有限信念集上修正的一种方法*

栾尚敏+, 戴国忠

(中国科学院 软件研究所,北京 100080)

An Approach to the Revision of a Finite Belief Set

LUAN Shang-Min+, DAI Guo-Zhong

(Institute of Software, The Chinese Academy of Sciences, Beijing 100080, China)

+ Corresponding author: Phn: 86-10-62540434, E-mail: shangmin_luan@sina.com

http://iel.iscas.ac.cn/

Received 2002-04-22; Accepted 2002-05-23

Luan SM, Dai GZ. An approach to the revision of a finite belief set. *Journal of Software*, 2003,14(5):911~917.
http://www.jos.org.cn/1000-9825/14/911.htm

Abstract: In this paper, an approach to the revision of a finite belief set is presented. First, a procedure for generating all the minimal inconsistent sets is introduced, and the correctness of the procedure is proved. Then what discussed further is that how to apply the procedure to the implementation of some representative methods, and a implemented prototype for belief revision is introduced. At last, the presented approach is compared with other related work.

Key words: belief revision; propositional logic; first-order logic; finite belief set

摘 要: 讨论了信念集是有限子句集时的信念修正方法.首先给出了一阶逻辑上求所有极小不协调子集的一个过程,证明了该过程的正确性;然后讨论了由有极小不协调的子集来实现信念修正的方法,介绍所开发的信念修正的原型系统;最后与相关工作进行了比较.

关键词: 信念修正;命题逻辑;一阶逻辑;有限信息集

中图法分类号: TP18 **文献标识码**: A

信念修正(belief revision)是当今人工智能领域中的研究热点之一,已取得了很多研究成果.真值维护系统[1]是第 1 个与信念修正有关的系统.Kleer[2]分析了该系统效率低的原因,给出了基于假设的真值维护系统.基于公式的方法[3,4]讨论了信念集是逻辑语句集时的信念修正过程.基于模型的方法[5~10]讨论了信念集为命题逻辑语句集时的修正方法.“理论修改逻辑”的方法[11]将信念集定义为一个理论闭包,给出了修正和约简的合理公设以及满足这些公设的修正方法,这些方法有:极大协调子集修正[11]、部分交集修正[11]、全部交集修正[11]、安全修正[12]、Systems of spheres[12]、基于认识牢固度的修正[13]等.基于模型的修正方法也满足部分公设[14].以信念集限制为理论基,人们[15~20]给出了与“理论修改逻辑”类似的结果.迭代的方法[21]研究信念集是形式理论的情况时

* Supported by the National Natural Science Foundation of China under Grant Nos.60033020, 60103020 (国家自然科学基金); the China Postdoctoral Science Foundation (中国博士后科学基金)

第一作者简介: 栾尚敏(1968－),男,山东莱芜人,博士,主要研究领域为人机交互技术,算法设计自动化,信念修正,形式化方法.

进行多次修正的性质和方法,并给出了可以归约出信念集的极大选择修正的演译方法,称为 R-演算[22].另外,还有基于顺序条件函数的修正[23]、自然修正[24]等方法.人们对信念修正在计算机上的可实现性也进行了研究.Dixon[25,26]对有限理论基上的基于认识牢固度修正[20]的可实现性进行了研究,并进一步讨论了一阶逻辑上修正的实现方法[27],给出了修正的原型系统,Williams[28]采用的策略是,在没有必要对一个信念的牢固度进行修改之前,该信念在系统中的牢固度保持不变,文献[29]讨论了扩充逻辑程序设计中的知识库修正问题.对于信念修正的研究,不仅具有理论意义,也有着广泛的应用前景,例如可以应用到基于知识的用户界面设计中[30].

当信念 A 加入到信念集Γ中时,若 A 和Γ不协调,则上述很多方法都用到极大协调子集以得到修正后的信念集,但都没有讨论如何得到Γ的和 A 极大协调的子集.本文将信念集限制为有限的子句集,信念限制为子句,给出一种有限信念集上可实现的方法.首先给出了一个求所有极小不协调子集的过程,然后讨论了如何由该过程来实现某些典型的信念修正方法,并介绍了我们用 C++语言开发的用于信念修正的原型系统,同时与相关工作进行了比较.关于数理逻辑方面的知识和记号,请见文献[31].

1 求所有极小不协调子集的方法

对一般情况给出求所有极小不协调子集的方法是比较困难的,但一阶逻辑语句可化为子句形式.本节的讨论只针对子句的情况.关于一般一阶逻辑语句化为子句的过程,请见文献[31].对于子句 A 和协调的子句集Γ,如何求得Γ的与 A 不协调的极小子集,这是本节讨论的内容.首先证明几个引理.

引理 1.1. 设Γ是子句集,$A=B_1\vee B_2\vee\ldots\vee B_n$ 为一子句,B_i 是文字,Ψ_i 是Γ的所有和 B_i 极小不协调的子句集的集合(i=1,2,…,n).令 $\Psi=\{\Gamma_1\cup\Gamma_2\cup\ldots\cup\Gamma_n|\Gamma_i\in\Psi_i,i=1,2,\ldots,n\}$,则 Ψ包含Γ的所有和 A 极小不协调的子句集的集合.

证明:对于任意$\Gamma'=\Gamma_1\cup\Gamma_2\cup\ldots\cup\Gamma_n\in\Psi$,因为$\Gamma_i\vdash\neg B_i$,所以$\Gamma'\vdash\neg B_1\wedge\neg B_2\wedge\ldots\wedge\neg B_n$,即$\Gamma'\vdash\neg(B_1\vee B_2\vee\ldots\vee B_n)$,也就是$\Gamma'\vdash\neg A$,即 A 和Γ'是不协调的.所以 A 和 Ψ中任意子句集都是不协调的.

其次,设 Γ''是 Γ 的和 A 极小不协调的子集,则 $\Gamma''\vdash\neg A$.由$\neg A=\neg B_1\wedge\neg B_2\wedge\ldots\wedge\neg B_n$,得到 $\Gamma''\vdash\neg B_1\wedge\neg B_2\wedge\ldots\wedge\neg B_n$,所以 $\Gamma''\vdash\neg B_i(i=1,2,\ldots,n)$,因此 B_i 和 Γ''也是不协调的,所以有 B_i 和 Γ''的一个子集 Γ_i 形成极小不协调的子句集,并且$\bigcup_{i=1}^{n}\Gamma_i$和 A 是不协调的,由 Γ''和 A 的极小不协调性可知$\bigcup_{i=1}^{n}\Gamma_i=\Gamma''$,即有 $\Gamma''\in\Psi$.所以,Ψ中包含了Γ的所有和 A 极小不协调的子句集. □

Ψ中可能有些子句集和 A 是不协调的,但不是极小的.若 Ψ中某个元素不是极小的,则 Ψ中存在 Γ''为该子句集的真子集,把这样的子句集从 Ψ中删除,Ψ就是 Γ 的所有和 A 极小不协调的子句集的集合.对于 $A=B_1\vee B_2\vee\ldots\vee B_n$,只要得到了 Γ的所有与 B_i 矛盾的子句集,可以得到 Γ的所有与 A 矛盾的子句集.下面我们将讨论如何求得 Γ 的所有和 A 极小不协调的子句集.首先证明下面的引理.

引理 1.2. 对于子句集 Γ和文字 B,令$\Delta=\{C|C$ 为 B 和 Γ中某个子句的归结式$\}$,$\Gamma'=\{C|C\in\Gamma$,且 B 不蕴涵 C,C 也不能和 B 执行归结操作$\}$.对于每一个 $C_i\in\Delta$,设 Ψ_i 是 $\Gamma'\cup\Delta-\{C_i\}$的所有和 C_i 极小不协调的子句集的集合(i=1,2,…,m),令 $\Psi'=\bigcup_{i=1}^{m}\Psi_i$ 和 $\Psi=\Psi\{\Gamma''-\Delta\cup\{D|C\in\Gamma''-\Delta$是 D 和 B 的归结式$\}|\Gamma''\in\Psi\}$,则 Ψ是 Γ的所有和 B 极小不协调子句集的集合.

证明:对于任意的$\Gamma''\in\Psi_i$,因为 C_i和Γ''是极小不协调的,所以存在一个推导出矛盾语句的归结过程.对于该推导过程中用到的Δ中的子句,因为它是 B 和 Γ中某个子句的归结式,所以增加推导出这些归结式的步骤就得到一个由 B 和 Γ 中的语句开始的推导出矛盾语句的过程,并且这个推导过程用到的语句就是集合 $\Gamma_1=\Gamma''-\Delta\cup\{D|C\in\Gamma''-\Delta$是 D 和 B 的归结式$\}$中的子句和 B.所以 Γ_1和 B 是不协调的.假设 Γ_1和 B 不是极小的,设 Γ_1中和 B 极小不协调的子句集为 Γ',则存在由 Γ'和 C_i 推导出矛盾语句的归结过程,所以如果将 Γ'中能和 B 进行归结的语句换成 B 和这些语句的归结式,新得到子句集 Γ_k,则由 Γ_k 也能推导出矛盾语句,所以 Γ_k 是不协调的,并且 Γ_k 是 Γ''的真子集,也就是 C_i 和 Γ''不是极小不协调的,这就产生了矛盾,所以 Γ_1和 B 是极小不协调的.

其次证明包含了 Γ的所有和 B 极小不协调子句集.对于 Γ的某个和 B 极小不协调子句集 Γ',Δ为 B 和 Γ'中子句的归结式集合,则集合 $\Gamma''=\Delta\cup\{D|D\in\Gamma'$,并且 D 和 B 不能执行归结操作$\}$是极小不协调的.所以,对于任意 $C\in\Delta$,有 $\Gamma''-\{C\}$属于某个 Ψ_i,并且 $\Gamma'=\Gamma''-\Delta\cup\{D|B\in\Gamma''-\Delta$是 D 和 C 的归结式$\}\in\Psi$.所以,Γ的任意一个和 B 极小不协调子句集都属于 Γ.结论得证. □

设 Γ 是一个子句集,$A=B_1\vee B_2\vee\ldots\vee B_n$ 是一个子句,根据引理 1.1 和引理 1.2,根据下面的过程得到的结果就可以构造出所有极小不协调子集.

第 1 步.对每个文字 B_i,求 B_i 和 Γ 中能执行归结操作的子句的归结式,得到归结式的集合为 Δ,令 $\Gamma'=\{C|C\in\Gamma$,且 B_i 不蕴涵 C,C 也不能和 B_i 执行归结操作$\}$.

第 2 步.如果 Δ 非空,则对于 Δ 中的每个非空子句 C,令 $A=C$,$\Gamma=\Gamma'\cup\Delta-\{C\}$.对于新得到的 A 和 Γ,执行第 1 步.

根据归结原理,如果每一次选择的 C 都处于 $\Gamma'\cup\Delta$ 的一个极小不协调的子集中,则最终会推导出矛盾语句.所以有下面的引理:

引理 1.3. 设 Γ 是一个子句集,$A=B_1\vee B_2\vee\ldots\vee B_n$ 为一个子句,且 Γ 至少有一个子集和 A 是极小不协调的,若每次选择的归结式都处于 $\Gamma'\cup\Delta$ 的一个极小不协调的子集中,则最终推导出矛盾语句.

证明:若 Γ 有一个子集和 A 是极小不协调的,则 $\Gamma\vdash\neg A$,也就得到,对于任意 B_i,$\Gamma\vdash\neg B_i$.所以 Γ 至少有一个子集和 B_i 是极小不协调的.第 1 步执行了多步归结的过程,所以归结式集中至少有一个归结处于 $\Gamma'\cup\Delta$ 的一个极小不协调子集中.根据归结原理,若每次选择的归结式都处于 $\Gamma'\cup\Delta$ 的一个极小不协调的子集中,则最终推导出矛盾语句. □

对于得到矛盾语句的一步推导,可以构造出极小不协调的子集,这样逐步向后推,最终能构造出所有极小不协调的子集来.若从 A 中的每一个文字开始都能推导出矛盾语句,则是否意味着 A 处于 Γ 的一个极小不协调子集中呢?这是肯定的,因为由上述推导出矛盾的过程构造出一个归结出矛盾语句的过程.但对于一阶逻辑的情况,谓词中存在变量,需要对变量进行替换.若文字 L 和子句 B 进行归结运算,且 B 中能 L 执行归结运算的文字为 $\neg L$.对于变量替换需要考虑如下两种情况: (1) 若 L 中不包含变量,则可以直接进行归结操作.(2) 若文字 L 中为变量,则需要记录 L 中的变量被替换成了什么值,以便在后面的过程中将 L 所在的子句中的其他包含该变量的文字的变量进行替换.如果不对变量进行变量替换,则会得到错误的结论.用下面例子进行说明.

例 1.1:设子句集 $\Gamma=\{\neg P_1(a),\neg P_2(b)\}$,若这时再加入子句 $F=P_1(x)\vee P_2(x)$.显然 $\Gamma\cup\{F\}$ 是协调的.但关于 Γ 和 F 执行上述过程,则从 F 中的任何一个文字都能推出矛盾语句 $\perp$.对于 $P_1(x)\vee P_2(x)$ 中的文字 $P_1(x)$ 和 $P_2(x)$ 分别执行归结规则,均得到了矛盾序语句 $\perp$.但在用 $P_1(x)$ 执行归结操作时,其替换是 x/a,而在用 $P_2(x)$ 作为归结操作时,用到的替换是 x/b.若首先处理 $P_1(x)$,则在处理 $P_2(x)$ 时就需要考虑 $P_1(x)$ 中变元 x 被替换为何值;若首先处理 $P_2(x)$,则在处理 $P_1(x)$ 时就需要考虑 $P_2(x)$ 中变元 x 被替换为何值,所以先处理文字得到的替换去置换后处理文字的变量,否则就会得到错误的结论.对于该例,可以如下处理:若首先处理 $P_1(x)$ 得到了替换 a/x,则在处理 $P_2(x)$ 时,将该文字中的变量 x 用 a 替换,得到文字 $P_2(a)$,由 $P_2(a)$ 开始推导不出矛盾语句 $\perp$;或者首先处理 $P_2(x)$ 得到替换 b/x,再处理 $P_1(x)$ 时,将 x 用 b 替换,得到文字 $P_1(b)$,由 $P_1(b)$ 开始推导不出矛盾语句 $\perp$.这就符合由数理逻辑推出的结论.对于一个变量,最终被替换成什么样的值,只能在推出矛盾语句时才能确定.用下面的例子进一步说明.

例 1.2:设 $\Gamma=\{\neg P_1(y)\vee P_2(y),\neg P_2(a),P_3(a)\}$,若再向 Γ 中加入子句 $P_1(x)\vee P_3(x)$,设首先用 $P_1(x)$ 和 Γ 中的子句执行归结操作,得到的替换为 y/x,归结式为 $P_2(y)$,再用 $P_2(y)$ 和 Γ 中的子句 $\neg P_2(a)$ 进行归结,得到的替换为 a/y,并且此时得到了矛盾语句,在这时才能确定 x 最终被替换成了 a;检查由 $P_3(x)$ 开始能否推导出矛盾语句 $\perp$ 时,对变量 $P_3(x)$ 中的变量 x 替换为 a,而不是 y.在推导出矛盾语句 $\perp$ 的过程中,变量进行了多次替换,如何得到一个变量的最终替换值呢?在这个例子中,就是如何由 x 替换为变量 y,而 y 又被替换为 a,得到 x 被替换为 a.对于替换 y/x,因为 a/y,可以将 y/x 中的 y 替换为 a,我们把这个过程称为替换的合成,它和归结原理中替换的合成稍有不同,在给出求解的过程时再给出其定义.

由一个谓词可以得到多个不同的文字,它们可以同时出现在一个子句中,并且还可能会有多个文字能和 A 进行归结运算,这时需要对这个子句连续执行多次归结操作,并将每次的变量替换记录下来,以便于后面的检查.在执行归结的过程中,对于每一种替换都能推出矛盾语句 $\perp$ 才能说明它是不协调的,若从某个替换不能推出矛盾语句,则是协调的.我们来看下面的例子.

例 1.3:设 $\Gamma=\{\neg P_1(a)\vee\neg P_1(b),\neg P_2(y)\}$,若再向 Γ 中加入子句 $P_1(x)\vee P_2(x)$,则首先处理 $P_1(x)$,得到归结式分别为 $\neg P_1(b)$ 和 $\neg P_1(a)$,替换分别是 a/x 和 b/x.用 $\neg P_1(b)$ 和 $P_1(x)$ 执行归结规则,得到矛盾语句,此时 b/x.也就是说,需要将 $P_2(x)$ 中的变量替换为 a,检查从 $P_2(a)$ 开始能否推出矛盾语句 $\perp$,再将 $P_2(x)$ 中的变量替换为 b,检查从 $P_2(b)$ 开

始能否推出矛盾语句⊥,若这两种情况都能推出矛盾,则是不协调的,否则就是协调的.由 $P_2(a)$和 $P_2(b)$均能推导出矛盾语句⊥,所以 Γ和 $P_1(x)\vee P_2(x)$是不协调的.

对于一阶逻辑,只要正确地将变量替换,就能得到一个正确的结果,所以有如下的引理:

引理 1.4. 设 Γ是一个子句集,A 为一个子句,若从 A 中的每一个文字开始都能推导出矛盾语句,并且对同一个子句文字中的变量进行了如前所述的替换,则 A 处于 Γ的一个极小不协调子集中.

证明:根据由递归执行上述两步推导出矛盾语句的过程,可以构造如下一个归结得到矛盾语句的过程:在第 1 步处理子句中的文字时,将该子句中还没有处理的文字和Δ中的每一个子句形成一个新子句,后面再处理其他文字时,就用这个文字所在的子句中的其他文字和归结式形成一个新子句,并将变量正确替换,这样,由上述推导出矛盾语句的过程就得到了一个归结出矛盾语句的过程,结论成立. □

在给出详细的求解过程之前,首先给出替换合成的定义.

$$\sigma=\{t_1/x_1,t_2/x_2,\ldots,t_n/x_n\}.$$

$$\sigma\circ\sigma'=\{\sigma'(t_1)/x_1,\sigma'(t_2)/x_2,\ldots,\sigma'(t_n)/x_n\};$$

$$\sigma'\circ(\sigma_1|\sigma_2|\ldots|\sigma_n)=(\sigma'\circ\sigma_1)|(\sigma'\circ\sigma_2)|\ldots|(\sigma'\circ\sigma_n);$$

$$(\sigma_1'|\sigma_2'|\ldots|\sigma_n')\circ(\sigma_1|\sigma_2|\ldots|\sigma_n)=(\sigma_1'\circ\sigma_1)|(\sigma_1'\circ\sigma_2)|\ldots|(\sigma_1'\circ\sigma_n)|(\sigma_2'\circ\sigma_1)|(\sigma_2'\circ\sigma_2)|\ldots$$
$$|(\sigma_2'\circ\sigma_n)|\ldots|(\sigma_n'\circ\sigma_1)|(\sigma_n'\circ\sigma_2)|\ldots|(\sigma_n'\circ\sigma_n).$$

根据上面的讨论,得到一阶逻辑上的求所有极小不协调子集的过程.过程 Incon-subset(A,Γ)求得了所有极小不协调的子集.下述过程中用到的$\sigma|\mathrm{Var}(L)$表示替换$\{t/x|t/x\in\sigma$,且$x\in\mathrm{Var}(L)\}$.InRes(L,Γ)关于文字 L 和子句集 Γ具体执行一次归结操作的过程.

```
Incon-subset(A,Γ)
  Φ = {⟨B,B⟩|B∈Γ};
  Ω = InResC(∅,A,A,Φ);
  Ψ1 =∪⟨σ,Ψ'⟩∈Ω Ψ'
  Ψ ={Γ'-{A}|Γ'∈Ψ1};
  For (each Γ1∈Ψ ) do
    If (存在 Γ2∈Ψ,使得 Γ2⊂Γ1) then
        Ψ =Ψ-{Γ1};
  Return Ψ ;
End (Incon).
```

```
InRes (L,Φ)
  Φ1=∅; γ=∅;
  while (Φ 非空) do
    Begin
      ⟨A,C⟩= delete(Φ);
      If (A 和 L 可以执行归结操作) then
        Begin
          γ= γ∪ {⟨σ,B,A⟩|B 是 A 和 L 的归结式,σ是最一般的替换或替换集};
          If (A 包含σ中的变量) then
            Φ1 = Φ1 ∪ {⟨A,C⟩};
        End.
      else if (L 不蕴涵 A) then Φ1 = Φ1 ∪ {⟨A,C⟩};
    End.
  Retrun ⟨γ, Φ1⟩;
End (Res).
```

过程 delete(S)表示从集合 S 中选择一个元素,把它作为返回值,并将它从 S 中删除.

```
InResC (σ,A,B,Φ)
  Λ = {L|L 是 A 中的文字};
  L = delete(Λ); Ω1 = InResL(L,Φ); Θ = ∅;
  For (each ⟨L,σ1, Ψ⟩∈Ω1) do
    Begin
      Λ1 = Λ;
      while (Λ1 非空) do
        Begin
          L = delete(Λ1); σ2 = NIL; Ψ' = {∅};
          For (σ1 中的每一个 σ') do
            Begin
              Ω = InResL(σ'(L),Φ);
              If(Ω 非空) then
                For (每一个⟨L,σ', Ψ''⟩∈Ω) do
                  Begin
                    Ψ' = {Γ1 ∪ Γ2 | Γ1∈ Ψ',Γ2∈ Ψ''};
                    σ2 = σ2|σ1∘σ' ∪ σ'|Var(L);
                  End
              Else 结束 while 循环;
            End
          σ1 = σ1∘σ2;
        End
      Θ = Θ ∪ {⟨σ∘σ1, Ψ'⟩};
    End
  Return Θ;
```

```
End(ResC).

InResL(L,Φ)
  Θ = ∅; ⟨γ,Φ1⟩= InRes(L,Φ);
  If (γ是空集) then Return ∅;
  else Begin
     γ1 = {⟨σ,⊥,B⟩|⟨σ,⊥,B⟩∈γ};
     Ω ={⟨L,σ,{{B}}⟩|⟨σ,⊥,B⟩∈γ1};
     γ=γ−γ1; γ2 =Y;
     while (γ非空) do
       Begin
         ⟨σ,A′,B′⟩= delete(γ);
         Φ3 = {A|⟨σ,A,B⟩∈γ2}−{A′};
         Φ′ = Φ1 ∪ Φ3;
         Θ = Θ ∪ InResC(σ,A′,B′,Φ′);
       End.
     End.
  Ω = Ω ∪ {⟨L,σ,Y⟩|⟨σ,Y⟩∈Θ};
  Return Ω;
End(ResL).
```

定理 1.1. 对于子句 A 和协调的子句集 Γ,若 A 和协调的子句集 Γ 是不协调的,则过程 Incon-subset(A,Γ)返回的集合 Ψ 就是 Γ 的和 A 极小不协调子集的集合;否则 Ψ 为空集.

证明:因为该过程是一个完全枚举的过程,根据引理 1.1~引理 1.4 可知,结论成立. □

2 典型的信念修正方法的实现

得到所有的极大协调子集以后,就可以实现信念修正的很多典型方法,具体叙述如下:

(1) 全部交集的方法:设 Ψ 是所有极小不协调子集的集合,则集合 $\Gamma-\{B|$存在一个集合 $\Gamma'\in\Psi$,使得 $B\in\Gamma'\}\cup\{A\}$ 就是修正后的集合.极大选择的方法:可以求得一个和 A 极大协调的子集 Γ',修正后得到的集合为 $\Gamma'\cup\{A\}$.部分交集的方法:求得部分和 A 极大协调的子集 $\Gamma_1,\Gamma_2,\ldots,\Gamma_k$,修正后得到的集合为 $(\Gamma_1\cap\Gamma_2\cap\ldots\cap\Gamma_k)\cup\{A\}$.

(2) 信念基上基于认识牢固度的方法.在基于极小不协调子集的方法中,当求得所有极小不协调子集之后,可以按如下方法删除原语句集中的语句:找出所有极小不协调的语句集中的认识牢固度最小的语句,将它从原语句集中删除,并且将所有包含该语句的极小不协调的子集删除,这样对剩余的极小不协调的子集再按照上面的方法进行处理,直到将所有极小不协调的子集处理完,就得到了我们想要的那个极大协调的子集 Γ_1.然后求出 $\Gamma_1\cup\{A\}$ 的闭包就得到最终结果.该过程用 REE(Γ,A,<)表示.

如果引起矛盾的语句的认识牢固度可以是相同的,也就是说,在极小不协调的语句集中可能存在两个认识牢固度相同的语句,从它们中删除任何一个都可能得到极大协调的语句集.

(3) 基于模型的方法得到的语句集不一定是极大的,如果选择不当,则可能会丢失一些信息.下面定义的基于模型的方法,就能消除这一问题.设 Ψ' 为 Γ 的所有和 A 极大协调子集的集合,修改后的方法描述如下:

Dalal 的方法:	$\Gamma*A=\Psi'\cap(\Gamma*_D A)$;	Satoh 的方法:	$\Gamma*A=\Psi'\cap(\Gamma*_s A)$;
Borgida 的方法:	$\Gamma*A=\Psi'\cap(\Gamma*_B A)$;	Weber 的方法:	$\Gamma*A=\Psi'\cap(\Gamma*_W A)$;
Forbus 的方法:	$\Gamma*A=\Psi'\cap(\Gamma*_F A)$;	Winslett 的方法:	$\Gamma*A=\Psi'\cap(\Gamma*_{win} A)$.

(4) 迭代以及 TMS 的方法.目前,迭代的方法主要基于认识牢固度方法.通过第 1 节可以看出,信念归结方法同样也可以实现基于认识牢固度的方法,这里我们进一步地讨论如何通过信念归结方法来实现迭代的方法.这里设 π 为修正序列 $A_1,A_2,\ldots,A_n$,并且设 REE(Γ,A,<)是上述(2)中给出的过程:找出所有极小不协调的语句集中的认识牢固度最小的语句,将它从 Γ 中删除,并且将所有包含该语句的极小不协调的子集删除,这样对剩余的极小不协调的子集再按照上面的方法进行处理,直到将所有极小不协调的子集处理完,则得到了我们想要的极大协调的子集 Γ_1.

我们用 C++语言实现了信念修正的一个原型系统,用类似 Prolog 的语法形式来描述子句,例如子句 $A(x)\vee B(x,y)\vee D_1(f(x,y))$ 就是如下的形式:"$A(x);B(x,y);D_1(f(x,y))$".,用符号";"表示逻辑或,用符号"."表示一个子句

的结束,与 Prolog 的规则相同.该原型系统由 4 部分组成:词法分析器、信念修正模块、合一模块和输入输出界面.当加入的子句和原来的子句集发生矛盾时,就报告用户发现了矛盾,然后提示用户采用什么样的修正方法,根据用户选择的修正方法进行修正.

还需要说明的两个问题是可判定性和计算复杂性.首先,根据一阶逻辑的不可判定性可知,针对一阶逻辑的结果只是对可判定的情况成立,如果没有特别说明,前面讨论一阶逻辑时,是指可判定的情况.虽然一阶逻辑的协调性检查是不可判定的,但很多实际问题的协调性检查却是可判定的,很多文献讨论过这个问题.其次,上述过程即便是对于可判定的情况,时间复杂性仍然是指数形式的,效率比较低,但一些特殊情况存在多项式时间算法,我们已经在其他文章中进行了阐述.

3 与相关工作的比较

真值维护系统[1]是第 1 个与信念修正有关的系统,该系统中每一个信念都附有论据,论据说明了在什么情况下该信念为真.从这里我们可以看到,真值维护系统已经首先给出了一个信念为真的条件,也就是事先已经给出了推理链,当产生矛盾时,只要改变这个推理链上的某些信念的状态就可以了.现实中不可能对每一个信念给出它成立的论据,即便是一个信念,有时也不可能给出它的所有论据,因为论据含在已知的事实和规则中,需要通过进行推理的一个过程才能找到.

Dixon[25]给出了信念修正的一个系统.该系统是基于认识牢固度的思想.他首先将 Wobcke[20]的理论基进一步限制为有限基,然后讨论了由基中信念的可信度得到基所能推出的信念的可信度.对于每一个信念都赋给它一个整数,称为该信念的秩,一个信念的秩越大,其可信度越大.并且假设在信念集中,公理的秩是最大的.若公理的秩为 n,则可以将信念集分为如下的集合:$\Gamma_1,\Gamma_2,\ldots,\Gamma_i,\ldots,\Gamma_n$,集合 Γ_i 中的信念的秩为 i.令 $\overline{\Gamma_i}=\bigcup_{j=1}^{n}\Gamma_j$,信念 A 的秩如下确定:

$$\text{rank}(\Gamma, A)=\begin{cases}\text{Max}\{\ i \mid \overline{\Gamma_i}\text{推出 } A\},\ \text{if } \overline{\Gamma_i}\text{推出 } A\\ 0,\qquad\qquad\qquad\ \text{if } \overline{\Gamma_i}\text{推不出 } A\end{cases},$$

则最保守的认识牢固关系定义如下:$A\leq_E B$ 当且仅当 $\text{rank}(\Gamma,A)\leq\text{rank}(\Gamma,B)$.利用定义的这个关系,Dixon 给出了约简和修正的过程.在该过程中,若信念 B 的可信度比要加入的信念 A 的可信度小,则从 Γ 中删除.Williams[28]也讨论了基于认识牢固度方法的实现问题,所采用的策略是,在没有理由修改某个信念的秩时,就保持它不变.基于认识牢固度的方法强调:修正时尽量对信念的认识牢固度做最小的修改,但是有些信念对协调性并没有什么影响,可能只是因为它的可信度比较小而被删除.其实,如果将信念集限制为有限的理论基,则信念集也就是一个知识库,信念修正问题也就是知识库维护问题.对于知识库的维护问题,从推理的角度考虑,在保持协调性的情况下,应该尽量少地删除规则.所以该方法并不合适.另外,Dixon 给出的这种可实现的认识牢固度还对信念基做了很多限制,例如不允许信念集是冗余的.

Damasio,Nejdl 和 Pereira[29]给出了用于扩充逻辑程序中知识库修正的系统,称为 REVISE.其基本思想是求出一个逻辑程序能推出的所有的结论来,根据这一结果来消除矛盾.消除矛盾的方法并不是从信念集中删除一些规则,而是在信念集中增加一些规则.因为增加一些规则使得某个语句 not(A)中的 A 由不成立变为成立,使得 not(A)不成立,由此消除了矛盾.但是这种方法会导致重复执行修正过程.

References:

[1] Doyle J. A truth maintenance system. Artificial Intelligence, 1979,12(3):231~272.

[2] Kleer JD. An assumption-based TMS. Artificial Intelligence, 1986,28(2):127~162.

[3] Fagin R, Ullman JD, Vardi MY. On the semantics of updates in databases. In: DeWitt DJ, Gardarin G, eds. Proceedings of the 2nd ACM SIGACT-SIGMOD Symposium on Principle of Database Systems. New York: ACM Press, 1983. 352~365.

[4] Ginsberg ML, Smith DE. Reasoning about action I: A possible worlds approach. Artificial Intelligence, 1988,35(2):165~195.

[5] Dalal M. Investigations into a theory of knowledge base revision: Preliminary report. In: Mitchell TM, Smith RG, eds. Proceedings of the 7th National Conference on Artificial Intelligence. AAAI Press, 1988. 475~479.

[6] Satoh K. Nonmonotonic reasoning by minimal belief revision. In: ICOT, ed. Proceedings of the International Conference on the 5th Generation Computer System. Berlin: Springer-Verlag, 1988. 455~462.

[7] Borgida A. Language features for flexible handling of exception in information systems. ACM Transactions on Database System, 1985,10(4):536~603.

[8] Weber A. Updating propositional formulas. In: Kerschberg L, ed. Proceedings of the 1st Conference on Expert Database Systems. Menlo Park: Benjamin Cummings, 1986. 487~500.

[9] Forbus KD. Introducing actions into qualitative simulation. In: Sridharan NS, ed. Proceedings of the International Joint Conference on Artificial Intelligence. San Francisco: Morgan Kaufmann Publishers, 1989. 1273~1278.

[10] Winslett M. Reasoning about action using possible models approach. In: Mitchell TM, Smith RG, eds. Proceedings of the 7th National Conference on Artificial Intelligence. AAAI Press. 1988. 89~93.

[11] Alchourron CE, Gardenfors P, Markinson D. On the logic of theory change: Partial meet contraction and revision functions. The Journal of Symbolic Logic, 1985,50(2):510~530.

[12] Gardenfors P. Knowledge in Flux: Modeling the Dynamics of Epistemic States. Cambridge: The MIT Press, 1988.

[13] Gardenfors P, Makinson D. Revisions of knowledge systems using epistemic entrenchment. In: Vardi MY, ed. Proceedings of the 2nd Conference on Theoretical Aspects of Reasoning About Knowledge. San Francisco: Morgan Kaufmann Publishers, 1988. 83~95.

[14] Katsuno H, Mendelzon AO. Propositional knowledge base revision and minimal change. Artificial Intelligence, 1991,52(3): 263~294.

[15] Nebel BA. Knowledge level analysis of belief revision. In: Brachman RJ, Levesque HJ, Reiter R, eds. Proceedings of the 1st International Conference on Principles of Knowledge Representation and Reasoning. San Francisco: Mrogan Kaufman Publishers, 1989. 301~311.

[16] Hansson SO. New operators for theory change. Theoria, 1989,50:114~132.

[17] Fuhrman A. Theory contraction through base contraction. Journal of Philosophical Logic, 1991,20:175~203.

[18] Rott H. A nonmonotonic conditional logic for belief revision I. In: Fuhrman A, Morreau M, eds. The Logic of Theory Change. Berlin : Springer-Verlag, 1991. 135~181.

[19] Williams M. Two operators for theory base change. In: Adams A, Sterling LS, eds. Proceedings of the 5th Australian Joint Conference on Artificial Intelligence. Singapore: World Scientific, 1992. 256~265.

[20] Wobcke WR. A belief revision approach to nonmonotonic reasoning. In: Adams A, Sterling LS, eds. Proceedings of the 5th Australian Joint Conference on Artificial Intelligence. Singapore: World Scientific, 1992. 278~283.

[21] Li W. A open logic system. Science in China (Series A), 1993,22(10):1103~1113.

[22] Li W, Shen NC, Wang J. R-Calculus: A logical approach for knowledge base maintenance. International Journal of Artificial Intelligence Tools, 1995,4(1&2):177~200.

[23] Darwiche A, Pearl J. On the logic of iterated belief revision. Artificial Intelligence, 1997,89(1~2):1~29.

[24] Boutilier C. Revision sequences and nested conditionals. In: Bajcsy R, ed. Proceedings of the 13th International Joint Conference on Artificial Intelligence. San Francisco: Morgan Kaufmann Publishers, 1993. 519~525.

[25] Dixon SE. Belief revision: a computational approach [Ph.D. Thesis]. Sydney: University of Sydney, 1994.

[26] Dixon SE. A finite base belief revision system. Australian Computer Science Communications, 1993,15(1):445~451.

[27] Dixon SE, Wocke WR. The implementation of a first-order logic AGM belief revision system. In: Bajcsy R, ed. Proceedings of the 13th International Joint Conference on Artificial Intelligence. San Francisco: Morgan Kaufmann Publishers, 1993. 534~539.

[28] Williams MA. Towards a practical approach to belief revision: reason-based approach. In: Aiello LC, Doyle J, Shapiro SC, eds. Proceedings of the 5th International Conference on Principles of Knowledge Representation and Reasoning. San Francisco: Morgan Kaufmann Publishers, 1996. 412~420.

[29] Damasio CV, Nejdl W, Pereira LP. REVISE: An extended logic programming systems for revising knowledge bases. In: Doyle J, Sandewall E, Torasso P, eds. Proceedings of the International Conference on Knowledge Representation and Reasoning. San Francisco: Morgan Kaufmann Publishers, 1994. 607~618.

[30] Sullivan JW, Tyler SW. Intelligent User Interfaces. New York: ACM Press, 1991.

[31] Gallier JH. Logic for Computer Science, Foundations of Automatic Theorem Proving. New York: John Wilsey&Sons Inc., 1987.