计算机科学1998 Vol. 25 №. 2

容错推理系统FTRS的设计

The Design of Fault-Tolerant Reasoning System

刘庆志　林作铨

（汕头大学计算机科学研究所　汕头 515063）

摘　要　This paper describes a fault-tolerant reasoning system FTRS base on Paraconsistent Circumscription LPCm theory. FTRS is a system of commonsense reasoning with incomplete and inconsistent knowledge.

关键词　Paraconsistent circumscription, Tableaux, Fault-tolerant reasoning, PROLOG

常识推理的难点是前提知识的不完备性、推理规则的不精确性以及知识中显性或隐性矛盾存在的普遍性。常识推理的第一个特点是“常识性”。“常识性”的结论，其含义是此结论在多数情况下是成立的，特殊的情况例外。在不知道是否为特殊事例的情况下，默认其不是特殊事例，从而得出一个符合常识的结论，这就是限制思想的出发点。常识推理的第二个特点是矛盾普遍地存在于常识之中。限制是进行常识推理的一种策略，但对于矛盾的事实，大多数的限制采取了排斥矛盾，采用各种策略维护信念的协调。这在某种意义上，违背了常识的特点——矛盾普遍存在。

超协调限制逻辑[4,6]是容错推理的一种基本形式[8]。我们提出一种超协调限制逻辑LPCm，它不同于其它限制理论的独特之处，在于它是三值逻辑，矛盾是完全允许存在的。允许其存在并不意味着对其不加以限制，LPCm是矛盾极小化的限制逻辑。

LPCm将非单调与超协调结合起来。LPCm在对不协调的信念集进行限制（即寻找极小模型）操作时，把对矛盾（原子命题取悖论值）的限制与谓词限制加以区分，将对矛盾的限制扩大到整个信念集，并且把对矛盾的限制比对谓词的限制处于优先考虑的位置。

基于LPCm，我们给出一种容错推理系统——FTRS（Fault-Tolerant Reasoning System），并用PROLOG语言实现。我们以数据库的方式记录事实集与规则集，允许用户添加、删除事实与规则，对于用户的询问命题，给出常识意义下其满足亦或不满足的回答，从而模拟人进行常识推理的过程。它不仅具有良好的界面和维护能力，而且也容易直接移植到其它程序语言（如C）。本文首先给出LPCm的语义，然后阐述了容错推理系统FTRS的实现原理，最后给出该系统的实现机制。

1　超协调限制LPCm的语义

设定L为一个（有限）一阶语言，L中公式定义如常，V={{1},{0},{01}}。一个赋值π赋予L中每个原子命题取V中三值之一：{0}表示仅假（经典假），{1}表示仅真（经典真），{01}表示既真又假（悖论真）。一个赋值π:L→V，使之：

①$1\in\pi(\neg A)$当且仅当$0\in\pi(A)$；$0\in\pi(\neg A)$当且仅当$1\in\pi(A)$。

②$1\in\pi(A\wedge B)$当且仅当$1\in\pi(A)$且$1\in\pi(B)$；$0\in\pi(A\wedge B)$当且仅当$0\in\pi(A)$或$0\in\pi(B)$。

③$1\in\pi(A\vee B)$当且仅当$1\in\pi(A)$或$1\in\pi(B)$；$0\in\pi(A\vee B)$当且仅当$0\in\pi(A)$且$0\in\pi(B)$。

称一个公式A为真，若$1\in\pi(A)$。称一个赋值为一个命题（命题集）的模型，如果在该赋值下这个命题（命题集中的每个成员）为真。

定义1　令S是一个句子集，Q为S中出现的所有谓词组成的集合，x为变元集，谓词集$P\subseteq Q$，π_1与π_2是LP的两个解释，相对于P意义下，我们定义$\pi_1 <_{(LPCm,P)}\pi_2$，当且仅当（分两种情形）

一：①$|\pi_1|=|\pi_2|$。

②对所有$p\in Q$，若$\pi_1(p(m))=01$，则$\pi_2(p(m))=01$。

③存在一个原子命题$p\in Q$，使得$\pi_2(p(m))=01$，而$\pi_1(p(m))\neq 01$。或者

二：①$|\pi_1|=|\pi_2|$。

②对所有$p\in Q$，若$\pi_1(p(m))=01$则$\pi_2(p(m))$

$=01$；若 $\pi_2(p(m))=01$则 $\pi_1(p(m))=01$。

③对所有 $p\in P$，若 $\pi_1(p(m))=1$则 $\pi_2(p(m))=1$。

④存在一个 $p\in P$ 使得 $\pi_2(p(m))=1$，但 $\pi_1(p(m))=0$。

这里 p(m)是以封闭项 m 替换出现在 p 中的 x 的结果。

对 $<_{(LPCm,P)}$定义，情形㊀表明对模型包含矛盾的极小化，情形㊁表明对模型中特例谓词的极小化。需要指出的是，两种情形满足一个即可。

LPCm 对矛盾的限制优先于对 abnormal 类谓词的限制。具体说，假如(模型 M1所含矛盾)⊂(模型 M2所含矛盾)，则由情形㊀的定义可知 $M1<_{(LPCm,P)}M2$，而不管 M1、M2中 abnormal 谓词的取值情况。只有在两个模型所含矛盾相同的情况下，才考虑 abnormal 谓词的取值情况来判断模型间的大小关系，正如情形㊁的定义。

定义2 S 的一个模型 π 称为关于谓词集 P 下的极小模型，当且仅当不存在 S 的另一个模型 π' 使得 $\pi'<_{(LPCm,P)}\pi$。

超协调限制 LPCm 的语义后承，记为$\models_{LPCm}$，可定义如下：

定义3 令 S 是一个理论集，Q 为 S 中出现的所有谓词组成集合，x 为变元集，受限谓词集 $P'\subseteq Q$，谓词 $A\in Q$，α 是一个 n 维常项，$CIRC_{LPCm}(S,P)\models_{LPCm}A(\alpha)$当且仅当 A(α)在 S 所有的关于 P 的极小模型中都为真。

例1 令 S 是如下公式的合取：

bird(Tweety)∧¬abnormal(Tweety)→fly(Tweety)
bird(Tweety)
yellow(Tweety)
¬yellow(Tweety)
penguin(Tweety)→¬fly(Tweety)

在 LPCm 下，令 P={abnormal}，我们得到解释 π_1{abnormal(Tweety)=0，penguin(Tweety)=0，bird(Tweety)=1，yellow(Tweety)=01，fly(Tweety)=1}是唯一的极小模型。在此模型下，fly(Tweety)=1 成立，即 $CIRC_{LPCm}(S,P)\models_{LPCm}$ fly(Tweety)，与人们关于鸟(Tweety)会飞的常识相符。

2 FTRS 系统概述

2.1 系统设计目标

系统设计目标包括：□一阶推理规则下的命题证明。□信念集具有容错特性。□允许用户自建信念集。□允许用户加入、删除信念。□提供易于操作的用户界面。□提供帮助文件。□对于不可证命题，给出原因。

2.2 系统结构(图1)

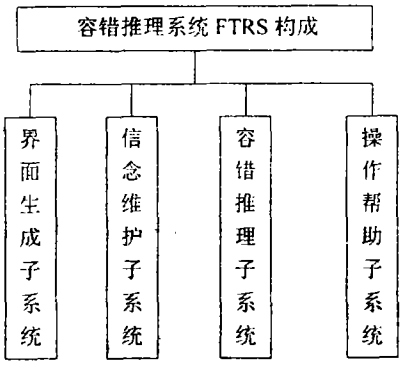

图1 FTRS 系统结构示意图

3 FTRS 的实现原理

FTRS 系统的实现原理是 LPCm 及其表演算[5,9]。

3.1 LPCm 的表演算

首先给出悖论逻辑 LP 表演算记号表的构造规则[5]。

定义4 在 L 中引入两个记号 T 和 F，一个记号公式是形如 TA 或 FA 的公式，其中 $A\in L$。我们扩充赋值 π 到记号公式如下：对任一公式 A，①$\pi(TA)=1$当且仅当 $\pi(A)=1$；$\pi(TA)=01$当且仅当 $\pi(A)=01$。②$\pi(FA)=1$当且仅当 $\pi(\neg A)=1$；$\pi(FA)=01$当且仅当 $\pi(\neg A)=01$。

定义5 一个表(tableaux)是一棵其结点为公式的树。令 Ψ 是一个记号公式(集)，对 Ψ 的记号表根据下面规则归纳地定义如下：①只有 Ψ 为唯一结点的表是一个对 Ψ 的记号表。②令 t 是一个对 Ψ 的记号表，b 是 t 的一个分支，那么

(a)若 $T\neg\neg\alpha$ 是 b 的一个结点，则由 t 通过伸展 b 到 Tα 的表是对 Ψ 的记号表；若 $F\neg\neg\alpha$ 是 b 的一个结点，则由 t 通过伸展 b 到 Fα 的表是对 Ψ 的记号表。

(b)若 $T(\alpha\wedge\beta)$是 b 的一个结点，则由 t 通过伸展 b 到 Tα 与 Tβ 的表是对 Ψ 的记号表；若 $F\neg(\alpha\wedge\beta)$是 b 的一个结点，则由 t 通过伸展 b 到 $F\neg\alpha$ 与 $F\neg\beta$ 的表是对 Ψ 的记号表。

(c)若 $T\neg(\alpha\wedge\beta)$是 b 的一个结点，则由 t 通过增加 $T\neg\alpha$ 与 $T\neg\beta$ 作为 b 的两个子结点的表是对 Ψ 的记号表；若 $F(\alpha\wedge\beta)$是 b 的一个结点，则由 t 通过增加 Fα 与 Fβ 作为 b 的两个子结点的表是对 Ψ

的记号表。

(d)若 $T(\alpha \vee \beta)$ 是 b 的一个结点，则由 t 通过增加 $T\alpha$ 与 $T\beta$ 作为 b 的两个子结点的表是对 Ψ 的记号表；若 $F\rightarrow(\alpha \vee \beta)$ 是 b 的一个结点，则由 t 通过增加 $F\rightarrow\alpha$ 与 $F\rightarrow\beta$ 作为 b 的两个子结点的表是对 Ψ 的记号表。

(e)若 $T\rightarrow(\alpha \vee \beta)$ 是 b 的一个结点，则由 t 通过伸展 b 到 $T\rightarrow\alpha$ 与 $T\rightarrow\beta$ 的表是对 Ψ 的记号表；若 $F(\alpha \vee \beta)$ 是 b 的一个结点，则由 t 通过伸展 b 到 $F\alpha$ 与 $F\beta$ 的表是对 Ψ 的记号表。

我们称一个公式是标记的，如果至少有一条规则可以施加其上。否则就称为未标记的。

定义6 一个分支 b 是完全的，当且仅当 b 中未标记的记号公式都是首字(形如 Tp，$T\rightarrow p$，Fp，$F\rightarrow p$ 的记号公式，其中 p 为原子公式)。亦即，能用来扩张 b 的每一条规则都至少施加一次。一个对 Ψ 的记号表是完全的，当且仅当 t 的每一个分支都是完全的。

定义7 一个记号表 t 的一个分支被称为封闭的，若它满足如下条件之一：①对某一公式 A，$TA\in b$ 且 $FA\in b$；②对某一公式 A，$FA\in b$ 且 $F\rightarrow A\in b$。

下面给出 LPCm 意义下表演算可略分支的定义。

假设 S 是一个变元个例化后的句子集，Q 为 S 中出现的所有谓词组成的集合，受限谓词集 $P\subseteq Q$，T 为 S 按悖论逻辑记号表构造规则生成的完全记号表。

令 t 为一个 T 的任一分支，我们记为 $\Pi(t)=\{Tp(m)\mid p(m)$是一个原子命题，$Tp(m)\in t$ 且 $T\rightarrow p(m)\in t\}$；$AB(t)=\{Tp(m)\mid p\in P, Tp(m)\in t\}$，m 为个体常量。下面定义 LPCm 意义下可略分支的概念。

定义8 一个分支 b 是可略的，当且仅当 b 是完全的，且存在另一个分支 b′ 使得 $\Pi(b')\subset\Pi(b)$；或者 $\Pi(b')=\Pi(b')$，但 $AB(b')\subset AB(b)$。

对 Ψ 的记号表 t 的极小(记号)表 Tm 是由 t 通过删除 t 的每一个可略分支获得的。

定义9 一个对 Ψ 的记号表 t 是极小封闭的，若 t 的每个不可略分支都是封闭的。

下面给出对于 LPCm 语义的极小表演算的可靠与完全性定理。

定理1 令 S 是公式集，A 为公式，$CIRC_{LPCm}(S, P)\models_{LPCm}A$ 当且仅当对 TS 和 FA 的记号表是极小封闭的。

3.2 FTRS 实现方法描述

在 FTRS 系统中，信念集分为规则集和事实集两部分。接受待证命题后，规则集进行自由变量个例化，使得规则集中的自由变元被验证命题的常量(或与此常量相关的那些常量)替代，规则集与事实集个例化后合并生成前提集。前提集与待证命题(加"F"标志)合并生成操作集 S_1。操作集是一个符合合取范式规范的命题集，极小不可略分支的搜索是在操作集 S_1 中进行的。

令 S_1 是一个操作集，$S_1=\{P_1, P_2, \cdots, P_n\}$，$P_i=\{A_{i1}, A_{i2}, \cdots, A_{ij}\}$ 为原子句子(命题)，P_i 内部的原子句子间是"或"的关系，P_i 之间是"与"的关系。

按照 LP 关于记号表的定义，由每一个 P_i 抽取一个元素组成的集合就是一条表演算记号表的分支。操作集 S_1 含有的分支数目为 $\prod_{i=1,n}L_i$，L_i 为 P_i 的长度，即 P_i 包含元素的个数。按 LPCm 表演算的定义，分支之间存在大小关系，相对于小分支而言，大的分支即为可略分支。实现过程是对操作集 S_1 的分支空间遍历(大小为 $\prod_{i=1,n}L_i$)，抽取分支。对于当前抽取的分支，判断其是否为极小不可略分支，仍要再次搜索操作集的分支空间，直至搜索到比当前分支更小的分支，表明当前分支是可略分支；或整个操作集分支空间遍历一遍未找到更小的分支，从而判断出当前分支是极小不可略分支。如果所有的极小不可略分支都是封闭的，则待证命题成立。

4 FTRS 的实现机制

4.1 FTRS 系统的数据结构

4.1.1 永久型数据的数据结构：知识库由两部分组成：事实知识库和规则知识库。事实知识库与规则知识库以文件的形式存储。

事实知识库中的每条知识，采用二重表(LIST*)的形式表示。表(LIST)为 PROLOG 的基本数据结构。例如，对于事实：Tweety 是红色或绿色的，在逻辑中表示为 red(Tweety) ∨ green(Tweety)，在事实库文件中该条事实表示为[[red, Tweety], [green, Tweety]]。表[red, Tweety]的表头表示谓词常量，表尾表示个体常量。整个事实知识库是一个三重表。

规则知识库也是采用表结构表示一条规则。每条规则都是以子句的形式建立的。例如：$\forall x$(penguin(x)→ →fly(x))，其等价的子句的形式为：{ →penguin(x), →fly(x)}。在我们的规则知识库中该条规则表示为[→penguin(x), →fly(x)]。

4.1.2 临时型数据的数据结构：操作集的数据结构是操作集由系统运行时临时生成，它由两部分结合而成。一部分为事实知识库中与待证命题的个体常量有关的那些事实；另一部分为规则知识库的全部规则经待证命题的个体常量例化后生成的子句集。

例如：事实知识库为：bird (Tweety)，fly (Oscar)；规则知识库为：$\forall x \rightarrow bird(x) \vee fly(x)$；待证命题为 fly(Tweety)；系统进行推理操作时生成的操作集为：1)表演算方法：S＝[[bird (Tweety)], [→bird(Tweety),fly(Tweety)],[Ffly(Tweety)]]；2)模型赋值方法：S＝[[bird (Tweety)], [→bird (Tweety),fly(Tweety)]]。

原子句子集是由知识库中出现的所有的含不同谓词常量的原子句子组成的集合，它是系统运行时通过扫描一遍知识库将所有的不同原子句子抽取出来。原子句子集采用一重表的结构。例如，操作集 S ＝ [[bird (Tweety)], [→bird (Tweety), fly (Tweety)], [→fly(Tweety)]]，则原子句子集 P＝[bird(Tweety),fly(Tweety)]。

谓词常量、个体常量采用字符串的结构。

4.1.3 一个约定的谓词常量名：对于 abnormal 类特殊的谓词常量，系统对于其命名有一个特殊的约定：前缀字符串“ab_”。之所以这么做，是为了便于系统识别，这是进行限制操作所必须的。例如，“特殊的鸟”，表示成谓词常量，可写成“ab_bird”。

4.2 FTRS 系统的主要功能模块

主要功能模块包括：(1)初始化模块；(2)显示事实与规则库模块；(3)清空事实或规则库模块；(4)删除事实或规则模块；(5)添加事实或规则模块；(6)容错推理模块。

下面给出表演算方法的几个主要函数：

⊙判断操作集 X 存在不可略且不封闭的分支的函数⊙

```
exist_undel_unclose_minibranch_in_LPCm(X):-
/*若操作集 X 存在不可略且不封闭的分支，返回 TRUE。*/
    branch(X1,X),/*从操作集 X 抽取分支 X1*/
    not(exist_smaller_branch(X1,X)),
    /*对于已抽取的分支 X1，X 中不存在比 X1更小的分支*/
    getcouplef(X1,Y),
    /*对于分支 X1，抽取封闭对(型如 P,FP)生成 Y*/
    Y=[],
    /*Y 为空，表明 X1不封闭。以下给出待证命题不成立的结论*/
    write ("Conclusion:this Proposition can' t be deduded
        in LPCm!"),nl,
    write("Because there exist a unclose_mini branch:"),
        nl,
    write(X1),nl,
    write("(Press any key to return Main Menu)"),
    readchar(_).
```

⊙判断存在更小分支的函数⊙

```
exist_smaller_branch(X,Y):-
/*若操作集 Y 中存在比分支 X 更小的分支(即分支 X 可略)，返回 true。*/
    branch(Y1,Y),/*从操作集 Y 任意抽取分支 Y1*/
    getcouple(X,Xx),/*从分支 X 抽取矛盾对生成集合 Xx。*/
    getcouple(Y1,Yy1),
    /*从分支 Y1抽取矛盾对集合生成 Yy1。*/
    real_subset(Yy1,Xx).
    /*Yy1是 Xx 的真子集，通过比较矛盾对，发现存在比分支 X 更小的分支，表明分支 X 可略*/
exist_smaller_branch(X,Y):-
    branch(Y1,Y),
    getcouple(X,Xx),
    getcouple(Y1,Yy1),
    equallist(Xx,Yy1),
    /*分支 X 和 Y1包含的矛盾对相同。*/
    get_true_abpredicate(Y1,Y2),
    /*从分支 Y1抽取特例谓词(“ab_”标志)生成集合 Y2。*/
    get_true_abpredicate(X,X2),
    /*从分支 X 抽取特例谓词(“ab_”标志)生成集合 X2。*/
    real_subset(Y2,X2).
    /*Y2是 X2的真子集。在矛盾对相同的情况下，通过比较含有的特例谓词，发现存在比分支 X 更小的分支，表明分支 X 可略*/
```

判断待证命题满足操作集，只需在程序中加入如下语句：

not (exist_ undel_ unclose_ minibranch_ in_ LPCm(X))

这里 X 为操作集。

参考文献

[1] G. Priest, Logic of Paradox, J. of Philosophical Logic, (8)1979

[2] G. Priest, et al. (Ed.), Paraconsistent Logic: Essays in the Inconsistency, Philosophia Verlag, 1989

[3] G. Priest, Reasoning about Truth, Artificial Intelligence, (39)1989

[4] 林作铨，一个在弗协调逻辑中的限制，软件学报，1995年第6卷第5期

[5] 林作铨、李未，超协调逻辑研究，技术报告，STU-AL-TR-45, 1993

[6] 林作铨，超协调限制逻辑，计算机学报，1995年第18卷第9期

[7] 林作铨，悖论逻辑的表演算，软件学报，1996

[8] 林作铨，容错推理，计算机科学，1994