命题知识库更新的算法及其复杂性*

陶雪红 孙 伟 马绍汉

(山东大学计算机系 济南 250100)

摘要 本文介绍了知识库更新的基本概念及命题知识库更新的复杂性研究现状.近年来,学者们提出了许多方法进行命题知识库的更新,一类是基于公式的方法;一类是基于模型的方法,但所有这些方法在通常情况下都是难解的.本文讨论基于公式的Ginsberg更新方法,并给出在公式个数远小于变元个数情况下的一个多项式时间算法.

关键词 人工智能,知识库更新,算法复杂性,可满足性问题,命题逻辑.

在知识库的设计与管理中,随着人们领域知识的增多,要不断向知识库中补充新知识,即对知识库进行更新(添加新知识、删除错误知识、修改不完善的知识等),使之能较准确地反映人们目前所掌握的知识.对知识库进行更新操作时,可能使知识库产生矛盾,所以还必须考虑如何消除矛盾.在知识库建立初期,由于知识库比较小,内容较简单,对其更新比较容易做到,可手工实现.但随着新知识的不断加入,知识库越来越大,内容越来越丰富,知识库各知识单元之间的相互影响和相互联系随之变得复杂且难以跟踪捉摸,加入新知识后,矛盾的检测和消除变得困难.在这种情况下,知识库的更新便不再适合手工处理,研究知识库的更新算法变得尤为重要.

1 知识库更新的研究现状

在知识库管理中,当人们获得新知识时,就需对原有知识库进行更新.对知识库的更新,从理论上讲,主要有以下3种基本操作[1]:

(1)扩充(Expansion):将一个新命题 p 直接加到原有知识库 T 中,这里新信息与原有信息之间没有矛盾.

(2)修改(Revision):一个与原知识库 T 不相容的新命题 p 被加入,为保证其所产生的新知识库的相容性,就得去掉原知识库中的某些命题.

(3)压缩(Contraction):当一个以前认为正确的命题 p 变得错误时,就要从知识库 T 中删去命题.

* 本文研究得到国家自然科学基金资助.作者**陶雪红**,女,1969年生,硕士,主要研究领域为知识库的更新.**孙伟**,1966年生,讲师,主要研究领域为并行算法,人工智能.**马绍汉**,1938年生,教授,中国科学院计算技术研究所*CAD*开放实验室兼职研究员,主要研究领域为算法分析与设计,人工智能和并行算法.

本文通讯联系人:陶雪红,济南 250100,山东大学计算机系

本文1995-03-03收到修改稿

本文仅讨论修改操作,以下所说的更新都指修改(Revision)操作. 近年来,对知识库的更新,学者们提出了许多具体的方法. 一类是基于公式的方法,当向知识库中加入新知识时,若发生矛盾,就删除原知识库中的一些公式,以维护其相容性. 在这种更新方法下,一个公式或者整个留在知识库中,或者整个被抛弃. 基于公式的方法主要有 Ginsberg 方法和 WIDTIO 方法等[2];另一类是基于模型的方法,这种方法考虑知识库模型的变化,即如何从旧模型转换成更新后知识库的新模型. 典型的方法有 Dalal 方法,Satoh 方法等.[2]所有这些方法都遵守修改最小化原则,即原则上去掉该去掉的信息,尽可能多地保留原有知识.

每种更新方法都是针对某种具体应用领域而提出的,大多数学者认为,这些方法中没有一种可以看作是通用方法. 而且现在普遍认为不存在通用的、独立于应用领域的知识库更新方法. Katsuno 和 Mendelzon 以及其他一些作者都指出[3~6],对知识库更新的研究,虽已提出了许多方法,但在通常情况下都是难解的(Intractable). 因此,我们不能期望找到一种通常情况下知识库更新的多项式时间算法,只能寻找在某些限定条件下,特定算法的易解性. 在命题逻辑中,Winslett 在文献[6]中,限制新命题 p 的长度 $\|p\| \leqslant K$(K 是常量),得到用 Winslett 方法和 Forbus 方法进行知识库更新的算法,其时间复杂性为 $O(\mathrm{Log}\, t)$,其中 t 为知识库中变元的数目. 文献[2]中限制知识库 T 是 Horn 公式的集合,新命题 p 是 Horn 公式且 $\|p\| \leqslant K$(K 是常量),得到用 Satoh 方法和 Winslett 方法进行知识库更新的算法,时间复杂性为 $O(\|T\|)$. 以上这些可解的算法均是采用基于模型的更新方法,对基于公式的更新方法尚未见到多项式时间算法. 本文将讨论基于公式的 Ginsberg 更新方法,并给出一种特定条件下的更新算法.

2　算法设计与分析

2.1　基本概念

本文在命题逻辑的范围内讨论知识库的更新. 我们用 L 表示有限命题变元集 U 上的语言,p,q 表示命题公式,T 表示知识库. 知识库是命题公式的有限集. 每个 $x \in U$ 是个原子,原子或原子的否定叫文字,原子叫正文字,原子的否定叫负文字,文字的析取叫子句. 以下将用符号"~"表示否定.

给定知识库 T 和公式 p,Top 表示向知识库 T 中加入新知识 p 更新后的知识库."o"叫更新操作符,常用下标表示所采用的更新方法,本文仅讨论 Ginsberg 方法.

定义.[2] Ginsberg 更新方法

令 T 是可满足的知识库,令

$$W(p,T)=\{T' \subseteq T \mid T' \not\vdash \sim p, T' \subset S \subseteq T \Rightarrow S \vdash \sim p\}$$

即 $W(p,T)$ 是 T 中所有与 p 相容的极大命题公式集的集合. 则

$$T o_G p=\{T' \cup \{p\} \mid T' \in W(p,T)\}$$

这就是说,$To_G p$ 是由一些子知识库构成,每个子知识库等于 T 中与 p 相容的极大命题公式集再添加上 p.

例:　　$T=\{a,b,a \wedge b \Rightarrow c\}, p=\sim c$,则

$W(p,T)=\{\{a \wedge b \Rightarrow c, a\},\{a \wedge b \Rightarrow c, b\},\{a,b\}\}$,因此

$$To_G p=\{\{a\wedge b\Rightarrow c,a,\sim c\},\{a\wedge b\Rightarrow c,b,\sim c\},\{a,b,\sim c\}\}.$$

在通常情况下,求 $To_G p$ 属于难解问题.如果限制公式均为子句形式,且 T 中公式个数远小于 $T\cup\{p\}$ 中变元个数,则求 $To_G p$ 是多项式时间可解的.下面我们先给出算法描述,然后分析其复杂性.

2.2 算法描述

算法的基本思想是首先对公式和变元的数目进行缩减,然后通过枚举缩减后变元集上的所有指派,求出 T 中可与 p 同时为真的公式集的集合,即与 p 相容的公式集的集合,进而求出所有与 p 相容的极大公式集的集合,最终得到 $To_G p$.文献[7]中曾采用这种先缩减变元集,再对其枚举的思想在多项式时间内解决了一类可满足性问题.

首先,定义变元集上的指派.变元集 U 上的指派是一个函数 $r:U\to\{True,False\}$,使 U 中每个变元对应一个真值($True$)或假值($False$).如果所有指派均使公式 q 取真值,则该公式称为重言式.其次,定义变元的出现模型(Pattern of Occurrence).[7]设子句集 $S=\{f_1,f_2,\ldots,f_n\}$,S 中不含重言式(同时包含某变元及其否定的子句是重言式),S 中所有变元的集合为 U.一个变元 $u\in U$ 的出现模型是长度为 n 的一维数组 A_u,其中

$$A_u[i]=\begin{cases}+1,u\text{ 出现在公式 }f_i\text{ 中}\\-1,\sim u\text{ 出现在公式 }f_i\text{ 中}\\0,u\text{ 和}\sim u\text{ 都未在公式 }f_i\text{ 中出现}\end{cases}\qquad(1\leqslant i\leqslant n)$$

设 $u_1,u_2\in U$,如果对所有 $i(1\leqslant i\leqslant n)$ 都有 $A_{u_1}[i]=A_{u_2}[i]$,则称 A_{u_1} 和 A_{u_2} 相同;如果对所有 $i(1\leqslant i\leqslant n)$ 都有 $A_{u_1}[i]=-A_{u_2}[i]$,称 A_{u_1} 和 A_{u_2} 相反.

算法. $REVISION(T,p)$

输入:可满足知识库 $T=\{f_1,f_2,\ldots,f_n\}$ 和公式 p,且所有公式均为子句形式.设 $T\cup\{p\}$ 中所有变元的集合为 $\{u_1,u_2,\ldots,u_m\}$.

输出:$To_G p$.

步骤:

step1:$S=T\cup\{p\}$ /* 即 $S=\{f_1,f_2,\ldots,f_n,p\}$. */

$U=\{u_1,u_2,\ldots,u_m\}$ /* U 为 S 中所有变元的集合. */

$R=\Phi$ /* 集合 R 存放同时属于 $W(p,T)$ 所有元素的公式. */

step2:(2.1)将所有重言式从 S 中删除并加入集合 R.

/* 重言式总可与 p 同时为真,即总与 p 相容,因此,属于 $W(p,T)$ 的任何元素. */

(2.2)若 $p\in R$,则 $To_G p=T\cup\{p\}$,返回.

/* 若 $p\in R$,则 p 是重言式,可满足知识库 T 与 p 相容,因此,$To_G p=T\cup\{p\}$. */

step3:对 S 中出现的每个变元,构造其对应的出现模型.

step4:(4.1)若有两个变元的出现模型相同,则将含这 2 个变元的所有子句从 S 中删除并加入 R.

/* 令这 2 个变元一个为真,一个为假,可使含这 2 个变元的所有子句为真,且不影响其它子句.可见,这些子句总可以与 p 同时为真,即总与 p 相容,因此,它们属于 $W(p,T)$ 的任何元素. */

(4.2)若 $p\in R$,则 $To_G p=T\cup\{p\}$,返回.

/* 若 $p\in R$,则存在 2 个变元 u_1 和 u_2 的出现模型相同,且 p 包含这 2 个变元,在执行(4.1)时将 p 和含这 2 个变元的其它子句加入 R. 设 r_T 为使可满足知识库 T 中所有公式同时为真的一个指派,我们令 u_1 和 u_2 一个为真,一个为假,$T\cup\{p\}$中其余变元若在 T 中出现过,则按指派 r_T 取值;否则任意取值,可使 $T\cup\{p\}$中公式同时为真,即 T 与 p 相容,所以,$To_Gp=T\cup\{p\}$. */

step5:(5.1)若 2 个变元的出现模型相反,则将含这 2 个变元的所有子句从 S 中删除并加入 R.

/* 令这 2 个变元同真或同假,可使含这 2 个变元的所有子句为真,且不影响其它子句. 同理,这些子句属于 $W(p,T)$的任何元素. */

(5.2)若 $p\in R$,则 $To_Gp=T\cup\{p\}$,返回.

/* 若 $p\in R$,则存在 2 个变元 u_1 和 u_2 的出现模型相反,且 p 包含这 2 个变元,在执行(5.1)时将 p 和含这 2 个变元的其它子句加入 R. 设 r_T 为使可满足知识库 T 中所有公式同时为真的一个指派,我们令 u_1 和 u_2 同为真或同为假,$T\cup\{p\}$中其余变元若在 T 中出现过,则按指派 r_T 取值;否则任意取值,可使 $T\cup\{p\}$中公式同时为真,即 T 与 p 相容,所以,$To_Gp=T\cup\{p\}$. */

step6:令 S'是 S 中剩余公式的集合,U'是 S'中变元的集合.

(6.1)$W=\Phi$

for 变元集 U'上的每种指派 r do

(6.2) if 指派 r 使 p 为真 then

(6.3) $Wr=\{f\mid f\in S'-\{p\}$,且指派 r 使 f 为真$\}$

/* Wr 是 T 中一个可与 p 同时为真的命题公式集,即一个与 p 相容的命题公式集. */

(6.4) if $Wr\notin W$ then $W=W\cup\{Wr\}$ endif

/* 该语句保证 W 中元素互不相同. */

endif

endfor.

/* W 是 T 中与 p 相容的命题公式集的集合. */

step7:$W(p,T)=\{X\cup R\mid X\in W$,且不存在 $Y\in W$,使 $X\subset Y\}$

/* $W(p,T)$是 T 中所有与 p 相容的极大命题公式集的集合. */

step8:$To_Gp=\{X\cup\{p\}\mid X\in W(p,T)\}$

2.3 算法分析

对算法进行描述的同时,我们在注释中对各步的正确性分别做了证明. 在算法 REVISION 中,step1 需 $O(1)$时间. step2,3,4,5 均需 $O(m^2n)$时间. 在 step6 中,(6.1)需 $O(1)$时间. 设 for 语句共循环 M 次,每次循环中,(6.2)需 $O(m)$时间,(6.3)需 $O(mn)$时间,(6.4)的复杂性主要来自对条件 $Wr\notin W$ 的判定,这需将 Wr 与当前 W 中至多$(M-1)$个元素逐个比较,共需 $O(Mm^2n^2)$时间. 所以 step6 可在 $O(M^2m^2n^2)$时间内完成. step7 要在至多 M 个元素的集合 W 中找出所有极大元素,需 $O(M^2m^2n^2)$时间. step8 需 $O(M)$时间. 因此,算法 REVISION的时间复杂性可表示为 $O(M^2m^2n^2)$. 设 U'中变元的个数为 m'则 $M=2^{m'}$,所以,

在未对 m 和 n 施加限制的情况下,该算法不是多项式时间算法.

在算法中,变元的出现模型是长度为($n+1$)的一维数组,数组的每个元素为+1、-1或,且数组的所有元素不同时为0,所以共有($3^{n+1}-1$)种不同的出现模型.由于step6开始时,U'中变元的出现模型各不相同,且没有2个变元的出现模型相反,因此,U'中变元个数m'满足

$$m' \leqslant \frac{3^{n+1}-1}{2}$$

若使该算法成为多项式时间算法,只需存在固定常量$k(k>0)$,满足

$$2^{m'} \leqslant m^k,$$

$$2^{(3^{n+1}-1)/2} \leqslant m^k,$$

$$\frac{3^{n+1}-1}{2} \cdot \ln 2 \leqslant k \cdot \ln m,$$

$$3^{n+1} \leqslant \frac{2k \cdot \ln m}{\ln 2}+1,$$

$$n \leqslant \frac{\ln(2k \cdot \ln m+\ln 2)-\ln\ln 2}{\ln 3}-1 \tag{1}$$

即(1)式成立时,REVISION为多项式时间算法.特别地,令$k=1/2$,则根据上述推导,当

$$n \leqslant \frac{\ln(\ln m+\ln 2)-\ln\ln 2}{\ln 3}-1 \tag{2}$$

则$2^{m'} \leqslant m^{1/2}$,即$M \leqslant m^{1/2}$,REVISION的时间复杂性为$O(m^3n^2)$.

因为$\ln m \geqslant 0$,$0<\ln 2<1$,所以$\ln m+\ln 2 \geqslant \ln m \geqslant \ln m \cdot \ln 2$,由此得

$$\frac{\ln\ln m}{\ln 3}=\frac{\ln(\ln m \cdot \ln 2)-\ln\ln 2}{\ln 3} \leqslant \frac{\ln(\ln m+\ln 2)-\ln\ln 2}{\ln 3}$$

因此$n \leqslant (\ln\ln m)/\ln 3-1$时,(2)式成立,REVISION为$O(m^3n^2)$时间算法.综上所述,可得如下定理:

定理. 当知识库T中公式和新公式p均为子句形式,且T中公式个数n和$T \cup \{p\}$中变元个数m满足$n \leqslant (\ln\ln m)/\ln 3-1$时,求$T \circ_G p$是多项式时间可解的,算法REVISION的时间复杂度为$O(m^3n^2)$.

3 结束语

知识库是人工智能系统的核心,知识库的更新是目前人工智能研究的一个热点.知识库的更新具有非单调性,即知识库中一些正确的知识,随着知识的增长,可能成为错误的.目前非单调推理研究已取得一些成果,将一些著名非单调推理系统,如界限理论(Circumscription)、缺省逻辑(Default Logic)、真值维持系统(Truth Maintenance System)中的一些方法用于知识库的更新,可能得到更好的结果,研究新的知识库更新算法是我们下一步做的工作.

参考文献

1 Gardenfors P. Epistemic importance and minimal changes of belief. Australasian J. Philos, 1984,**62**:136~157.

2 Eiter T, Gottlob G. On the complexity of propositional knowledge base revision, updates, and counterfactuals. Artificial Intelligence, 1992, 57: 227～270.

3 Katsuno H, Mendelzon A O. On the difference between updating a knowledge base and revising it. In: Proceedings KR－91, 1991. 387～395.

4 Grahne G. Updates and counterfactuals. In: Proceedings KR－91, 1991. 269～276.

5 Nebel B. Belief revision and default reasoning: syntax－based approaches. In: Proceedings KR－91, 1991. 417～428.

6 Winslett M. Updating logical databases. Cambridge, England: Cambridge University Press, 1990.

7 Paul Walton Purdom Jr, Cynthia A Brown. Polynomial－average－time satisfiability problems. Information Sciences, 1987, 41: 23～42.

THE ALGORITHM AND COMPLEXITY OF PROPOSITIONAL KNOWLEDGE BASE REVISION

Tao Xuehong　Sun Wei　Ma Shaohan

(*Department of Computer Science　Shandong University　Ji'nan*　250100)

Abstract　This paper gives an outline of knowledge base revision and some recently presented complexity results about propositional knowledge base revision. Different methods for revising propositional knowledge base have been proposed recently by several researchers, some are formula－based methods and the others are model－based methods, but all methods are intractable in the general case. This paper discusses a formula－based method——Ginsberg's method, and presents a polynomial algorithm when the number of formulas is far less than the number of variables.

Key words　Artificial intelligence, knowledge base revision, algorithmic complexity, SAT problem, propositional logic.