldots,\beta_m}{\gamma}\in D-D^*$, either $L_{D^*}\not\vdash\alpha$ or $L_{D^*}\vdash\neg\beta_i$ for some $i(1\leqslant i\leqslant m)$.

3 The Computation of $SL(S)$

In the last section, we note that in the computation of Roos-extensions $SL(S)$ must be considered at each stage. In this section we will discuss how to decompose a set of clauses S in order to get $SL(S)$. We introduce three resolutions about clauses as follows:

Algorithm 3.1. [5,6] Unit-Resolution

Input: A CNF formulas $S=\bigwedge_{i=1}^{m}C_i$ or $S=\{C_1,\ldots,C_m\}$.

Output: Reduced CNF formulas to be equivalent to S.

Step 1. **if** S contains no unit clause **then** Stop and Return S

else if S contains two conflicting unit clauses $C_i=l$ and $C_j=\neg l$

then Stop and Return $S=\{\ \}$.

Step 2. Let $C_i=l$ be a unit clause on literal l.

Step 3. Delete all clauses in S that contain l.

Step 4. Delete all appearances of $\neg l$ in the remaining clauses, and go to Step 1.

Clearly, the complexity of the algorithm is $O(n^2)$, where n is the number of variables appearing in S.

Algorithm 3.2. Bi-Resolution

Input: A *CNF* formulas $S=\bigwedge_{i=1}^{m}C_i$ or $S=\{C_1,\ldots,C_m\}$.

Output: Reduced a collection of *CNF*, $\boldsymbol{F}$, whose disjunction is equivalent to S.

Step 1. Let $\boldsymbol{F}=\{S_R\}$, which S_R is Unit Resolution of S.

Step 2. **if** $\boldsymbol{F}$ contains no *CNF* S' in which there exist clauses C_i, C_j with same literals
then Stop and Return $\boldsymbol{F}$.

Step 3. Let $S'\in\boldsymbol{F}$ and S' have at least two clauses containing a common literal l.

Step 4. Let $C^1=l\vee C^1_{rest},\ldots,C^p=l\vee C^p_{rest}$ be all clauses containing literal l in S'.

(Notice that $C^1=l\vee C^1_{rest},\ldots,C^p=l\vee C^p_{rest}$ are nonempty.)

Step 5. $S'_1=(S'-\{C^1,\ldots,C^p\})\cup\{l\}$ and $S'_2=(S'-\{C^1,\ldots,C^p\})\cup\{C^1_{rest},\ldots,C^p_{rest}\}$.

Step 6. Delete S' from $\boldsymbol{F}$.

Step 7. Apply Unit Resolution to S'_1 and S'_2, yielding S''_1 and S''_2, respectively.

Step 8. Add S''_1 and S''_2 to F and go to Step 2.

Algorithm 3.3. Tri-Resolution

Input: A *CNF* formulas $S=\bigwedge_{i=1}^{m}C_i$ or $S=\{C_1,\ldots,C_m\}$.

Output: Reduced a collection of *CNF*, $\boldsymbol{F}$, whose disjunction is equivalent to S.

Step 1. Let $\boldsymbol{F}=\{S_R\}$, which S_R is Unit Resolution of S.

Step 2. $\boldsymbol{F}$ is a collection of *CNF* by applying Bi-Resolution to *CNF* S_R.

Step 3. **if** $\boldsymbol{F}$ contains no *CNF* S' in which there exist clauses C^i,C^j with a pair of literals, l and $\neg l$,
then Stop and Return $\boldsymbol{F}$.

Step 4. Let $S'\in\boldsymbol{F}$ and clauses C^i and C^j in S' such that $C^i=l\vee C^i_{rest},C^j=\neg l\vee C^j_{rest}$.

Step 5. $S'_1=(S'-\{C^i,C^j\})\cup\{l\},S'_2=(S'-\{C^i,C^j\})\cup\{\neg l\}$ and $S'_3=(S'-\{C^i,C^j\})\cup\{C^i_{rest},C^j_{rest}\}$.

Step 6. Delete S' from $\boldsymbol{F}$.

Step 7. Apply Unit Resolution to S'_1,S'_2 and S'_3, yielding S''_1, S''_2 and S''_3, respectively.

Step 8. Apply Bi-Resolution to S''_1, S''_2 and S''_3, yielding $\boldsymbol{F}_1$, $\boldsymbol{F}_2$ and $\boldsymbol{F}_3$, respectively.

Step 9. Add $\boldsymbol{F}_1$, $\boldsymbol{F}_2$ and $\boldsymbol{F}_3$ to $\boldsymbol{F}$ and go to Step 3.

We note that applications of Unit Resolution, Bi-Resolution and Tri-Resolution have preference. The purpose is to reduce complexity of computations and to ensure *CNF* as small as possible.

From the above algorithms, the following theorem and propositions are trivial.

Theorem 3.4. Let S be a set of clauses. We have the following results.

(1) If S_U is a set of clauses applying Unit Resolution to S, then S_U contains no unit clauses that occur in other clauses as a literal.

(2) If $\boldsymbol{F}_B$ is a collection of *CNF* applying Bi-Resolution to S, then every *CNF* in $\boldsymbol{F}_B$ contains no different clauses C_i and C_j that have the same literals.

(3) If $\boldsymbol{F}_T$ is a collection of *CNF* applying Tri-Resolution to S, then every *CNF* in $\boldsymbol{F}_T$ contains no different clauses C_i and C_j that have the same proposition variables.

Let S be a set of clauses, we can construct a collection *Spread*(s) of *CNF* from S in Tri-Resolution by collecting and adding literals in Unit-Resolution and Bi-Resolution. *Spread*(s) is called the *spread* of S.

Example 3.5. Assume $S=\{a,(a\vee b),(b\vee c),(b\vee d\vee\neg e),(b\vee e\vee f\vee g)\}$, then *Spread*$(S)=\{\{a,b\},\{a,c,e,d\},\{a,c,\neg e,(f\vee g)\},\{a,c,d,(f\vee g)\}\}$.

Proposition 3.6. Let S be a set of clauses. *Spread*(S) is the spread of S. Then for each set of clauses S' in *Spread*(S), $Var(C_i)\bigcap Var(C_j)$ is an empty set for any different clauses C_i and C_j in S'.

(where $Var(C)$ is the set of variables in the clause C.)

Proposition 3.7. Let S' be a consistent set of clauses. If $Var(C_i)\cap Var(C_j)$ is an empty set for any clauses C_i, C_j in S' $(i\neq j)$, then

(1) A set of literals L is in $SL(S')$ if and only if L contains exactly one literal of each clause in S'.

(2) $|SL(S)|=\prod_{C\in S'}|C|$, where $|C|$ is the length of clause C.

Theorem 3.8. Let S be a set of clauses and $Spread(S)$ the spread of S, then the followings hold:

(1) For any S' and S'' in $Spread(S)$, the symmetry difference L' and L'' is nonempty for any $L'\in SL(S')$ and $L''\in SL(S'')$.

(2) For any $S'\in Spread(S)$, the symmetry difference L' and L'' is nonempty for any $L',L''\in SL(S'')$.

(3) $SL(S)=U_{S'\in Spread(S)}SL(S')$.

(4) $|SL(S)|=\sum_{S'\in Spread(S)}\left(\prod_{C\in S'}|C|\right)$.

Example 3.9. (Example 3.5 continued).

$$SL(S)=\{\{a,b\},\{a,c,e,d\},\{a,c,\neg e,f\},\{a,c,\neg e,g\},\{a,c,d,f\},\{a,c,d,g\}\}.$$

4 Conclusions

We presented a kind of tree method to investigate the reasoning by cases in default logic and algorithms to compute the smallest set of literals from a set of clauses. The algorithms can be applied to the computation of Roos' extensions.

References:

[1] Reiter, R. A logic for default reasoning. Artificial Intelligence, 1980,13:81~132.

[2] Roos, N. Reasoning by cases in default logic. Artificial Intelligence, 1998,99:165~183.

[3] Zhang, Ming-yi. On reasoning by cases in default logic. Proceedings of the International Symposium on Future Software Technology ISFST-99. Tokyo: Software Engineers Association, 1999. 375~376.

[4] Marek, V. W., Truszczynski, M. Nonmonotonic Logic. Berlin: Springer-Verlag, 1993.

[5] Kleine Buening, H. Propositional Logic: Deduction and Algorithm. Cambridge University Press, 1999.

[6] Chandru, V., Hooker, J. Optimization Methods for Login Inference. New York: John Wiley & Sons, Inc., 1999.

[7] Zhang, Ming-yi. A new reasoning into default logic. Information and Computation, 1996,129(2):73~85.

子句型缺省逻辑中的分情形推理

许道云[1], 丁德成[1], 张明义[2]

(南京大学 数学系,江苏 南京 210093);

[2](贵州科学院,贵州 贵阳 550001)

摘要: 引进一种树型方法以研究缺省逻辑中分情形推理下的 Roos 扩张,深入讨论了 Roos 扩张的计算,并分析了 Roos 扩张与 Reiter 扩张的关系.为计算 Roos 扩张,引入了从子句集分解最小文字集的算法.方法对于在缺省逻辑中计算 Roos 扩张以及分析分情形推理的计算复杂性是有用的.

关键词: 缺省逻辑;扩张;分情形推理

中图法分类号: TP301 **文献标识码**: A