计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.240700139

不协调知识的表示和推理系统实现

朱福喜[1]　朱丽达[2]
1 武汉学院人工智能应用研究中心　武汉 430212
2 华中农业大学信息学院　武汉 430070
(ldzhu@mail.hzau.edu.cn)

摘　要　次协调逻辑作为一种非传统逻辑,能够合理地表示和处理不协调知识,但如何实现不协调知识的表示和推理,仍是一个亟待研究的课题。文中采用一种次协调逻辑系统——标记逻辑作为实现不协调知识的表示和推理的模型,以 Python 作为不协调知识的标记逻辑形式的表示工具,并在此基础上实现不协调知识下的推理。

关键词:不协调知识;次协调逻辑;标记逻辑;Python 语言;推理系统
中图分类号　G645

Representation and Reasoning System Realization of Inconsistent Knowledge

ZHU Fuxi[1] and ZHU Lida[2]
1 AI Applied Research Center,Wuhan College,Wuhan 430212,China
2 College of Informatics,Huazhong Agricultural University,Wuhan 430070,China

Abstract　As a type of non-traditional logic,paraconsistent logic is capable of representing and addressing inconsistent knowledge in a rational manner. However,the realization of representation and reasoning of inconsistent knowledge remains an urgent research topic. This paper utilizes a paraconsistent logic system-annotated logic-as a model for achieving the representation and reasoning of inconsistent knowledge. Specifically,Python is employed as the tool for representing the basis.

Keywords　Inconsistent knowledge,Paraconsistent logic,Annotated logic,Python language,Reasoning system

在知识发现系统中,由于知识的提取和表示受信息来源或时序性的影响以及处理方法的不得当,导致产生知识的不相容性。最简单的情形是,如果从知识源 s_1 得到某个断言 A,从知识源 s_2 得到某个断言 B,而从知识源 s_3 得到断言 A 和 B 不可能同时成立,这就立刻产生了不协调性,因为从这 3 个知识源得到了 $\{A,B,\sim A \vee \sim B\}$ 这样的不协调信息。例如,对于某种疫情,从某个知识源发掘到的知识认为这是由病毒 1 引起的,另一个知识源认为这是由病毒 2 引起的,而第三个知识源认为病毒 1 和病毒 2 不可能同时存在于某个人身上。如果这 3 种知识都被挖掘到一个知识库中,就产生了不协调性。

不协调知识还会以更复杂的形式出现,它的存在体现了现实世界的多样性和复杂性。如果知识库要反映这种多样性,那么就要允许一些有意义的矛盾存在于知识库中。这种思想有其广泛的社会背景和哲学动机,即绝大部分系统要能够容纳和处理有意义的矛盾。我国的“一国两制”就是这种哲学思想的典型范例。

综上所述,在知识发现和知识处理的过程中,处理不协调性是不可避免的,但经典逻辑面对这种不协调性却无能为力。因为在经典逻辑中,只要有矛盾存在就可以推出任何结论。这样,一个巨大而丰富的知识库就可能因为一个小小的不协调知识子集而崩溃。因此,研究不协调知识的表示、推理有着非常重要的意义。

本文的目标是构建容纳一个不协调知识的知识库系统,具体研究标记逻辑系统的表示和推理及实现,使之在知识处理过程中,出现矛盾信息后不再造成整个系统的崩溃,从而克服经典逻辑在推理系统中的脆弱性。这在专家系统、知识库系统、数据挖掘与知识发现、信息安全等领域里都有着重要的意义和作用。

1　国内外研究现状

不协调逻辑的研究一直受到人们的重视。巴西圣堡罗大学的逻辑学家 Da Costa 及其团队较早提出了一套形式的次协调演算 C 系统[1-3]。C 系统的技术途径是通过直接削弱经典逻辑演绎能力来达到次协调性;受 Da Costa 的思路启发,Kifer 等引入了标记逻辑(Annotated Logic)[4-5]。标记逻辑的基本思想是把公式的真值作为公式的标记。这样附有标记的公式用标记来表示不协调性。一般研究次协调逻辑的途径都是将经典逻辑转化为多值逻辑,标记逻辑就是一个多值逻辑,它通过公式的标记可以保留经典逻辑(二值逻辑)的框架,又可以扩展对不协调性的表达能力。Kifer 和 Lozinskii 把标记逻辑系统进一步展开为标记逻辑演算 APC(Annotated Logic Calculus)[4];Arieli 和 Avron 在研究了用双格(Bilattices)来表示次协调逻辑后,又给出将多值逻辑转换成二值逻辑的方法,并采用约束理论进行次协调推理[6];Arenas 等在使用标记逻辑对不协调数据库进行查询方面作了大量工作[7-8];Birnbaum 和 Lozinskii 则试图通过在不协调系统中寻找最大相容子集

基金项目:湖北省教育厅哲学社会科学研究项目(20G102)
This work was supported by the Hubei Provincial Department of Education Social Science Research Project(20G102).
通信作者:朱丽达(ldzhu@mail.hzau.edu.cn)

的方法来避免不协调性，这种做法虽然能够使整个不协调系统不至于废弃，但它不能容纳和利用有意义的矛盾[9]。多位学者研究了用多值逻辑来处理不协调性，他们提出的方法都是采用一个格(lattice)的元素作为逻辑值，并且一般都包含4个以上的逻辑值[10-13]。

在国内，哲学领域率先引进了次协调逻辑，并进行了广泛讨论和研究。例如，武汉大学的桂起权、中国社会科学院的张清宇、中山大学的鞠实儿根据不同见解，把 Paraconsistent logic 翻译成次协调逻辑、弗协调逻辑和超协调逻辑等[14-18]，并进行了一些扩展性研究[19-22]；在计算机科学领域，汕头大学的林作铨教授、北航的李未院士、中科院计算所的史忠植院士等也对不协调逻辑展开过深入的研究[23-25]。

从以上国内外研究动态来看，关于次协调逻辑的研究，哲学领域先于计算机科学领域，且次协调逻辑在社会科学方面的应用非常成功，而在计算机科学领域还没有找到处理不协调性的有效途径，但研究方向已逐步从注重研究内涵完整、协调和精确的问题转向研究内涵不完整、不协调和不精确的问题，本文的工作就是这一研究的有益尝试。

2 标记逻辑的推理

标记逻辑作为一个新的逻辑系统，旨在合理地处理不协调性，在矛盾中求协调。标记逻辑应用于知识发现领域，需确保在知识发现的过程中产生不协调性时仍可以正常地进行推理。要达到此目标，首先要采用一种恰当的次协调逻辑表示方法。

2.1 标记逻辑的表示

标记逻辑是一种较为直观的次协调逻辑形式系统。该逻辑引入一个标记集合 G。若 φ 是谓词逻辑中的公式，λ 是一个取值于标记集合 G 的标记，则形如 $\varphi:\lambda$ 的式子是一个标记公式。在标记逻辑中，标记 λ 表示在推理系统中原子命题的信任度、不确定度或可靠度等。

为了便于完成对标记的运算，限定 G 是一个完全格。例如，令 $G=FOUR=\{t,f,\top,\bot\}$ 是一个格，格中 t 和 f 分别表示真和假，$\top$ 表示“次协调”，$\bot$ 表示“未知”。如果定义格 $FOUR$ 上的序关系 $\leqslant$ 为：

1) $\forall x\in FOUR$，有 $x\leqslant x$(即格中的值都具有自反性)；

2) $\forall x\in FOUR$，有 $\bot\leqslant x$(即 $\bot$ 为 $FOUR$ 的下界)；

3) $\forall x\in FOUR$，有 $x\leqslant\top$(即 $\top$ 为 $FOUR$ 的上界)；

则在这个格中满足关系 $\bot\leqslant t\leqslant\top$，$\bot\leqslant f\leqslant\top$，但 $f\leqslant t$ 和 $t\leqslant f$ 都不满足，即 t 和 f 不具有可比性。格 FOUR 的结构如图1所示。

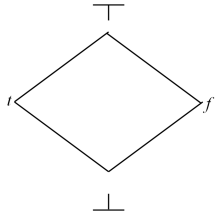

图1 格 FOUR

Fig. 1 Grid FOUR

标记值取值于格 $FOUR$ 就构成了一个简单的标记逻辑系统。直观上，$A:t$ 可理解为“A 为真”，$A:\top$ 为“A 为不协调”。

在标记逻辑中有两种否定，一种是强否定“$\neg$”，可以理解为具有本质性质的否定，$\neg\ p:\alpha$ 被解释为以程度 α 信任 p 的否定；另一种否定是弱否定“$\sim$”，可以理解为在认知层面上的否定。

定义1 在格 $FOUR$ 中，一个标记的否定“$\sim$”定义为：

$\sim(t)=f;\sim(f)=t;\sim(\bot)=\bot;\sim(\top)=\top$

不难看出，这个格是对称的，对所有的 $\alpha\in FOUR$，有 $\sim\sim\alpha=\alpha$。

定义2 在谓词逻辑中，蕴含 $\varphi\rightarrow\psi$ 定义为 $\psi\vee\neg\varphi$，而标记逻辑中有两种蕴含，即本质蕴含 $\psi\leftarrow\varphi\equiv\psi\vee\neg\varphi$ 和认知蕴含 $\psi<\sim\varphi\equiv\psi\vee\sim\varphi$。

两种蕴含各有不同的性质，尤其在不协调逻辑推理中差异很大。

定义3 标记逻辑公式的定义为：

1)如果 $A:\mu$ 是一个标记原子公式，则带否定词的 $\neg\ A:\mu$，$\sim A:\mu$ 也是一个公式；

2)如果 F_1 和 F_2 是公式，则通过几个基本的连接词连接而成的 $F_1\vee F_2$，$F_1\wedge F_2$，$F_1\leftarrow F_2$，$F_1<\sim F_2$ 也是公式；

3)如果 F 是公式，x 是任何变量符号，则带全称量词和存在量词的 $(\forall x)F$ 和 $(\exists x)F$ 也是公式。

更进一步推广，有如下变换公式：

4) $\sim p:\alpha\equiv\ p:\sim\alpha$

5) $\sim\neg\alpha\equiv\neg\sim\alpha$

6) $\sim(\varphi\vee\psi)\equiv\sim\varphi\wedge\sim\psi;\sim(\varphi\wedge\psi)\equiv\sim\varphi\vee\sim\psi$

7) $\sim(\forall x)\varphi\equiv(\exists x)\ \sim\varphi;\ \sim(\exists x)\varphi\equiv(\forall x)\sim\varphi$

2.2 标记逻辑的推理

考虑公式集 $S=\{p:t,p:f\vee q:t,p:f\vee q:f,r:t\}$。若要用谓词演算来表达与之相同的含义，则相应的公式集为 $S'=\{p,\neg\ p\vee q,\neg\ p\vee\neg q,r\}$。在传统的谓词演算中，$S'$ 是不协调的，按标准的谓词演算，它能推出任何结论，尤其是能推出 r 和 $\neg r$。从直观上看，r 与其他的3个公式对不协调性所起的作用是不同的，整个公式集的不协调与 r 无关。

标记逻辑引入表示不协调的逻辑值 $\top$，将传统逻辑中的某一部分“矛盾”搁置起来。具体来说，在标记逻辑中，对某个原子公式 p，如果有 $S\models p:\top$，则表示在知识集 S 中，至少有一个不协调的信息。下面研究如何实现这种形式推理。

类似于谓词逻辑的归结推理，标记逻辑推理也需要将标记公式转化为标记子句，这里标记子句的定义也类似于谓词逻辑的子句。

定义4 标记子句是标记逻辑的原子公式或原子公式的否定的析取式。

例如：$D_1(x):f\ \vee\neg(D_2(x):t)$ 是一个标记子句；$D_2(x):f\ \vee\sim(D_1(x):t)$ 也是一个标记子句。

2.2.1 将标记逻辑公式转化为标记子句的基本步骤

在进行谓词逻辑的归结推理之前，需要用9个步骤将谓词逻辑公式转化为标记子句。类似于这个过程，将标记逻辑公式转化为标记子句也需要完成如下基本步骤：

1)用 $\psi\vee\neg\varphi$ 取代 $\psi\leftarrow\varphi$，消去“$\leftarrow$”符号；用 $\psi\vee\sim\varphi$ 取代 $\psi<\sim\varphi$，消去“$<\sim$”符号。

2)降低¬和～号的辖域，直到它们在标记原子公式之前或消去¬和～号。

(1)降低强否定¬辖域的方法为：

用 $\neg\varphi\wedge\neg\psi$ 代替 $\neg(\varphi\vee\psi)$；用 $\neg\varphi\vee\neg\psi$ 代替 $\neg(\varphi\wedge\psi)$

用 $(\exists x)\neg\varphi$ 代替 $\neg(\forall x)\varphi$；用 $(\forall x)\neg\varphi$ 代替 $\neg(\exists x)\varphi$

用 α 代替 $\neg\neg\alpha$。

(2)依据定义3，降低弱否定～辖域的方法也是类似的，

具体操作为：

用$\neg\sim\alpha$代替$\sim\neg\alpha$；

用$p:\sim\alpha$代替$\sim p:\alpha$；

用$\sim\varphi\wedge\sim\psi$代替$\sim(\varphi\vee\psi)$；用$\sim\varphi\vee\sim\psi$代替$\sim(\varphi\wedge\psi)$；

用$(\exists x)\sim\varphi$代替$\sim(\forall x)\varphi$；用$(\forall x)\sim\varphi$代替$\sim(\exists x)\varphi$；

用α代替$\sim\sim\alpha$；$\neg\sim\alpha$代替$\sim\neg\alpha$。

3)运用定义1消去标记公式中标记部分弱否定符号“$\sim$”。

标记逻辑公式转化为标记子句的后续步骤与一阶谓词逻辑类似，其步骤为：将公式变为前束范式($Prefix$)；消去存在量词；消去全称量词；将公式变换为析取式的合取式；消去$\wedge$连词，使公式成为若干子句；将子句之间同名变量换名等。

2.2.2 标记逻辑的归结推理

如同谓词逻辑的归结，标记逻辑的推理也依赖合一算法和归结式。合一算法完成原子公式的变量代换，归结式和归结推理则要重新定义。

定义5(归结式，Resolvent) 设Q为标记子句$A_1:\mu_1\vee\cdots\vee\neg A_i:\mu_i\vee\cdots\vee A_k:\mu_k$，$C$为标记子句$B:\rho\vee B_1:\rho_1\vee\cdots\vee Br:\rho r$，其中$B$和$A_i$通过最一般合一元$\theta$可合一，且$\mu_i\leqslant\rho$，则$B$和$C$称为亲本子句或母子句，$Q$和$C$的归结式为：

$$(A_1:\mu_1\vee\cdots\vee A_{i-1}:\mu_{k-1}\vee B_1:\rho_1\vee\cdots\vee Br:\rho r\vee A_{i+1}:\mu_{k+1}\vee\cdots\vee A_k:\mu_k)\theta$$

其中，$\leqslant$为格的序关系。归结式中分别消去了母子句中的正文字和负文字，这是归结推导中得到空子句的必经途径。

定义6(推导，Deduction) 推导是一个从初始子句Q_0和子句集中子句C_1出发得到的一个系列：$\langle Q_0,C_1,\theta_1\rangle\cdots\langle Q_i,C_{i+1},\theta_{i+1}\rangle\cdots$，其中$Q_{i+1}$是$Q_i$和$C_{i+1}$的归结式，$\theta_{i+1}$为最一般合一元MGU(Most General Unifier)。

定义7(归结证明，refutation) 归结证明是一个从初始子句Q_0开始、长度为n的推导$\langle Q_0,C_1,\theta_1\rangle,\cdots,\langle Q_n,C_{n+1},\theta_{n+1}\rangle$，其中$Q_n$是空子句。

标记逻辑的归结推理就是通过一个推导，得到空子句的过程。推理系统就是通过归结算法实现这个推导。

2.2.3 推理策略

在实际归结过程中，还需要加入一些推理策略，以简化推理过程或使推理结果更清晰。

1)化简策略

子句对$p(u):s\vee\varphi$和$p(u'):r\vee\psi$的化简式为$p(u)\theta:lub(s,r)\vee\varphi\theta\vee\psi\theta$，其中$\theta=MGU(p(u),p(u'))$。

由前面的讨论可知，对于$\forall x\in FOUR$，有$\bot\leqslant x$(即$\bot$为$FOUR$的最大下界)。因此，若s,r之一为$\bot$，不妨设$r=\bot$，则$lub(s,r)=s$，简化式为$p(u)\theta:s\vee\varphi\theta\vee\psi\theta$。

另外，对$\forall x\in FOUR$，有$x\leqslant\top$(即$\top$为$FOUR$的最小上界)。因此，若s,r之一为$\top$，则$lub(s,r)=\top$，简化式为$p(u)\theta:\top\vee\varphi\theta\vee\psi\theta$。

特别是若s,r为t,f，或为f,t，则$lub(s,r)=\top$，简化式为$p(u)\theta:\top\vee\varphi\theta\vee\psi\theta$。在比较极端的情况下，若子句集为$C=\{p(u):t,p(t'):f\}$，则$C$可简化为$\{p(u)\theta:\top\}$，即$C$是不协调的。

2)删除策略

删除是要删除对推理无用的部分。

(1)删除子句C中形如$\neg p(u):\bot$的文字

值得注意的是，这里只是删除文字，而不是删除子句。例如，若对子句$\neg P(u):\bot$，则产生空子句；若对$\neg P(u):\bot\vee C$，则产生子句C。

(2)删除重言式

对标记子句的重言式有下列两种判别方法：

①凡是含有标记$\bot$的正文字的子句均为重言式；

②形如$\neg C_1:s\vee C_2:p$，且满足$C_1=C_2$，$s\geqslant p$的子句为重言式。

若一个子句经上述两种判别方法之一判别为重言式，则可将整个子句删除。经过简化后的子句仍为原始子句集的逻辑结论。下面讨论如何用归结证明的方法，推出一个子句集的逻辑结论。

2.3 实现归结证明的基本步骤

与谓词演算类似，利用标记的归结证明的基本算法为：

1)将所有已知的知识和事实改写成标记子句。

2)将要证明的目标子句Q取非，即$\neg Q$改写成标记子句。

3)将1)和2)所得到的子句合并成子句集S，在得到空子句或不再产生新子句之前执行：

(1)用简化策略对子句集进行简化；

(2)从子句集中选一对母子句；

(3)将母子句归结成一个归结式；

(4)若归结式为非空子句，则将其加入子句集。

4)若归结式为空，则归结结束。

此算法有两个出口：一是有空子句产生，此时归结证明成功；二是没有新的归结式产生，此时归结过程终止，即归结证明失败，则表明该子句集推不出要证明的目标子句。

3 标记逻辑归结证明系统的实现

有了前面的归结证明步骤，使用Python语言将标记逻辑归结证明过程实现为一个系统，即标记逻辑的推理系统。利用该系统进行推理的过程如图2所示。

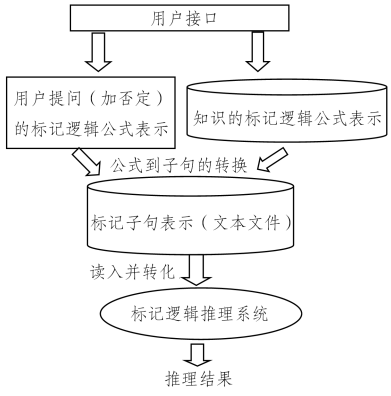

图2 标记逻辑推理系统的推理过程

Fig. 2 Inference process of annotated logic reasoning system

类似于谓词逻辑的归结原理推理系统，用户必须将背景知识或专家领域知识表达为标记逻辑的逻辑公式，再由这些公式按上述将公式转换为子句的方法进行相应的转换，并将这些子句的文本表示写入文件中，然后交给标记逻辑推理系统进行推理。

3.1 原子标记公式的表示

首先，对于形如$\varphi:\lambda$的原子标记公式，其组成部分为：[-] $predicate\_name(argument\_list):value$

这个原子标记公式实际有4部分：

[$sign$] 符号部分，为否定时用一个字符“-”表示

$predicate\_name$ 谓词部分，用一个字符串表示

argument_list 谓词的参数列表

value 标记部分，*value* 取值于格 *FOUR*

这 4 部分表示为 Python 语言中的 List。其中，参数表 *argument_list* 也要表示为 Python 的 List 数据结构，因为谓词的参数可能有多个，每个参数有可能是变量、常量或函数，并且函数也可以带有参数。显然，函数的参数用 List 表示可以嵌套地表示。这里约定 *argument_list* 中 3 种量的表示方法：小写字母表示变量；大写字母表示常量；函数则用一个 List 表示。

3.2 子句的表示

设标记逻辑的一个子句形式为：

$C=A_0:\lambda_0 \vee A_1:\lambda_1 \vee \cdots \vee A_n:\lambda_n$

有了原子标记公式的 Python 表示，一个子句仍可定义为 Python 语言的一个 List：

$[A_0:\lambda_0, A_1:\lambda_1, \cdots, A_n:\lambda_n]$

这样的表示约定了表中的项之间的关系是“∨”。

一个标记逻辑所表达的不协调知识库由若干子句 C_1，C_2，C_3，…，C_n 组成，该知识库可用 Python 的 List 定义为：

$[C_1, C_2, C_3, \cdots, C_n]$

显然，这个 List 就是一个不协调知识库，它的每个项是一个子句。

3.3 不协调知识库中的推理

从 Python 实现的角度看，一个不协调知识库至少嵌套三级的 List：最高层是知识库级 List，它的每一项都是一个中级的子句 List 表示，这个 List 包含的是标记公式 List，标记公式 List 则有可能包含多层嵌套的谓词变量 List。

有了不协调知识库的表示方法，在实现不协调知识库中的推理时，需要用 Python 语言实现归结证明的基本算法。实现该算法的核心步骤是在知识库中寻找一对母子句，判断两个子句是否构成一对母子句，则要用合一算法检验其是否匹配。这里的合一算法对传统的谓词逻辑合一算法进行了改写，其步骤是在遍历知识库的每个子句项（也是 List）时，挑选出两个子句，这两个子句谓词相同、符号相反，且带否定符号的谓词公式的标记值 μ 与另一公式的标记值 ρ 之间满足 $\mu \leqslant \rho$，即两个子句分别包含如下两个项：

$I_1:[-, P, [p\_argList], \mu]$

$I_2:[, Q, [q\_argList], \rho]$

满足：其中一项带有符号“-”，$P=Q$，$\mu \leqslant \rho$。

满足这 3 个条件后，检验两个子句项 List 的参数列表 *p_argList* 与 *q_argList*（仍是 List）是否匹配。寻找项与项之间合适的变量置换，使两个表达式一致的过程称为合一（Unify）过程。合一过程是为了找到一个最一般合一元的 σ，使得 $(p\_argList)\sigma=(q\_argList)\sigma$，即它们通过代换后是相等的。

用合一算法找到合一元后，即可定义 5 中的归结式。归结式作为一个新的子句加入子句集中，这就完成了归结证明的基本算法的 3.4 步。

从上文可以看出，实现标记逻辑归结系统的关键是 Python 语言完成了标记逻辑归结算法的第 3 步和第 4 步。

3.4 标记逻辑推理的应用

下面用一个实例展示使用标记逻辑在现实生活中对实际问题进行推理的过程。

实例 1：不协调的医学专家系统，首先考虑通过向两位医生 Dr_1 和 Dr_2 咨询来构造医学专家知识。

Dr_1 提供的知识为：

C_1：如果某病人表现有症状 S_1 为真，症状 S_2 为假，那么该病人患有疾病 D_1 为真或患有疾病 D_2 为真；

C_2：如果某病人患有疾病 D_2 为真，那么该病人患有疾病 D_1 为假；

C_3：如果某病人患有疾病 D_1 为真，那么该病人患有疾病 D_2 为假。

医生 Dr_2 提供的知识为：

C_4：如果病人检查否定了症状 S_2，但肯定了症状 S_3，则病人肯定患病 D_1；

C_5：如果病人检查否定了症状 S_3，则病人肯定不可能患病 D_2。

对病人 P_1 和 P_2 检查的结果为：

P_1 有症状 S_1 和 S_3，没有症状 S_2；P_2 有症状 S_1，而没有 S_2，对症状 S_3 未下任何结论。

现在将上述知识表达成标记逻辑子句的形式。

医生 Dr_1 提供的知识表示成子句形式为：

$C_1:D_1(x):t \vee D_2(x):t \vee \neg(S_1(x):t) \vee \neg(S_2(x):f)$

$C_2:D_1(x):f \vee \neg(D_2(x):t)$

$C_3:D_2(x):f \vee \neg(D_1(x):t)$

其中 x 代表某病人。

医生 Dr_2 提供的知识表示成子句形式为：

$C_4:D_1(x):t \vee \neg(S_2(x):f) \vee \neg(S_3(x):t)$

$C_5:D_2(x):f \vee \neg(S_3(x):f)$

将病人 P_1 和 P_2 检查的结果表示成子句形式为：

$C_6:S_1(P_1):t$

$C_7:S_1(P_2):t$

$C_8:S_2(P_1):f$

$C_9:S_2(P_2):f$

$C_{10}:S_3(P_1):t$

将 C_1-C_{10} 合并成子句集 C，若要从 C 推断出 P_1 患病 D_1，只需证明 $D_1(P_1):t$ 是 C 的逻辑结论。根据以上步骤，需对 $\{C_1, \cdots, C_{10}\} \cup \{\neg D_1(P_1):t\}$ 进行归结，得到空子句。

归结过程如下：

$E_1:\neg D_1(P_1):t$ （目标子句的否定）

$E_2:\neg(S_2(P_1):f) \vee \neg(S_3(P_1):t)$ （由 E_1 和 $C_4:P_1/x$）

$E_3:\neg(S_3(P_1):t)$ （由 E_2 和 C_8）

$E_4:\square$ （由 E_3 和 C_{10}）

类似地，推断 P_1 没有患病 D_2 的归结过程如下：

$F_1:\neg D_2(P_1):f$ （初始查询）

$F_2:\neg D_1(P_1):t$ （由 F_1 和 $C_3:P_1/x$）

$F_3:\neg(S_2(P_1):f) \vee \neg(S_3(P_1):t)$ （由 F_2 和 $C_4:P_1/x$）

$F_4:\neg(S_3(P_1):t)$ （由 F_3 和 C_8）

$F_5:\square$ （由 F_4 和 C_{10}）

以上推导所涉及的知识都是协调的，下面考虑不协调的情况。

假设第三位医生 Dr_3 提供了以下知识：

$C_{11}:D_2(x):t \vee \neg(S_3(x):t)$，即某人有症状 S_3，则他肯定患疾病 D_2。

将 3 个医生提供的规则和病理医生提供的“事实”全部合并，得到子句集 Q，这时 Q 就出现了不协调。不协调的原因在于：医生 Dr_2 认为病人有症状 S_3，则他肯定患疾病 D_1；医生 Dr_3 认为病人有症状 S_3，则他肯定患疾病 D_2；而医生 Dr_1 认为

一个人不可能同时患 D_1 和 D_2 两种疾病。但标记逻辑能够处理子句集 Q 的不协调性。

下面通过归结来推断出 P_1 既患疾病 D_1 又没有患疾病 D_1，即 $D_1(P_1):\top$ 是 Q 的逻辑结论。

$G_1:\neg D_1(P_1):\top$ （初始查询的否定）

$G_2:D_2(P_1):t$ （由 C_{10} 和 $C_{11}:P_1/x$）

$G_3:D_1(P_1):f$ （由 G_2 和 $C_2:P_1/x$）

$G_4:D_1(P_1):t\vee\neg(S_3(P_1):t)$ （由 C_4 和 $C_8:P_1/x$）

$G_5:D_1(P_1):t$ （由 G_4 和 $C_{10}:P_1/x$）

$G_6:D_1(P_1):\top$ （由 G_5 和 C_3）

$G_7:\square$ （由 G_1 和 G_6）

虽然这个知识集存在不协调性，可以推出 P_1 患疾病 D_1 的不协调结论，但这种不协调性的存在并不允许任意推出其他结论，例如，不能推断出 P_2 患疾病 D_1 或不患疾病 D_1，也不影响对其他疾病（如 D_3，D_4，D_5，…）的推断。

结束语 本文用一种次协调逻辑系统——标记逻辑作为不协调知识的表示和推理模型，并重点研究了它的推理系统的实现问题。首先，引出次协调性逻辑的实际背景；然后讨论了次协调逻辑的表示及推理问题，具体涉及标记逻辑公式的表示、归结证明及必要的推理策略；接着讨论标记逻辑推理的实现问题，即如何用当今流行的 Python 语言，实现一个可行的标记逻辑归结证明系统；最后展示了使用标记逻辑进行推理的应用实例。

后续工作可以考虑将该推理系统与知识发现系统融为一体，以便在知识挖掘系统出现矛盾时，能够有效合理地利用其他有用知识进行推理。

参 考 文 献

[1] DA COSTA N. Automated Theorem Proving in Paraconsistent Logic[J]. Journal of Automated Reasoning, 1992, 9(1): 170-210.

[2] DA COSTA N, RONDE D C. The Paraconsistent Logic of Quantum Superpositions[J]. Foundation of Physics, 2013, 43: 845-858.

[3] DA COSTA N C A, RONDE D C. Non-Reflexive Logical Foundation for Quantum Mechanics[J], Foundation of Physics, 2014, 44: 1369-1380.

[4] KIFER M, LOZINSKII E L. A Logic for Reasoning with Inconsistency[J]. Journal of Automated Reasoning, 1992, 9(2): 179-215.

[5] KIFER M, SUBRAHMANIAN V S. Theory of Generalized Annotated Logic Programming and its Applications[J]. Journal of Logic Programming, 1992, 12(4): 335-368.

[6] ARIELI O, AVRON A. Logical Bilattices and inconsistent data [C]// Proceedings 9th IEEE Annual Symp. On Logic in Computer Science, Piscataway, N. J: IEEE Press, 1994: 468-476.

[7] ARENAS M, BERTOSSI L, CHOMICKI J. Consistent Query Answers in Inconsistent Databases [C] // Proceedings ACM Symposium on Principles of Database Systems. Philadelphia: ACM PODS'99, 1999: 68-79.

[8] ARENAS M, BERTOSSI L, KIFER M. Application of Annotated Predicate Calculus to Querying Inconsistent Databases[C]// 6th International Conference on Rule and Objects in Databases (DOOD'2000). Berlin: Springer, 2000: 926-941.

[9] BIRNBAUM E, LOZINSKII E L. Consistent Subsets of Inconsistent Systems. [EB/OL]. [2002-06-09]. http://citeseer.nj.nec.com/birnbaum02consistent.html.

[10] MEHEUS J. Do We Need Paraconsistency in Commonsense Reasoning? Logic and Philosophy of Science[R]. Belgium: Ghent University, 2003.

[11] MARE DARC DENECKER A. Circumscriptive Approaches to Paraconsistent Reasoning. [EB/OL]. [2001-03-06]. http://www.cs.kuleuven.ac.be.

[12] BLAIR H. Paraconsistent Logic Programming [J]. Theoretical Computer Science, 1989(68): 340-360.

[13] GINSBERG M L. Multivalued Logics: A Uniform Approach to Reasoning in Artificial Intelligence [J]. Computational Intelligence, 1988(4): 265-316.

[14] GUI Q Q. What is Paraconsistent Logic [J]. Logic and Language Learning, 1988, 4: 20-23.

[15] GUI Q Q. Paraconsistent Formal Systems: The Logic of Seeking Coordination in Contradictions[J]. Journal of Wuhan University, 1989(6): 34-38.

[16] WANG WC, GUI QQ. Non-classical Logical Interpretations of Quantum Superposition State and Quantum Identity[J]. Journal of Dialectics of Nature, September 2017, 39(5): 38-44.

[17] GUI Q Q. On Complementary Logic, Dialectical Logic and Paraconsistent Logic[J]. Henan Social Sciences, March 2010, 18(2): 57-60.

[18] ZHANG Q Y. Paraconsistent Logic[M]. Beijing: China Social Sciences Press, 2003.

[19] JU S E, LIU H. A Three-valued Logic Based on the Presupposition of an Open World[C]// 863 Program Intelligent Computer Theme Academic Conference. 2001: 507-514.

[20] ZHANG G Q. An Introduction to David Bohm's Philosophy of Nature[M]// Jointly Published by the Management Committee of China Development Foundation in Taiwan and Hongye Culture Enterprise Co., Ltd., 2002: 85-116.

[21] GUI Q Q, CHEN Z L, ZHU F X. Paraconsistent Logic and Artificial Intelligence (Academic Series of Wuhan University) [M]// Wuhan: Wuhan University Press, 2002: 176-180.

[22] REN X M, GUI Q Q. A Phylogenesis Study of Non-Classical Logic Systems: Also on the Central Issue of the Philosophy of Logic[M]// Tianjin: Nankai University Press, 2011: 190-198.

[23] LIN Z Q, LI W. Inconsistent Logic [J]. Computer Science, 1994 (5): 190-198.

[24] CHEN Z L, SHI Z Z. New Axiom RF of Paraconsistent Logic [C]// Proceedings of the Artificial Intelligence Academic Conference. 1993: 147-151.

[25] ZHU F X, GONG C S, YU Z K. Paraconsistent Reasoning Based on XML[J]. Journal of Wuhan University, 2006, 52(1): 1023-1028.

ZHU Fuxi, born in 1957, Ph.D, professor, Ph.D supervisor. His main research interests include artificial intelligence and so on.

ZHU Lida, born in 1984, Ph. D, is a member of CCF (No. C0784M). Her main research interest includes artificial intellige.