软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2012,23(10):2550−2563 [doi: 10.3724/SP.J.1001.2012.04194]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

一种有效的分层加权库编译方法*

赖 永[1,2], 刘大有[1,2+]

[1](吉林大学 计算机科学与技术学院,吉林 长春 130012)

[2](符号计算与知识工程教育部重点实验室(吉林大学),吉林 长春 130012)

Efficient Compilation Approach on Stratified Weighted Bases

LAI Yong[1,2], LIU Da-You[1,2+]

[1](College of Computer Science and Technology, Jilin University, Changchun 130012, China)

[2](Key Laboratory of Symbolic Computation and Knowledge Engineering of Ministry of Education (Jilin University), Changchun 130012, China)

+ Corresponding author: E-mail: dyliu@jlu.edu.cn, http://www.jlu.edu.cn

Lai Y, Liu DY. Efficient compilation approach on stratified weighted bases. *Journal of Software*, 2012, 23(10):2550−2563 (in Chinese). http://www.jos.org.cn/1000-9825/4194.htm

Abstract: To be in accordance with the thinking habits of human and improve the efficiency of reasoning, this study argues to stratify weighted bases. The study shows that the existing compilation approach to non-stratified weighted bases can also be applied to COMPILE stratified weighted bases; however, its time and space costs are relatively high because of redundant information in the compilation results. The paper proposes a novel compilation approach, which can remove the redundant information in the process of compilation, and presents two optimization techniques to further improve the time efficiency. As with the existing approach, re-compiling a stratified weighted base is not required whenever the weights associated with soft constraints change with time. The approach is tested by compiling random instances into ROBDD-normal bases, and the preliminary experimental results show that the time and space costs of this approach are lower than the existing approach for most instances.

Key words: preference; penalty logic; stratified weighted base; knowledge compilation

摘 要: 提议对加权库进行分层,一方面符合人类的思维习惯,另一方面能够提高推理效率.首先说明现有的针对非分层加权库的编译方法也适用于编译分层加权库,但是,由于存在较多冗余信息而效率不高.提出一种新的编译方法,能够在编译过程中去除冗余信息,并提出两种优化技术提高时间效率.该方法与现有方法相同,当软约束权值改变时无需重新编译.选择 ROBDD 为目标语言,使用随机问题对该方法进行测试.结果表明:对于非分层加权库,该方法的空间效率高于已存在方法;对于分层加权库,该方法的时间和空间效率均高于已存在方法,且当层数越多时,该方法的效率越高.

关键词: 偏好;惩罚逻辑;分层加权库;知识编译

中图法分类号: TP181 **文献标识码**: A

* 基金项目: 国家自然科学基金(61133011, 61170092, 60973088, 60873149); 吉林大学研究生创新基金(20111060)
收稿时间: 2011-08-30; 定稿时间: 2012-01-17

如何对偏好进行表示和推理,是人工智能、数据库和运筹学等领域的重要问题之一[1−4].惩罚逻辑(penalty logic)是一种基于逻辑形式化的偏好表示方法,具有可读性好、能对偏好关系进行压缩表示等特点[5−7],其高效的推理方法在非单调推理、缺省推理和软约束求解等问题中有着重要应用[8,9].

惩罚逻辑中,用带权值(正整数或+∞)的命题公式表示约束,加权库为有限约束集.在任意加权库下,所有状态都被附加一个惩罚,其值等于加权库中不被该状态满足的公式的权值总和,最偏好状态为惩罚最小的状态.与其他逻辑形式化方法相同,模型检测(model checking)和子句推理(clausal inference)是惩罚逻辑中两个重要的问题.对于给定的加权库,前者判断给定状态是否为最偏好状态,后者判断是否加权库的任意最偏好状态都满足给定子句,二者的不易处理性(intractability)是限制惩罚逻辑在实际中被广泛使用的主要原因之一[9].

作为解决不易处理性的重要研究方向,知识编译的主要思想是将推理过程分为离线的编译阶段(off-line compilation phase)和在线的查询应答阶段(online query-answering phase),在离线时,将知识库或信念库编译为易处理的目标语言(target language),从而有效地支持在线查询[10−12].在文献[9]中,Darwiche 和 Marquis 对文献[13]的工作进行了扩展,将知识编译方法引入惩罚逻辑,提出一个能将任意加权库编译为等价 L-规范加权库的框架,其中,L 为命题逻辑的任意完备子集,该编译过程被称为加权库的 L-编译.为了便于叙述,下文称该工作为 DM 框架或 DM 方法.对于一些命题逻辑片段,例如精简有序二元决策图(reduced ordered binary decision diagram,简称 ROBDD)[14]、可分解否定范式(decomposable negation normal form)[15]等,通过 DM 方法编译之后的 L-规范加权库能够支持多项式时间的模型检测和子句推理.Darwiche 和 Marquis 指出,文献[13]中的工作对应于 DM 框架下的 ROBDD-编译.

离线阶段的编译时间和编译结果的大小是任意知识编译方法的两个重要方面.文献[13,16]中的工作以及我们的实验结果表明,DM 方法耗费了较多的编译时间,其编译结果的规模较大,影响了它在实际中的应用.

注意到,在一些与惩罚逻辑关系密切的形式化表示与推理方法中,例如 Kappa 演算[17]、基于相干性的非单调推理方法(coherence-based nonmonotonous reasoning approach)[13,16]等,为了提高推理效率,通常将知识库或信念库进行分层处理.本文借鉴该思想,提出分层加权库的概念.分层的主要思想是,按权值大小对约束进行分层,将权值相差不大的约束分在相同层,相差较大的分在不同层,以便于推理.需要指出的是,对加权库进行分层的思想符合人类的思维习惯.例如,人们去商场购买衣服时,头脑中会存在一些关于衣服的尺寸、价位、款式、颜色等限制条件或约束,一些约束的重要性(例如尺寸)明显高于另外一些(例如颜色),因而人们会有意或无意地对它们进行分层处理,重要性高的约束需优先得到满足.

本文致力于分层加权库的编译工作,首先说明 DM 方法也可用于编译分层加权库,但编译结果中存在较多冗余信息.本文将这些信息分为强冗余信息和弱冗余信息:去除强冗余信息后,不影响最后模型检测和子句推理结果;去除弱冗余信息后,若分层加权库不重新分层,也不影响最后模型检测和子句推理结果.基于此,我们提出一种新的编译方法,可根据实际应用中加权分层库的层数变化频繁与否,在编译过程中选择去除所有冗余信息或仅去除强冗余信息.为进一步提高该方法的时间效率,我们基于 GSAT[18]和 ROBDD,提出两种优化技术.最后,在随机问题上,对我们的方法和 DM 方法进行了比较.结果表明,无论是否分层,本文方法的空间效率都高于 DM 方法;当层数大于 1 时,本文所提方法的时间效率高于 DM 方法;层数越多时,本文方法的时间和空间效率越高.

本文第 1 节介绍相关的背景知识.第 2 节给出分层加权库的定义.第 3 节说明可用 DM 方法对分层加权库进行编译.第 4 节、第 5 节给出我们的编译方法及其优化技术.第 6 节对我们的方法与 DM 方法进行实验比较.最后为结论.

1 背景知识

在本文中,分别使用 x,y,z 等表示布尔变量,X,Y,Z 等表示变量集.文字 l 表示变量 x 或其否定 $\neg x$,项和子句分别为若干文字的合取($\wedge$)和析取($\vee$).公式由逻辑常量 true、false、变量以及逻辑连接符 $\neg,\wedge$ 和 $\vee$ 组成.变量集 X 上的状态 ω 由 X 中变量或其否定组成,且每个变量正好出现 1 次,2^X 表示 X 上所有状态的集合.给定 X 上状态 ω 和公式 φ,ω 满足和不满足 φ 分别记为 $\omega\models\varphi$ 和 $\omega\not\models\varphi$,其中,$\omega\models$true 和 $\omega\not\models$false 恒成立,$\omega\models\neg\varphi$ 当且仅当 $\omega\not\models\varphi,\omega\models\varphi\wedge\varphi'$ 当

且仅当$\omega\models\varphi$且$\omega\models\varphi'$,$\omega\models\varphi\vee\varphi'$当且仅当$\omega\models\varphi$或$\omega\models\varphi'$.当存在状态$\omega$满足公式或公式集中所有公式时,称该公式或公式集一致,ω称为其模型.给定 X 上的公式φ和φ',φ蕴含φ'(记为$\varphi\models\varphi'$)当且仅当 X 上满足φ的状态都满足φ',φ与φ'等价(记为$\varphi\equiv\varphi'$)当且仅当二者相互蕴含.由于任意公式都能在线性时间内等价转化为逻辑常量或不含常量的公式(将φ中$\neg$true,$\neg$false,true$\wedge\varphi'$,false$\wedge\varphi'$,true$\vee\varphi'$和 false$\vee\varphi'$部分分别替换为 false,true,φ',false,true 和φ',直到φ为true,false 或不含常量),下文假定非常量公式不含常量.给定公式φ和一致项 T,φ对于 T 的调整(conditioning,记为$\varphi|T$)定义为通过如下方式得到的公式:首先,对于出现在 T 中的文字 x,将φ中所有 x 替换为 true,对于出现在 T 中的文字$\neg y$,将φ中所有 y 替换为 false,然后去除其中常量.

知识编译是解决通用推理问题不易处理性的重要方法之一,其基本思想是,首先,离线地将知识库或信念库编译为易处理的目标语言,从而高效地应答在线查询,离线阶段的编译时间可在多次在线查询中补偿回来.目标语言是任意编译方法的重要方面之一,到目前为止,已提出多种目标语言,包括 Horn 理论、ROBDD、DNNF 和 EPCCL(each pair containing complementary literal)理论[19]等,其中,ROBDD 是易处理性最好的语言之一,且应用最为广泛.本文将以 ROBDD 为例,对我们的编译方法进行说明和测试.为了完整性起见,我们根据文献[15]对 ROBDD 定义如下:

定义 1. 二元决策图(BDD)为有根的有向无环图,其中,节点分为终止节点和非终止节点,终止节点又分为$\top$和$\bot$.任意非终止节点 v 都被标记为变量 $var(v)$,包含左右子节点 $lo(v)$和 $hi(v)$,v 与其左右子节点之间的边分别用虚线和实线表示.任意节点 v 表示的公式递归定义如下:

$$\phi(v)=\begin{cases}\text{false}, & v=\bot\\ \text{true}, & v=\top.\\ (\neg var(v)\wedge\phi(lo(v)))\vee(var(v)\wedge\phi(hi(v))), & \text{其他}\end{cases}$$

定义 2. 满足如下条件的二元决策图称为有向二元决策图(OBDD):

1) 变量集被附加一个线性序$\prec$;

2) 给定任意节点 u 及其子节点 v,若 v 为非终止节点,则 $var(u)\prec var(v)$.

定义 3. 满足如下条件的有向二元决策图称为精简有向二元决策图(ROBDD):

1) 不存在同构节点,即任意不同节点不同时包含相同的变量和左右子节点;

2) 任意节点的子节点不同.

任意公式都存在唯一的 ROBDD 与之对应.在构造 ROBDD 时,给定节点 v,本文使用 $MK(v)$保证结果的精简性.也即,若与 v 同构的节点已经存在,则返回原来节点;若 v 的左右子节点相同,则返回子节点.利用哈希表,$MK(v)$可在常数时间终止[20].给定根为 v 的 ROBDD,下文使用$|v|$表示其中节点数目.

2 分层加权库的定义

为了便于描述,本文忽略所有硬约束的权值$+\infty$,为每条软约束赋予一个 ID 变量.

定义 4. 变量集 X 上层数为 m 的分层加权库为 m+1 个元素的序列 $B=(H,S_1,\ldots,S_m)$,其中,H 为硬约束集,硬约束为 X 上的公式.第 $i(1\leqslant i\leqslant m)$层的 S_i 为软约束集,每条软约束为三元组$\langle z,\varphi,\kappa\rangle$,$z$ 称为 ID 变量;φ为 X 上的公式;正整数κ为权值,即,当φ不被满足时需付出的代价.

分层加权库 B 对于一致项 T 的调整(记为 $B|T$)定义为分别对其中公式按照 T 进行调整得到的分层加权库.下文中,对于任意分层加权库 B 的状态ω,不加说明时默认 B 和ω基于相同变量集.

给定分层加权库 $B=(H,S_1,\ldots,S_m)$,对于状态ω和状态集Ω,本文引入如下表示:

- $V_W(\omega,i)$表示第 i 层$(1\leqslant i\leqslant m)$中被$\omega$违反的软约束,即 $V_B(\omega,i)=\{\langle z,\varphi,k\rangle\in S_i:\omega\not\models\varphi\}$;
- $V_B(\omega)$表示被ω违反的所有软约束,即 $V_B(\omega)=V_B(\omega,1)\cup\ldots\cup V_B(\omega,m)$;
- $V_B(\Omega)$表示由Ω中所有元素违背的软约束集组成的簇,即 $V_B(\Omega)=\{V_B(\omega):\omega\in\Omega\}$.

定义 5. 给定变量集 X 上的分层加权库 $B=(H,S_1,\ldots,S_m)$,状态ω在 B 下的权值定义为

$$K_B(\omega)=(K_B(\omega,1),\ldots,K_W(\omega,m)),$$

其中,对于任意 $1\leqslant i\leqslant m$,有

$$K_B(\omega,i)=\begin{cases}\sum\limits_{\langle z,\varphi,\kappa\rangle\in V_B(\omega,i)}\kappa, & \omega\vDash H\\ +\infty, & \text{否则}\end{cases},$$

其中,$\prec$为$\{K_B(\omega):\omega\in 2^X\}$上的字典序关系.也即给定任意状态$\omega$和$\omega'$,$K_B(\omega)\prec K_B(\omega')$当且仅当存在 $1\leqslant i\leqslant m$ 使得 $K_B(\omega,i)<K_B(\omega',i)$,且对于任意 $1\leqslant j<i$ 都有 $K_B(\omega,j)=K_B(\omega',j)$,$K_B(\omega)\preccurlyeq K_B(\omega')$表示 $K_B(\omega)=K_B(\omega')$或 $K_B(\omega)\prec K_B(\omega')$.

给定状态集Ω的最偏好状态集 $P_B(\Omega)$是由Ω中权值最小的状态组成的集合,形式化表示为

$$P_B(\Omega)=\{\omega.\omega\in\Omega,\forall\omega'\in\Omega.K_B(\omega)\preccurlyeq K_B(\omega')\}.$$

B 的最偏好状态集 $P(B)=P_B(2^X)$,B 的权值$K(B)$为其最偏好状态的权值.

例 1:给定两层加权库

$$B=(\{x_1\vee\neg x_2\},\{\langle z_1,\neg x_1\vee x_3,20\rangle,\langle z_2,x_2\vee\neg x_3,15\rangle\},\{\langle z_3,\neg x_1\vee\neg x_2,1\rangle,\langle z_4,\neg x_1\vee\neg x_3,1\rangle,\langle z_5,x_2\vee x_3,3\rangle\}).$$

给定软约束集 S,本文使用 $Z(S)$表示 S 中所有 ID 变量的集合.

表 1 列举了 B 上所有状态的权值.显然,$P(B)=\{\omega_8\}$,$K(B)=(0,2)$.

Table 1 Weights of all worlds on the stratified weighted base in Example 1

表 1 例 1 中分层加权库状态的权值

ω	$\omega\vDash H$?	$Z(V_B(\omega))$	$K_B(\omega)$
$\omega_1=\{\neg x_1,\neg x_2,\neg x_3\}$	√	$\{z_5\}$	(0,3)
$\omega_2=\{\neg x_1,\neg x_2,x_3\}$	√	$\{z_2\}$	(15,0)
$\omega_3=\{\neg x_1,x_2,\neg x_3\}$	×	$\{\cdot\}$	$(+\infty,+\infty)$
$\omega_4=\{\neg x_1,x_2,x_3\}$	×	$\{\cdot\}$	$(+\infty,+\infty)$
$\omega_5=\{x_1,\neg x_2,\neg x_3\}$	√	$\{z_1,z_5\}$	(20,3)
$\omega_6=\{x_1,\neg x_2,\neg x_3\}$	√	$\{z_2,z_4\}$	(15,1)
$\omega_7=\{x_1,x_2,\neg x_3\}$	√	$\{z_1,z_3\}$	(20,1)
$\omega_8=\{x_1,x_2,x_3\}$	√	$\{z_3,z_4\}$	(0,2)

定义 6. 在分层加权库 B 下,公式φ蕴含φ'(记为$\varphi|\sim_B\varphi'$,当φ=true 时,简单记为$|\sim_B\varphi'$)当且仅当 $P(B)$中满足φ的状态都能满足φ'.

对于例 1 中的分层加权库有$\neg x_1|\sim_B\neg x_2$,由定义 5 和定义 6 可知,对于给定分层加权库 B,模型检测问题等价于判断给定状态是否属于 $P(B)$,子句推理问题等价于判断给定子句在 B 下是否被 true 蕴含.子句推理的重要性在于,任意公式φ都能等价地表示为若干子句的合取 $C_1\wedge\ldots\wedge C_n$,$|\sim_B\varphi$当且仅当$|\sim_B C_1,\ldots,|\sim_B C_n$同时成立.

3 使用 DM 方法编译分层加权库

惩罚逻辑与基于相干性的非单调推理方法之间关系紧密,字典序策略下,怀疑推理(skeptical inference under lexicographic policy)可直接转化为惩罚逻辑下子句推理问题[9,13,16].文献[13]提出了一种能将字典序策略下进行怀疑解释的分层信念库编译为 ROBDD 的方法.Darwiche 和 Marquis 在文献[9]中扩展了该方法,提出了一个能将任意非分层加权库编译为强等价(文献[9]中称为等价,本文为了区别之后的弱等价概念,称其为强等价)L-规范库(L-normal base)的框架,其中,L 为命题逻辑的任意完备子集.我们首先将强等价和 L-规范库的定义扩展到分层加权库.

定义 7. 给定分层库 B,B'和变量集 X,若如下条件成立,则称 B X-强等价于 B':对于任意 X 上的公式φ和φ',若$\varphi|\sim_B\varphi'$,则$\varphi|\sim_{B'}\varphi'$;反之亦然.若对于任意变量集 X 都有 B X-强等价于 B',则称 B 强等价于 B'.

定义 8. 分层加权库 $B=(H,S_1,\ldots,S_m)$为规范形式当且仅当 $S_1\cup\ldots\cup S_m$ 中任意公式为文字,且对它们进行合取得到的项一致.给定命题逻辑下的完备子集 L,若 B 规范且 H 仅含一个 L-公式,则称 B 为 L-规范库.

命题 1. 对于任意分层加权库 $B=(H,S_1,\ldots,S_m)$,定义其对应的单层加权库 $B^{\downarrow}=(H,S_1'\cup....\cup S_m')$,其中,

$$S_i'=\begin{cases}S_m, & i=m\\ \left\{\left\langle z,\varphi,\kappa\cdot\left(1+\sum_{i<j\leqslant m}\sum_{\langle z',\varphi',\kappa'\rangle\in S_j'}\kappa'\right)\right\rangle:\langle z,\varphi,\kappa\rangle\in S_i\right\}, & 1\leqslant i<m\end{cases}.$$

B 与 $B{\downarrow}$强等价.

证明:对于任意软约束集 S,令 $K_{sum}(S)=\sum_{\langle z,\varphi,\kappa\rangle\in S}\kappa$.由于$\prec$为严格全序关系且 B 与 $B{\downarrow}$的硬约束部分相同,只需证明:任意 H 的模型ω和ω',$K_B(\omega)\prec K_B(\omega')$当且仅当 $K_{B{\downarrow}}(\omega)<K_{B{\downarrow}}(\omega')$.显然,$Z(V_B(\omega))=Z(V_{B{\downarrow}}(\omega))$.当 $K_B(\omega)\prec K_B(\omega')$时,设 $1\leqslant i\leqslant m$ 使得 $K_B(\omega,i)<K_B(\omega',i)$,且对于任意 $1\leqslant j<i$ 都有 $K_B(\omega,j)=K_B(\omega',j)$,因而,

$$\begin{aligned}K_{B{\downarrow}}(\omega')-K_{B{\downarrow}}(\omega)=&(K_B(\omega',i)-K_B(\omega,i))\cdot(1+\textstyle\sum_{i<j\leqslant m}K_{sum}(S_{j+1}'))+\ldots+\\&(K_B(\omega',m-1)-K_B(\omega,m-1))\cdot(1+\textstyle\sum_{m-1<j\leqslant m}K_{sum}(S_m'))+\\&(K_B(\omega',m)-K_B(\omega,m)).\end{aligned}$$

因为该等式右半部分除去第 1 项外其他项之和不大于 $\sum_{i<j\leqslant m}K_{sum}(S_{j+1}')$,所以 $K_{B{\downarrow}}(\omega)<K_{B{\downarrow}}(\omega')$.

对于 $1\leqslant i\leqslant m$,令 $B_i=(H,S_i'),B_i'=(H,\bigcup_{i\leqslant j\leqslant m}S_j')$,若 $K_{B_i}(\omega)<K_{B_i}(\omega')$,必然有 $K_{B_i'}(\omega)<K_{B_i'}(\omega')$.

当 $K_{B{\downarrow}}(\omega)<K_{B{\downarrow}}(\omega')$时,根据 $B{\downarrow}$中软约束权值的定义,必然存在 $1\leqslant i\leqslant m$ 使得 $K_{B_i}(\omega)<K_{B_i}(\omega')$,且对于任意的 $1\leqslant j<i$ 都有 $K_{B_j}(\omega)=K_{B_j}(\omega')$,因此有 $K_B(\omega,i)<K_B(\omega',i)$,且对于任意 $1\leqslant j<i$ 都有 $K_B(\omega,j)=K_B(\omega',j)$,即 $K_B(\omega)\prec K_B(\omega')$.

综上所述,结论成立. □

定理 1. 给定任意变量集 X 上的分层信念库 $B=(H,S_1,\ldots,S_m)$,定义 B 对应的规范库 $B{\downarrow}=(H',S_1',\ldots,S_m')$,其中,$H'=H\cup\bigcup_{1\leqslant i\leqslant m}\{z\rightarrow\varphi:\langle z,\varphi,\kappa\rangle\in S_i\}$.对于任意的 $1\leqslant i\leqslant m$,有 $S_i'=\{\langle z',z,\kappa\rangle:\langle z,\varphi,\kappa\rangle\in S_i,z'$为新引入的唯一 ID 变量$\}$,$B$ 与 $B{\downarrow}$ X-强等价.

证明:首先,显然有$(B{\downarrow}){\downarrow}$与$(B{\downarrow}){\downarrow}$相同,根据文献[9]中的命题 4.1(第 97 页),$B{\downarrow}$与$(B{\downarrow}){\downarrow}$ X-强等价;再根据命题 1 及强等价的传递性,结论成立. □

由上述定理可知,DM 框架也能用来编译分层加权库:给定分层加权库 B 和命题逻辑的完备子集 L,第 1 步,得到规范库 $B{\downarrow}$;第 2 步,将 $B{\downarrow}$中硬约束集编译为 L 公式.文献[13]对应为 DM 框架下的 ROBDD-编译,也即在第 2 步中将硬约束集编译为 ROBDD,假定其根节点为 v.在在线查询阶段,首先对以 v 为根的 ROBDD 根据软约束权值调用算法 Remove-Fat,得到另一个等价于 $P(B)$的 ROBDD.显然,该 ROBDD 支持线性时间的模型检测和子句查询应答.由于实际应用中 ROBDD 的规模与变量顺序密切相关,文献[13]主张第 2 步中得到的 ROBDD 的变量顺序$\lessdot$应满足任意约束$\langle z,\varphi,k\rangle$的 ID 变量 z 出现在φ中所有变量之后,且与φ中在$\lessdot$下的最大变量距离最近.我们称该变量序为 CLS 变量序.文献[13]的作者通过实验验证了在 CLS 变量序下能够产生较小规模的 ROBDD.

例 1(续):根据 DM 框架,首先,将 B 转化为 $B{\downarrow}=(H,S_1,S_2)$,其中,

- $H=\{x_1\vee\neg x_2,z_1\rightarrow\neg x_1\vee x_3,z_2\rightarrow x_2\vee\neg x_3,z_3\rightarrow\neg x_1\vee\neg x_2,z_4\rightarrow\neg x_1\vee\neg x_3,z_5\rightarrow x_2\vee x_3\}$;
- $S_1'=\{\langle z_1',z_1,20\rangle,\langle z_2',z_2,15\rangle\}$;
- $S_2'=\{\langle z_3',z_3,1\rangle,\langle z_4',z_4,1\rangle,\langle z_5',z_5,3\rangle\}$.

然后,将 H 编译为图 1(a)中的 ROBDD.在查询应答阶段,例如,当需要回答模型检测问题ω_8是否为 B 最偏好状态时,根据 S_1' 和 S_2' 中的软约束权值,对图 1(a)中的 ROBDD 调用 Remove-Fat 算法后,得到与 $x_1\wedge x_2\wedge x_3$ 等价的 ROBDD(假定其根为 v).显然,$\omega_8\models\phi(v)$,可知ω_8为最偏好状态.

完成 DM 框架下的第 1 步后,得到的硬约束集与如下两种因素无关:分层加权库的软约束权值;分层加权库如何进行分层.因此,当分层加权库重新分层或其软约束权值发生变化时,只需对编译结果中的软约束部分做出相应改变即可.也即重新分层或改变权值,硬约束部分无需重新编译.DM 方法的缺点在于编译时间花费较高,编译结果的规模较大(由于我们的方法和 DM 方法的编译结果中的软约束部分相同,下文将使用唯一的硬约束的规模表示编译结果的规模),因而也影响了在线查询效率.

重新考虑例 1 中的分层加权库,由表 1 和图 1(a)可知,DM 方法的编译结果包含了所有满足 B 中全部硬约束

的状态信息,或者说每个满足 B 中全部硬约束的状态都能被扩展为满足编译结果中硬约束的状态,我们将满足 B 中全部硬约束的状态分为 4 类:

1) ω_8:当前最偏好状态;
2) ω_1:当权值变化时可能成为最偏好状态;
3) ω_2,ω_7:当权值变化或重新分层时可能成为最偏好状态;
4) ω_3,ω_6:无论权值变化还是重新分层都不可能成为最偏好状态.

我们可以验证:若分层信念库不重新分层,编译结果中是否包含第 3)类信息和第 4)类信息不会影响到最后模型检测和子句推理的结果;即使分层库进行重新分层,编译结果中是否包含第 4)类也不会影响到最后模型检测和子句推理的结果.例如,表 1 中的状态 ω_1 仅违反了 ID 变量为 z_5 的软约束,而 ω_5 违反了 ID 变量为 z_1 和 z_5 的软约束,因此,无论 B 中软约束的权值如何变化或者重新分层,始终有 $K_B(\omega_1)\prec K_B(\omega_5)$.我们将这种情况称为 DM 方法的编译结果包含冗余信息,第 3)类信息称为弱冗余信息,第 4)类信息称为强冗余信息.DM 方法正是由于存在较多的冗余信息而导致编译结果规模较大.同时,编译过程耗费了较多时间.

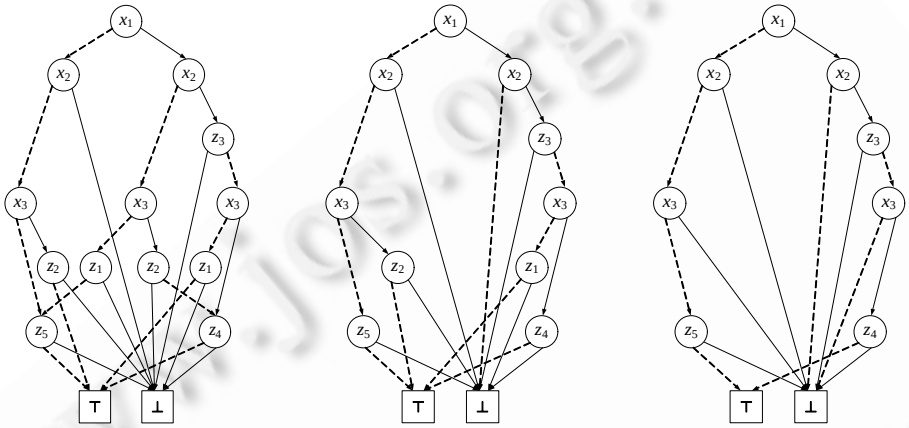


Fig.1 Three ROBDDs are over the variable order $x_1\prec x_2\prec z_3\prec x_3\prec z_1\prec z_2\prec z_4\prec z_5$

图 1 3 个变量序 $x_1\prec x_2\prec z_3\prec x_3\prec z_1\prec z_2\prec z_4\prec z_5$ 下的 ROBDD

4 新的编译方法

考虑到对于实际应用中的分层加权库,相对于软约束权值发生变化的可能性而言,重新分层的可能性相对较小.例如,不同的人对于衣服尺寸和颜色的要求不尽相同,但是对大多数人而言,尺寸的重要性都明显高于颜色.本文方法的主要思想是,在编译过程中去除所有冗余信息,从而减小编译结果的规模,节约编译时间.另外,我们的方法也能通过与命题 1 结合而仅删除强冗余信息,使得即使重新分层也无需重新编译.给定硬约束集 H 和状态集 Ω,我们记 Ω 中满足 H 的状态集 $\Omega^H=\{\omega\in\Omega:\omega\models H\}$.首先给出如下定义:

定义 9. 给定任意分层加权库 $B=(H,S_1,\ldots,S_m)$ 以及 $S_1\cup\ldots\cup S_m$ 的两个子集 S 和 S',若存在 $1\leqslant i\leqslant m$ 使得 $S\cap S_i\subset S'\cap S_i$,且对于任意 $1\leqslant j<i$ 都有 $S\cap S_j=S'\cap S_j$,则称 $S\sqsubset S'$.$S\sqsubseteq S'$ 表示 $S=S'$ 或 $S\sqsubset S'$,$S\sqsubseteq S'$ 不成立,记为 $S\not\sqsubseteq S'$.

定义 10. 给定变量集 X 上的分层加权库 $B=(H,S_1,\ldots,S_m)$,任意状态集 Ω 中的状态 ω 冗余当且仅当满足如下条件:$\omega\in\Omega^H$;存在状态 $\omega'\in\Omega^H$,使得 $V_B(\omega')\sqsubset V_B(\omega)$.给定冗余状态 ω,ω 中所有文字否定的析取(即 $\vee\{\neg l:l\in\omega\}$)称为冗余约束.$\Omega$ 相对于 B 的所有冗余约束的集合记为 $R_B(\Omega)$;当 $\Omega=2^X$ 时,简单记为 $R(B)$.

定义 11. 给定分层加权库 B,B' 和变量集 X,若如下条件成立,则称 B 与 B' X-弱等价:对于任意公式 φ,$|\sim_B\varphi$ 当且仅当 $|\sim_{B'}\varphi$.若对于任意变量集 X 都有 B X-弱等价于 B',则称 B 弱等价于 B'.

显然,弱等价关系具有自反、对称和传递性.若 B 与 B' X-强等价,则 B 必然 X-弱等价于 B';反之不成立.

定理 2. 对于任意分层加权库 $B=(H,S_1,\ldots,S_m)$和公式集 $H'\models H$,对于如下命题:

1) $H\cup R(B)\models H'$成立;

2) 无论 B 中软约束的权值如何变化,$(H',S_1,\ldots,S_m)$始终与 B 弱等价,

则有:命题 1)是命题 2)的充分条件;当 H 一致时,命题 1)为命题 2)的充要条件.

证明:将$(H',S_1,\ldots,S_m)$记为 B',分两个方向进行证明:

($\Rightarrow$)因为 $R(B)$与 B 中软约束的权值无关,故只需证明 $P(B)=P(B')$.我们使用归纳法进行证明.当 $H'\equiv H$ 时,显然成立.假设满足 H'的状态数比满足 H 的数目少 n 时成立.当满足 H'的状态数比满足 H 的数目少 n+1 时,不妨设$\omega\not\models H'$且$\omega\models H$.

令 $H''=H\cup R(\omega),B''=(H'',S_1,\ldots,S_m)$.显然,$H\cup R(B'')\equiv H\cup R(B)$,因而有 $H\cup R(B'')\models H'\models H''$和 $H\cup R(B)\models H''\models H$ 成立.又知满足 H''的状态数比满足 H'的状态数多 n、比满足 H 的状态少 1,根据归纳假设以及弱等价的传递性,无论 B 中软约束的权值如何变化,都有 $P(B)=P(B')$.

($\Leftarrow$)使用反证法进行证明.假设存在ω满足$\omega\models H\cup R(B)$且$\omega\not\models H'$,变换 B 中的软约束权值,使得 B'与之不弱等价.令 $V_B(\omega)$中的软约束权值为$|S_1|+\ldots+|S_m|+1$,其他软约束的权值为 1.若存在ω'使得 $K_B(\omega')\prec K_B(\omega)$,必然有 $V_B(\omega')\sqsubset V_B(\omega)$,与$\omega\models H\cup R(B)$矛盾.显然有,$\omega\in P(B)$.又因为 H 一致,必然有$\omega\notin P(B')$.即 B 不与 B'弱等价.

综上,结论成立. □

给定任意变量集 X 上的分层加权库 $B=(H,S_1,\ldots,S_m)$,我们定义 $\boldsymbol{B}(B)=\{(H',S_1,\ldots,S_m):H\cup R(B)\models H'\models H\}$以及 $\boldsymbol{B}(B)$上的关系$\triangleright=\{((H',S_1,\ldots,S_m),(H'',S_1,\ldots,S_m)):H'\models H''\}$.若将 $\boldsymbol{B}(B)$中硬约束集等价的分层加权库视为相同元素,则$\triangleright$为半序关系且$(H\cup R(B),S_1,\ldots,S_m)$为最小元素,$B$ 为最大元素.根据定理 2,给定变量集 X 上的分层加权库 B,任意 X-强等价于 $\boldsymbol{B}(B)$中元素的分层加权库 B'都能满足模型检测和子句推理这两项任务的要求,也即:1) 给定 2^X 中任意状态$\omega\in P(B)$当且仅当$\omega\in P(B')$;2) 给定任意 X 上的公式φ,$|\sim_B\varphi$当且仅当$|\sim_{B'}\varphi$.显然,DM 框架选择了将 $\boldsymbol{B}(B)$中的最大元素编译为 X-强等价的分层加权库.本文为了去除所有冗余信息,将选择把 $\boldsymbol{B}(B)$中的最小元素编译为 X-强等价的分层加权库,本文方法的关键部分在于如何计算 $R(B)$.

给定变量集 X 上的分层信念库 $B=(H,S_1,\ldots,S_m)$和状态集Ω,我们计算 $R(B)$的方式为:首先将Ω赋值为空集,然后一步步往Ω中增加状态并计算 $R_B(\Omega)$,直至$\Omega=2^X$,因而需要判断状态集Ω中的状态ω是否冗余.我们将 $V_B(\Omega^H)$在半序关系$\sqsubseteq$下的所有极小元素组成的集合记为 $V_B(\Omega^H)^{\min}$(当$\Omega=2^X$,简单记为 $V(B)^{\min}$).由定义 9 可知如下结论成立:1) 任意状态$\omega\in\Omega^H$ 不冗余当且仅当 $V_W(\omega)\in V_B(\Omega^H)^{\min}$;2) 任意状态$\omega\notin\Omega$在$(\Omega\cup\{\omega\})^H$ 中不冗余当且仅当 $V_B(\Omega)^{\min}$ 中不存在 $V_B(\omega)$的严格子集,此时,$V_B((\Omega\cup\{\omega\})^H)^{\min}=\{S\in V_B(\Omega^H)^{\min}:S$ 非 $V_B(\omega)$超集$\}\cup\{V_B(\omega)\}$.

在给出我们的算法之前,首先对软约束集 S 引入如下表示:$Z_f(S)$表示 S 中对应公式为 false 的软约束的 ID 变量集,即 $Z_f(S)=\{z:\langle z,\text{false},k\rangle\in S\}$;$S^\sigma$表示 S 中所有非 false 的公式集,即 $S^\sigma=\{\varphi:\langle z,\varphi,k\rangle\in S$ 且 $z\notin Z_f(S)\}$;τ为将 S 中所有非 false 的公式映射为 true 的替换,即 $S^\tau=\{\langle z,\text{false},k\rangle\in S:z\in Z_f(S)\}\cup\{\langle z,\text{true},k\rangle:\langle z,\varphi,k\rangle\in S$ 且 $z\notin Z_f(S)\}$.我们编译算法(如图 2 所示)的主要思想如下:对于给定的分层加权库 B 和命题逻辑的完备子集 L,若 B 的硬约束集不一致,则将 $B\downarrow$中所有硬约束替换为 true 后返回 $B\downarrow$;否则,首先调用函数 FIND-MIN 计算 $V(B)^{\min}$,然后调用 COMPUTE-R 计算表示 $H\cup R(B)$的 ROBDD,最后将 B 中硬约束替换为$\{\phi(v)\}$并将其编译为 L-规范库.需要指出的是,FIND-MIN 和 COMPUTE-R 都基于 DPLL 程序,因而启发式、二元约束传播(binary constraint propagation)以及文献[21]提出的 CNF 缓存方案等技术都能引入到二者中加速求解.另外,我们将在下一节中给出两种新技术进一步加速 $V(B)^{\min}$ 的计算.

命题 2. 给定任意的分层加权库 $B=(H,S_1,\ldots,S_m)$,$FIND\text{-}MIN(B,\varnothing)=V(B)^{\min}$.

证明:FIND-MIN 基于 DPLL 算法,搜索过程为树状结构.FIND-MIN 的搜索树中,任意终止节点都对应一个状态集,不妨按遍历的先后顺序设这些状态集分别为$\Omega_1,\ldots,\Omega_j,\ldots,\Omega_k,\ldots,\Omega_n$.假定 B 基于变量集 X,显然,上述状态集为 2^X 的划分.搜索树中,终止节点分为两类:失败的节点和成功的节点.假设Ω_j 对应失败节点,也即对应算法中

的第 6 行或第 8 行:对于前者,因为不满足第 4 行的条件,因而对于任意的$\omega\in\Omega_j$,存在$\omega'\in\Omega_1\cup\ldots\cup\Omega_{j-1}$,使得$V_B(\omega')\sqsubset V_B(\omega)$;对于后者,任意的$\omega\in\Omega_j$都至少违反一条硬约束.因此,在这两种情况下都无需改变 *MIN* 的值.假设Ω_k对应成功节点,此时,B'中所有软约束的公式都为常量,对任意的$\omega\in\Omega_k$都有 $V_B(\omega)=S$.因此,按照第 9 行对 *MIN* 进行更改.最后,当第 4 行中的条件成立时,第 i 层中再有一条约束不被满足,将导致当前状态集中的任意状态都冗余,因而我们将第 i 层中剩余的非常量公式都放入硬约束集中,并将原公式替换为 true.

综上所述,结论成立. □

命题 3. 给定任意的分层加权库 $B=(H,S_1,\ldots,S_m)$,*COMPUTE-R*$(B,V(B)^{\min})$的返回结果 v 满足$\phi(v)\equiv H\cup R(B)$.

证明:算法 COMPUTE-R 与文献[22]中的 ROBDD 编译算法的区别在于,第 17 行中要求保证 $V(B)^{\min}$ 中必然存在 S 的超集,这一约束刚好保证了任意满足$\phi(v)$的状态为 B 的非冗余状态,因此结论成立. □

根据定理 1、定理 2、命题 2、命题 3 以及软等价的传递性,很容易得到如下结论:

定理 3. 对于任意变量集 X 上的分层加权库 B 和命题逻辑的完备子集 L,无论 B 中软约束的权值如何变化,算法 *COMPILE*(B,L)的运行结果始终与 B X-弱等价.

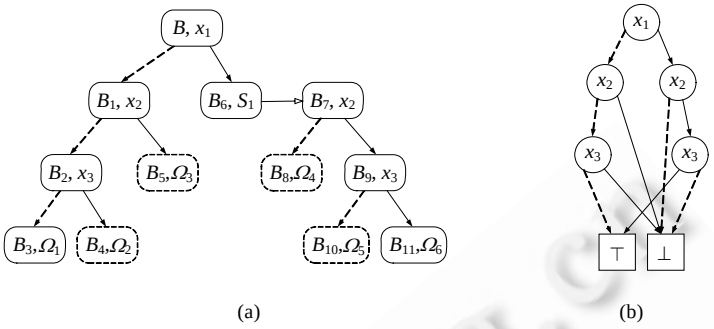
Algorithm *COMPILE*(*B*,*L*).
Input: A stratified weighted base B, and a complete subset L of propositional logic;
Output: An L-normal stratified weighted base which is weakly equivalent to W.
1: **function** *FIND-MIN*(B',*MIN*) //$B'=(H,S_1,\ldots,S_m)$
2: $i\leftarrow\min\{m,\max\{j$: all formulas in $S_1\cup\ldots\cup S_{j-1}$ are constants$\}\}$
3: $S\leftarrow\{\langle z,\varphi,\kappa\rangle\in B:z\in Z_f(S_1\cup\ldots\cup S_i)\}$
4: **if** $S\in\{\{\langle z,\varphi,\kappa\rangle\in S':z\in Z(S_1\cup\ldots\cup S_i)\}:S'\in MIN\}$ **then**
5: **return** *FIND-MIN*$((H\cup S_i^{\sigma},S_1,\ldots,S_{i-1},S_i^{\tau},S_{i+1},\ldots,S_m),MIN)$
6: **else-if** there exists $S'\in MIN$ such that $S'\sqsubseteq S$ **then return** *MIN*
7: **else-if** false$\in H$ or all formulas in $S_1\cup\ldots\cup S_m$ are constants **then**
8: **if** H is inconsistent **then return** *MIN* **end-if**
9: **return** $\{S'\in MIN:S'\not\in S\}\cup\{S\}$
10: **else**
11: Select a literal l whose variable appears in S_i
12: **return** *FIND-MIN*$(B|_{\neg l},$*FIND-MIN*$(B|_l,MIN))$
13: **end-if**
14: **end-function**
15: **function** *COMPUTE-R*(B',*MIN*)
16: $S\leftarrow\{\langle z,\varphi,\kappa\rangle\in B:z\in Z_f(S_1\cup\ldots\cup S_i)\}$
17: **if** false$\in H$ or there exists some superset of S in *MIN* **then return** $\bot$
18: **else-if** all formulas in S are constants **then return** $\top$
19: **else**
20: Create a new node u associated with the minimal variable in W
21: $lo(u)\leftarrow$*COMPUTE-R*$(B'|_{\neg var(u)},MIN)$
22: $hi(u)\leftarrow$*COMPUTE-R*$(B'|_{var(u)},MIN)$
23: **return** *MK*(u)
24: **end-if**
25: **end-function**
26: **if** the hard constraints set of B is inconsistent **then**
27: Displace all the hard constraints of $B\downarrow$ with *true* and then return $B\downarrow$
28: **else**
29: $v\leftarrow$*COMPUTE-R*$(B,$*MINI-CLUSTER*$(B,\varnothing))$
30: Displace all the hard constraints of $B\downarrow$ with $\phi(v)$ and then return $L(B\downarrow)$
31: **end-if**

Fig.2 Description of algorithm COMPILE

图 2 COMPILE 算法描述

例 1(再续):我们调用算法 COMPILE 将 B 编译为 ROBDD-规范库.首先,调用 *FIND-MIN*$(B,\varnothing)$计算 $V(B)^{\min}$,其搜索树如图 3(a)所示.其中,带实心箭头的虚线和实线边分别表示当前变量赋值为 false 和 true;实线框和虚线框的节点$\langle B_j,\Omega_k\rangle$分别表示成功和失败的终止节点,$\Omega_k$表示该节点对应的状态集;带空心箭头的边表示将 B_6 第 1

层中非 false 软约束中的公式转化为硬约束得到 B_7.*FIND-MIN*(B,∅)的返回结果为{{z_3},{z_4,z_5}}.其次,我们调用 COMPUTE-R 计算表示 B 中硬约束并上 $R(B)$的 ROBDD,如图 3(b)所示.最后,与 B 弱等价的 ROBDD-规范库中硬约束部分对应的编译结果如图 1(c)所示.



(a) (b)

Fig.3 Compilation process on the stratified belief base in Example 1

图 3 例 1 中分层库的编译过程

由例 1 可知,使用本文方法编译后的结果其规模明显小于使用 DM 方法的规模.实际上,对于某些问题,本文方法可节省指数次的空间代价.

例 2:给定分层加权库 B=({·},S),其中,S={⟨z_1,x_1∨(¬x_1∧…∧¬x_n),1⟩,…,⟨z_n,x_n∨(¬x_1∧…∧¬x_n),1⟩},我们使用 DM 方法和本文方法分别将 B 编译为 ROBDD-规范库.显然,能满足 B 中所有约束的状态包括{x_1,…,x_n}和{¬x_1,…,¬x_n},也即 $P(B)$={{x_1,…,x_n},{¬x_1,…,¬x_n}},$V(B)^{\min}$={∅}.在使用本文方法得到的 ROBDD-规范库中,硬约束集等价于(x_1∧…∧x_n)∨(¬x_1∧…∧¬x_n).当 n=1 时,表示其 ROBDD 的节点数为 1;否则,节点数为 2n+1.按照 DM 框架下的 ROBDD-编译,也即文献[13]中的工作,B↓中的硬约束集为

$$H=\{z_1\rightarrow(x_1\vee(\neg x_1\wedge\ldots\wedge\neg x_n)),\ldots,z_n\rightarrow(x_1\vee(\neg x_1\wedge\ldots\wedge\neg x_n))\}.$$

在 CLS 变量序下,{x_1,…,x_n}中的变量比{z_1,…,z_n}中的变量先出现,与 H 等价的 ROBDD 的节点数为 n 的指数形式.

对于给定的分层加权库 B,假定 COMPILE 算法的返回结果为 B',当编译过程中去除的冗余信息中包含弱冗余信息时,我们无法保证当 B 重新分层时,对 B'做出同样的重新分层后,二者弱等价,因而需重新编译.例如,对于例 1 中的分层加权库 B,在编译过程中去除了关于ω_2和ω_7的弱冗余信息,当 B 中的软约束合并为一层时,B'中的软约束也合并为一层后,二者显然弱等价,因而需对合并后的 B 重新编译.对于这种情况,可通过编译 B↓使得编译过程中仅删除强冗余信息,因而,即使 B 重新分层也无需重新编译,代价是增加了时间和空间开销.对例 1 中的分层加权库调用 *COMPILE*(B↓,*ROBDD*)返回结果中硬约束如图 1(b)所示.定理 4 保证了这一做法的正确性.

定理 4. 对于任意变量集 X 上的分层加权库 B 和命题逻辑的完备子集 L,假定算法 *COMPILE*(B↓,L)返回的 L-规范库中的唯一硬约束为φ,无论 B 重新分层或其中的软约束权值发生变化,将 B↓中的硬约束集替换为{φ}后得到的分层加权库始终与 B X-弱等价.

证明:假定将 B 中的硬约束集替换为{φ}后得到的分层加权库为 B'.根据命题 1、定理 3 以及弱等价的传递性,B'与 B X-弱等价.又因为φ与 B 如何分层以及其中的软约束权值大小无关,因此结论成立. □

最后,对于任意分层信念库,将所有信念赋予相同的权值后,将得到一个分层加权库(假定为 B,其硬约束为 H),分层信念库的字典序策略下,怀疑推理等价于 B 下的子句推理,因而本文方法能用来编译字典序策略下进行怀疑解释的分层信念库;分层信念库的包含偏好策略(inclusion-preferred policy)下,怀疑推理[13,16]等价于以 H∪$R(B)$为知识库的推理问题,因而在包含偏好策略下进行怀疑解释的分层信念库,也能使用本文方法进行编译.

5 新编译方法中的优化技术

我们通过实验发现,COMPILE 算法中,调用函数 FIND-MIN 的时间在总编译时间占很大比例,因此在本节给出两种优化技术,进一步提高FIND-MIN的执行效率:第1种属于剪枝策略,第2种使用ROBDD压缩表示MIN加速算法 COMPILE 中的第 4 行、第 6 行和第 9 行的执行.

5.1 扩展GSAT近似计算MIN

首先,很容易对命题 2 进行扩展,得到更一般化的结论.

命题 4. 给定任意的分层加权库 $B=(H,S_1,\ldots,S_m)$,M 为由 B 中所有软约束的子集组成的簇,若 M 满足如下两个条件,则有 $FIND\text{-}MIN(B,M)=V(B)^{\min}$:

1) 对于任意的 $S\in M$,都存在状态 ω 满足 $\omega\models H$,且 $V_B(\omega)=S$;

2) 对于任意 M 的元素 S 和 S',满足 $S\not\subseteq S'$.

给定任意的分层加权库 B,由于在实际应用中约束中的公式通常表示为子句形式,我们根据上述性质扩展 GSAT 算法(如图 4 所示):首先,快速求解 $V(B)^{\min}$ 的一个满足命题 4 中条件的近似 M;然后,在 FIND-MIN 利用 M 进行有效的剪枝,也即在算法 COMPILE 的第 5 行增加硬约束或在第 6 行提前退出递归调用.

Algorithm ***GSAT-SWB*****(*****B*****)**.
Input: A stratified weighted base $B=(H,S_1,\ldots,S_m)$, all formulas of B are in clausal form;
Output: A cluster of the subsets of $S_1\cup\ldots\cup S_m$.
1: $M\leftarrow\varnothing$
2: **while** the number of loops is less than a preset value **do**
3: Generate a random world ω
4: **while** true **do**
5: $\Omega\leftarrow\{\omega'$: ω' can be got by flipping one literal in $\omega\}$;
6: **if** $K_B(\Omega)\preccurlyeq K_B(\omega)$ **then break end-if**
7: Let ω be one of worlds in $P_B(\Omega)$
8: **end-while**
9: **if** $\omega\models H$ **then** $M\leftarrow\{S'\in M:S'\not\subseteq S\}\cup\{S\}$ **end-if**
10: **end-while**
11: **return** M

Fig.4 Description of algorithm GSAT-SWB
图 4 GSAT-SWB 算法描述

5.2 使用ROBDD压缩表示MIN

在 COMPILE 算法中,对于给定的分层加权库 B,$V(B)^{\min}$ 中元素的个数通常较多,在最坏情况下,其数目甚至为指数形式.例如,对于 $B=(\varnothing,\{\langle z_1,x_1,1\rangle,\langle z_2,\neg x_1,1\rangle,\ldots,\langle z_{2n-1},x_n,1\rangle,\langle z_{2n},\neg x_n,1\rangle\})$,有$|V(B)^{\min}|=2^n$.因此,对于这类分层加权库,当其规模较大时,一方面,COMPILE 算法中存储 MIN 的空间代价不可承受;另一方法,调用函数 FIND-MIN 的时间代价过高(第 4 行、第 8 行和第 11 行中都需要遍历 MIN 中的所有元素).为了解决上述问题,首先引入如下表示:给定软约束集合 S,定义$\gamma^+(S)=\wedge Z(S)$和$\gamma^-(S)=\wedge\{\neg z:z\in Z(S)\}$;对于 S 的任意子集 S',定义$\gamma(S',S)=\gamma^-(S')\wedge\gamma^+(S\backslash S')$;对于 2^S 的子集 M,$\gamma(M,S)=\vee\{\gamma(S',S):S'\in M\}$.对于给定的分层加权库 $B=(H,S_1,\ldots,S_m)$,我们使用与$\gamma(MIN,S_1\cup\ldots\cup S_m)$等价的 ROBDD 压缩表示 MIN,能大量节省空间开销.对于上一段中的例子,在变量序 $z_1\prec\ldots\prec z_{2n}$ 下表示$\gamma(V(B)^{\min},S)$的 ROBDD 的节点数仅为 $4n+1$.

命题 5. 给定变量集合 X 上的分层加权库 $B=(H,S_1,\ldots,S_m)$及 $S_1\cup\ldots\cup S_m$ 的子集集合 M,令$\varphi=\gamma(M,S_1\cup\ldots\cup S_m)$,有如下结论成立:

1) 给定 $1\leqslant i\leqslant m$ 和 $S\subseteq S_1\cup\ldots\cup S_i$,$S\in\{S'\cap(S_1\cup\ldots\cup S_i):S'\in M\}$当且仅当$\varphi\not\models\neg\gamma(S,S_1\cup\ldots\cup S_i)$;

2) 给定 $S\subseteq S_1\cup\ldots\cup S_m$,存在 $S'\in M$ 使得 $S'\sqsubseteq S$,当且仅当存在 $1\leqslant i\leqslant m$ 使得$\varphi\not\models\neg\gamma(S,S_1\cup\ldots\cup S_{i-1})\vee\neg\gamma^+(S_i\backslash S)$;

3) 给定 $S\subseteq S_1\cup\ldots\cup S_m$,$\gamma(\{S'\in M:S'\not\subseteq S\},S_1\cup\ldots\cup S_m)\equiv\varphi\wedge\neg\gamma^-(S)$;

4) 给定 $S\subseteq S_1\cup\ldots\cup S_m$,$\gamma(M\cup\{S\},S_1\cup\ldots\cup S_m)\equiv\varphi\vee\gamma(S,S_1\cup\ldots\cup S_m)$.

证明:结论 1)和结论 4)显然成立;结论 2)根据定义 8 很容易证明;对于结论 3),有

$$\gamma(\{S'\in M:S'\not\subseteq S\},S_1\cup\ldots\cup S_m)\equiv\varphi\wedge\neg\gamma(\{S'\in M:S'\sqsubseteq S\},S_1\cup\ldots\cup S_m)\equiv\varphi\wedge\neg(\varphi\wedge\gamma^{-}(S))\equiv\varphi\wedge\neg\gamma^{-}(S).$$

证毕. □

根据上述性质,COMPILE 算法中的第 4 行、第 6 行分别等价于判断$\gamma(MIN,S_1\cup\ldots\cup S_m)$是否蕴含单个或多个子句.由于$\gamma(MIN,S_1\cup\ldots\cup S_m)$被表示为 ROBDD,子句蕴含操作能在线性时间(以 ROBDD 中的节点数为参数)内完成[12,15,20].COMPILE 算法中的第 9 行以及 GSAT-SWB 中的第 9 行都能转化为$\gamma(MIN,S_1\cup\ldots\cup S_m)$合取子句$\neg\gamma^{-}(S)$后再与项$\gamma(S,S_1\cup\ldots\cup S_m)$进行析取.假定表示$\gamma(MIN,S_1\cup\ldots\cup S_m)$的 ROBDD 的根为 v,根据已有的工作[12,15,20],$\phi(v)\wedge\neg\gamma^{-}(S)$和$\phi(v)\vee\gamma(S,S_1\cup\ldots\cup S_m)$的时间复杂度分别为 $O(|v|\cdot|S|)$和 $O(|v|\cdot|S_1\cup\ldots\cup S_m|)$.本文根据任意 MIN 的元素 S 和 S'满足 $S\not\subseteq S'$这一性质,分别给出两种时间复杂度更低的合取和析取算法(如图 5 和图 6 所示).

Algorithm *CONJOIN*(v,C).
input: An ROBDD rooted at v satisfying that the number of nodes in each path from v to some terminal node is the same, and a clause C such that each variable v in C appears at most once and v appears in the ROBDD;
output: The root of the ROBDD representing $\phi(v)\wedge C$.

```
1:  if G(v)≠empty then return G(v) end-if
2:  if v is non-terminal then return v end-if
3:  Create a new BDD node u associated with var(v)
4:  if ¬var(v)∈C then lo(u)←lo(v)
5:  else lo(u)←CONJOIN(lo(v),C\{var(v)})
6:  end-if
7:  if var(v)∈C then hi(u)←hi(v)
8:  else hi(u)←CONJOIN(hi(v),C\{¬var(v)})
9:  end-if
10: G(v)←MK(u)
11: return G(v)
```

Fig.5 Description of algorithm CONJOIN

图 5 CONJOIN 算法描述

Algorithm *DISJOIN*(v,T).
input: An ROBDD rooted at v and a consistent term T satisfy that the variables on each path from v to some terminal node exactly appear in T;
output: The root of the ROBDD representing $\phi(v)\vee T$.

```
1:  Create a stack path
2:  while v is non-terminal do
3:      if var(v)∈T then v←hi(v); T←T\{var(v)}
4:      else v←lo(v); T←T\{¬var(v)}
5:      end-if
6:      Push v into path
7:  end-while
8:  Let u be the root of the ROBDD representing T
9:  while path is not empty do
10:     Assign v as top(path) and then pop path
11:     if hi(top(path))=v then u←⟨var(top(path)),lo(top(path)),u⟩
12:     else u←⟨var(top(path)),u,hi(top(path))⟩
13:     end-if
14:     u←MK(u)
15: end-while
16: return u
```

Fig.6 Description of algorithm DISJOIN

图 6 DISJOIN 算法描述

命题 6. 给定根为 v 的 ROBDD 和子句 C,任意从 v 到终止节点的路径上出现的变量都相同,且 C 中出现的变量皆为 ROBDD 中的变量,CONJOIN 算法可在 $O(|v|+|C|)$时间内计算出表示$\phi(v)\wedge C$ 的 ROBDD.

证明:哈希表 G 保证了每个非终止节点最多遍历一次,而从 v 到终止节点的路径上出现的变量都相同这一条件,保证了连接至终止节点的非终止节点最多为 2 个且不同,因而任意终止节点最多遍历 2 次.我们将 C 表示为布尔向量后,CONJOIN 算法的任意行都能在常数时间内终止,因而 CONJOIN 算法可在 $O(|v|+|C|)$时间内终止.下面通过归纳证明其结果等价于$\phi(v)\wedge C$:

当$|v|=1$ 时,结论显然成立.假设当$|v|\leqslant n$ 结论成立.当$|v|=n+1$ 时,令 u 为表示$\phi(v)\wedge C$ 的 ROBDD 的根节点.

- 若$\neg var(v)\in C$,$\phi(lo(u))\equiv\phi(u)|\neg var(v)\equiv\phi(v)|\neg var(v)\equiv\phi(lo(v))$,也即 $lo(u)=lo(v)$;
- 否则,$\phi(lo(u))\equiv\phi(u)|\neg var(v)\equiv(\phi(v)|\neg var(v))\wedge C\backslash\{var(v)\}\equiv\phi(lo(v))\wedge C\backslash\{var(v)\}$.

根据归纳假设,$lo(u)=CONJOIN(lo(v),C\backslash\{var(v)\})$.因此,第 4 行和第 5 行能够正确计算 u 的左子节点.同理,第 7 行和第 8 行能够正确计算 u 的右子节点.第 10 行保证了结果必然为 ROBDD.因此结论成立. □

命题 7. 给定根为 v 的 ROBDD 和项 T,任意从 v 到终止节点的路径上出现的变量都与 T 中的变量相同,DISJOIN 算法可在 $O(|T|)$时间内计算出表示$\phi(v)\vee T$ 的 ROBDD.

证明:算法沿着从根到终止节点再回到根的路径进行遍历,遍历的节点数不多于 $2\cdot|T|+2$,将 T 表示为布尔向量后,任意行都能在常数时间内终止,因而 DISJOIN 算法的时间复杂度为 $O(|T|)$.执行算法后得到的 ROBDD 表示的公式显然等价于$\phi(v)\vee T$. □

6 实验结果

在本节中,我们选择 ROBDD 为目标语言对本文方法和 DM 方法的编译时间花费以及编译结果大小进行比较.我们实现了文献[22]中的 ROBDD 编译算法作为本文实验的 ROBDD 编译器,在编译过程中使用 CLS 变量顺序引导编译.实验平台的 CPU 主频为 2.99GHz,内存大小为 2GB,编程环境为 Microsoft Visual C++ 6.0.

与文献[13,16]中的测试方案一样,我们所有的测试都基于随机 3 文字子句集.由于实验的测试量较大,我们将变量数定义为 30.为了忽略硬约束对编译过程的影响,硬约束集中的子句数被固定为 60.因为对于两种方法而言,软约束的权值都不影响编译结果的规模和编译时间,所以实验中所有软约束的权值都被赋值为随机的正整数.软约束的数目从 20 开始,以 20 为增量,最多为 300.对于 DM 方法,是否分层对测试结果没有影响;对于本文方法,我们将所有软约束平均分为 1 层、2 层或者 3 层(对于不能整分的数目,对于前面的层多分一条约束.例如,当软约束数为 100、层数为 3 时,每层约束数分别为 34,33 和 33).每个测试点的测试次数为 100,为了避免少数极难问题导致曲线的波动,测试结果取中值.当测试时间大于 1 000s 或内存溢出时中断测试.实验结果如图 7 所示,其中,随机问题的变量数为 30,硬约束数固定为 60,x 轴为所有软约束数,y 轴分别为编译结果的节点数和编译时间.层数和编译方法的不同体现在不同的曲线上,每个数据点包含 100 个实例.“*”表示在本文方法中不调用 GSAT-SWB.

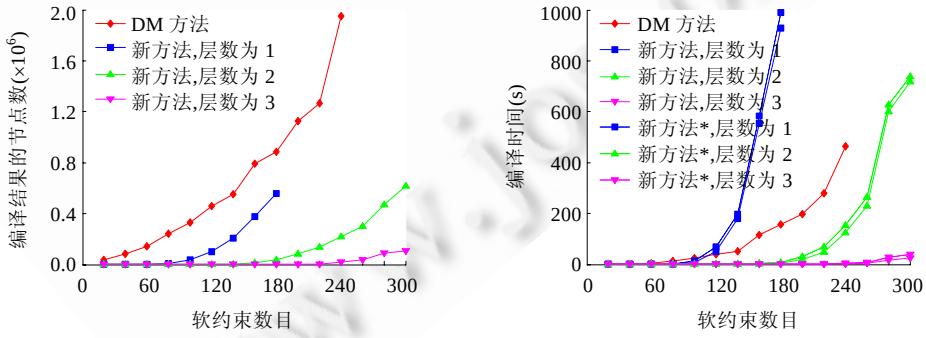


Fig.7 Comparison results on random stratified weighted bases between our approach and DM approach

图 7 本文方法与 DM 方法关于随机问题的对比实验结果

由图 7 中的实验结果可知:

- 无论是否分层,本文方法的编译结果规模都小于 DM 方法;
- 对于我们的方法,当层数越多时,编译结果的规模越小;对于 DM 方法,当软约束的数目大于 240 时,内存溢出;
- 当软约束分为 2 层或 3 层时,本文方法的编译效率高于 DM 方法;
- 若不进行分层,当软约束数目小于等于 100 时,本文方法的效率较高;当软约束数目大于 100 时,DM 方法编译效率较高;当软约束数目大于 180 时,本文方法不能在 1 000s 内完成编译;
- GSAT-SWB 能提高本文算法的编译效率.

7 结 论

本文提出了分层加权库的概念,首先说明了 DM 方法也适用于编译分层加权库,但由于存在较多的"冗余"信息而效率不高.本文给出了一种新的分层加权库编译方法,并基于GSAT和ROBDD提出了两种优化技术提高该方法的时间效率.对于给定的分层加权库 B,若其层数变化不频繁,可直接对 B 进行编译,从而能够删除编译结果中所有的冗余信息;若其层数变化较频繁,可对 $B^{\downarrow}$进行编译,从而仅删除编译结果中的强冗余信息,使得即使重新分层也无需重新编译.本文使用随机问题对该方法进行测试,结果表明:对于非分层加权库,该方法的空间效率高于已存在方法;对于分层加权库,本文方法的时间和空间效率均高于已存在方法,且层数越多时,该方法效率越高.因此在实际应用中,对于层数变化不频繁的分层加权库,可使用本文方法对其进行编译;对于层数变化较频繁的分层加权库,可根据具体的时间和空间代价考虑,选择 DM 方法或本文方法进行编译.本文方法也能用来编译字典序和包含偏好策略下进行怀疑解释的分层信念库.考虑到本文方法或者删除了分层加权库中的所有弱冗余信息,或者保留所有的弱冗余信息,我们猜测,当保留部分弱冗余信息时,可能会使一些情况下的重新分层无需重新编译.

References:

[1] Chevaleyre Y, Endriss U, Lang J, Maudet N. Preference handling in combinatorial domains: From AI to social choice. AI Magazine, 2008,29(4):37−46.

[2] Wilson N. Efficient inference for expressive comparative preference languages. In: Boutilier C, ed. Proc. of the 21st Int'l Joint Conf. on Artificial Intelligence (IJCAI 2009). AAAI Press, 2009. 961−966.

[3] Boutilier C, Brafman RI, Domshlak C, Hoos HH, Poole D. CP-Nets: A tool for representing and reasoning with conditional ceteris paribus preference statements. Journal of Artificial Intelligence Research, 2004,21:135−191. [doi: 10.1613/jair.1234]

[4] Chomicki J. Preference formulas in relational queries. ACM Trans. on Database Systems, 2003,28(4):1−40. [doi: 10.1145/958942.958946]

[5] Pinkas G. Propositional nonmonotonic reasoning and inconsistency in symmetric neural networks. In: Mylopoulos J, Reiter R, eds. Proc. of the 12th Int'l Joint Conf. on Artificial Intelligence (IJCAI'91). San Francisco: Morgan Kaufmann Publishers, 1991. 525−530.

[6] Pinkas G. Reasoning, nonmonotonicity and learning in connectionist networks that capture propositional knowledge. Artificial Intelligence, 1995,77(2):203−247.

[7] Coste-Marquis S, Lang J, Liberatore P, Marquis P. Expressive power and succinctness of propositional languages for preference representation. In: Dubois D, Welty CA, Williams M, eds. Proc. of the 9th Int'l Conf. on Principles of Knowledge Representation and Reasoning (KR 2004). AAAI Press, 2004. 203−212.

[8] de Saint-Cyr FD, Lang J, Schiex T. Penalty logic and its link with Dempster-Shafer theory. In: de Mántaras RL, Poole D, eds. Proc. of the 10th Annual Conf. on Uncertainty in Artificial Intelligence (UAI'94). San Francisco: Morgan Kaufmann Publishers, 1994. 204−211.

[9] Darwiche A, Marquis P. Compiling propositional weighted bases. Artificial Intelligence, 2004,157(1-2):81−113. [doi: 10.1016/j.artint.2004.04.005]

[10] Cadoli M, Donini F. A survey on knowledge compilation. AI Communications, 1997,10(3-4):137−150.

[11] Selman B, Kautz H. Knowledge compilation and theory approximation. Journal of the ACM, 1996,43(2):193−224. [doi: 10.1145/226643.226644]

[12] Darwiche A, Marquis P. A knowledge compilation map. Journal of Artificial Intelligence Research, 2002,17(1):229−264.

[13] Cayrol C, Lagasquie-Schiex M, Schiex T. Nonmonotonic reasoning: From complexity to algorithms. Annals of Mathematics and Artificial Intelligence, 1998,22(3-4):207−236. [doi: 10.1023/A:1018939502485]

[14] Darwiche A. Decomposable negation normal form. Journal of the ACM, 2001,48(4):608−647. [doi: 10.1145/502090.502091]

[15] Bryant RE. Graph-Based algorithms for Boolean function manipulation. IEEE Trans. on Computers, 1986,C-35:677−691. [doi: 10.1109/TC.1986.1676819]

[16] Coste-Marquis S, Marquis P. On stratified belief base compilation. Annals of Mathematics and Artificial Intelligence, 2004,42(4):399−442. [doi: 10.1023/B:AMAI.0000038313.15152.5c]

[17] Benferhat S, Kaci S, Berre DL, Williams M. Weakening conflicting information for iterated revision and knowledge integration. Artificial Intelligence, 2004,153(1-2):339−371. [doi: 10.1016/j.artint.2003.08.003]

[18] Selman B, Levesque HJ, Mitchell DG. A new method for solving hard satisfiability problems. In: Hayes-Roth B, Korf RE, eds. Proc. of the 12th National Conf. on Artificial Intelligence (AAAI'92). AAAI Press/Massachusetts: The MIT Press, 1994. 440−446.

[19] Lin H, Sun JG. Knowledge compilation using the extension rule. Journal of Automated Reasoning, 2004,32(2):93−102. [doi: 10.1023/ B:JARS.0000029959.45572.44]

[20] Andersen H. An introduction to binary decision diagrams. Technical Report, Department of Information Technology, Technical University of Denmark, 1998. http://www.eng.utah.edu/~cs6100/lectures/lec6/anderson-bdd.pdf

[21] Darwiche A. New advances in compiling CNF to decomposable negation normal form. In: de Mántaras RL, Saitta L, eds. Proc. of the 16th Eureopean Conf. on Artificial Intelligence, (ECAI 2004). Amsterdam: IOS Press, 2004. 328−332.

[22] Huang J, Darwiche A. The language of search. Journal of Artificial Intelligence Research, 2007,29:191−219. [doi: 10.1613/jair.2097]

赖永(1985－),男,江西萍乡人,博士生,主要研究领域为知识表示与推理.

刘大有(1942－),男,教授,博士生导师,CCF高级会员,主要研究领域为知识工程与专家系统,不确定性推理,时空推理.