ISSN 1000-9825, CODEN RUXUEW
*Journal of Software*, Vol.20, No.7, July 2009, pp.1714−1725
doi: 10.3724/SP.J.1001.2009.03320

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

# 一种基于扩展规则的#SAT 求解系统*

殷明浩[1,2,3+], 林 海[1,2], 孙吉贵[1,2]

[1](吉林大学 计算机科学与技术学院,吉林 长春 130012)

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

[3](东北师范大学 计算机学院,吉林 长春 130117)

## Solving #SAT Using Extension Rules

YIN Ming-Hao[1,2,3+], LIN Hai[1,2], SUN Ji-Gui[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, Changchun 130012, China)

[3](College of Computer Science, Northeast Normal University, Changchun 130117, China)

+ Corresponding author: E-mail: mhyin@nenu.edu.cn

**Yin MH, Lin H, Sun JG. Solving #SAT using extension rules. *Journal of Software*, 2009,20(7):1714−1725.**
http://www.jos.org.cn/1000-9825/3320.htm

**Abstract**: #SAT problem is the extension of SAT problem. It involves counting models of a given set of proposition formulae. By using the inverse of resolution and the inclusion-exclusion principle to circumvent the problem of space complexity, this paper proposes a framework for both model counting and weighted model counting. It suggests a complementary method for current model counting methods. These methods are proved to be sound and complete. A model counting system, namely JLU-ERWMC, is built based on these methods. JLU-ERWMC outperforms the most efficient model counting methods in some cases.

**Key words**: extension rule; model counting; knowledge compilation; weighted model counting

**摘 要**: #SAT 问题是 SAT 问题的扩展,需要计算出给定命题公式集合的模型个数.通过将问题求解沿着归结的反方向进行,并利用容斥原理解决由此带来的空间复杂性问题,提出了一种基于扩展规则的模型计数和加权模型计数问题求解框架,可以看作是目前所有模型计数问题求解方法的一种补方法.证明了该方法的完备性和有效性,设计了基于扩展规则的#SAT 求解系统:JLU-ERWMC.实验结果表明,JLU-ERWMC 在有些问题中优于目前最为高效的#SAT 问题求解系统.

**关键词**: 扩展规则;模型计数;知识编译;加权模型计数

**中图法分类号**: TP18 **文献标识码**: A

* Supported by the National Natural Science Foundation of China under Grant Nos.60573067, 60773097 (国家自然科学基金); the Specialized Research Fund for the Doctoral Program of Higher Education of China under Grant No.20050183065 (国家高等学校博士学科点专项科研基金); the Science Foundation for Young Teachers of Northeast Normal University of China under Grant No.20070601 (东北师范大学青年基金)

Received 2007-05-31; Accepted 2008-02-20

随着命题可满足问题(SAT)的研究的日益成熟,求解 SAT 问题的能力近年来有了很大程度的提高.目前 Zchaff,Survey Propagation 等高效的 SAT 求解系统已经可以求解含有 100 000 个以上变量的 SAT 问题实例[1−3].另一方面,SAT 领域的成功也使得其他研究领域受益匪浅,例如在经典智能规划研究领域,SATPLAN 和 BLACKBOX 等经典智能规划系统通过将规划问题编译为 SAT 问题实例,借助高效 SAT 求解系统,在近年来的国际智能规划竞赛中屡次取得优异的成绩,获得了多项冠军[4,5].事实上,对于几乎所有的 NP 完备问题,基于 SAT 的求解方法大多效果显著,这是因为它们总存在到 SAT 问题的多项式归约.

然而,在人工智能研究领域存在着很多计算复杂性高于 NP 的问题,例如将概率推理问题[6]、一致性概率规划问题[7]转换为命题公式集合后,仅仅判断给定公式集合是否是可满足的往往是不够的,这时还需要计算出该公式集合中所有模型(可满足赋值)的个数.#SAT 问题即是这样一类问题,它需要计算出给定命题公式集合的模型的个数,其计算复杂度要高于 SAT 问题,是#P 完备的[8].

近年来,#SAT 问题受到研究人员的广泛关注.Birnbaum 和 Lozinskii 证明,只要子问题不含不可满足子句,则可以直接扩展 Davis-Putnam(DP)过程来求解模型计算问题,并提出了基于 DP 的模型计算算法 CDP(counting models using Davis-Putnam procedure)[9].算法 CDP 的基本思想如下:设 $M(\Sigma)$表示给定公式集合$\Sigma$的模型个数,由于子公式$(\Sigma\wedge a)$和$(\Sigma\wedge\neg a)$的模型交集为空,所以 $M(\Sigma)=M(\Sigma\wedge a)+M(\Sigma\wedge\neg a)$.若$\Sigma$中有 $n$ 个变量,其中$\Sigma\wedge a$ 中有 $n_1$ 个变量出现,$\Sigma\wedge\neg a$ 有 $n_2$ 个变量出现,则

$$M(\Sigma)=M(\Sigma\wedge a)\times 2^{n-n_1}+M(\Sigma\wedge\neg a)\times 2^{n-n_2}.$$

Bayardo 和 Pehoushek 利用连接图来描述变量之间的关系:连接图的节点为出现在公式集合中的变量,若两个变量出现在同一子句中,则代表这两个变量的节点之间就存在一条边.通过这种方式可以将$\Sigma$中的所有变量分成多个不相连的组件,进而将$\Sigma$分成多个子公式单独求解.Sang 等人设计的 Cachet 系统在 DP 过程中混合使用了子句学习和组件缓存的策略,通过记录已经出现的冲突和子公式的模型个数来避免重复计算[10,11].在文献[12]中,Sang 等人在 Cachet 的基础上设计了 Cachet-WMC 系统,该系统可以处理加权模型计算问题.Selman 等人提出了一种近似的模型计算和加权模型计算方法,该方法通过采样的方法来估计模型的个数[13].Darwiche 等人则提出可以通过知识编译的方法来求解模型计算问题.通过将给定命题公式转换为 DNNF 范式后,可以在多项式过程内求解模型计算问题[14−16].

无论对于 SAT 问题还是#SAT 问题,归结原理都是最重要也是最基本的推理方法.它的基本思想是通过推出空子句的方法来判定某个给定子句集的不可满足性.我们提出了一种与归结方法对称的 SAT 求解方法——扩展规则方法[17].该方法的主要思想是,沿着归结的逆向求解 SAT 问题,并且借助容斥原理解决由此带来的空间复杂性问题.我们还提出一种基于扩展规则的知识编译方法,可以在多项式时间内求解编译后的 SAT 问题[18].在此基础上,我们将扩展规则方法推广到模态逻辑中,并优化了基于扩展规则的求解方法[19,20].本文的目的在于将扩展规则方法推广到#SAT 问题求解中,并介绍基于扩展规则的#SAT 问题求解器——JLU-ERWMC.

本文第 1 节简单回顾扩展规则.第 2 节介绍基于扩展规则的#SAT 问题求解方法.第 3 节介绍基于扩展规则的加权模型计算方法.第 4 节给出实验结果.最后给出全文的总结和未来的相关工作.本文相关定理的证明详见附录.

## 1 扩展规则

为了讨论方便,先来介绍本文需要的相关概念和命题.首先引入扩展规则,它是本文其他工作的基础.关于扩展规则的详细定义,可以参照文献[17].扩展规则定义如下:

**定义 1**[17]. 给定一个子句 $C$ 和一个原子的集合 $M$:$D=\{C\vee a, C\vee\neg a | a\in M$ 并且 $a$ 和$\neg a$ 都不在 $C$ 中出现$\}$,我们把从 $C$ 到 $D$ 中元素的推导过程叫做扩展规则,$D$ 中的元素叫做应用扩展规则的结果.

例如,给定子句 $A\vee B$ 和集合$\{A,B,C\}$,对子句 $A\vee B$ 应用扩展规则的结果就是 $A\vee B\vee C, A\vee B\vee\neg C$.下面来看一下子句 $C$ 及其扩展后的结果 $D$ 之间的关系.

**定理 1**[17]. 子句 $C$ 及其扩展后的结果 $D$ 是等价的.

这样,我们可以对子句集中的子句应用扩展规则.定理 1 保证了应用扩展规则后的子句集和原子句集等价,因此,扩展规则可以被看作是一条推理规则.

**定义 2**. 一个非重言式子句是集合 $M$ 上的极大项当且仅当它包含集合 $M$ 上的所有原子或其否定.

例如,给定原子集合 $M=\{A,B,C\}$,$A\vee B\vee C$ 就是 $M$ 上的极大项,$A\vee B$ 则不然.

**定理 2**[17]. 给定一个子句集$\Sigma$,它其中所有原子的集合是 $M(|M|=m)$,若$\Sigma$中的子句都是 $M$ 上的极大项,则子句集$\Sigma$不可满足当且仅当$\Sigma$中含有 $2^m$ 个互不相同的子句.

## 2 基于扩展规则的模型计数方法

本节介绍如何利用扩展规则来求解#SAT 问题,包括两部分内容:第 2.1 节介绍如何利用扩展规则直接求解#SAT 问题,第 2.2 节介绍如何通过知识编译将给定子句集合转换成易于求解#SAT 问题的形式.我们首先对本文中一些符号做出约定:用$\Sigma,\Sigma_1,\Sigma_2,\ldots$表示 CNF(conjunctive normal form)形式的子句集.用 $C,C_1, C_2,\ldots$表示单个的子句,有时也把子句理解成文字集合的形式.通常,$M$ 表示一个子句集$\Sigma$中所有原子的集合.

### 2.1 利用扩展规则求解模型计数问题

**定理 3**. 给定一个子句集$\Sigma$,其中所有原子的集合是 $M(|M|=m)$,若$\Sigma$中的子句都是 $M$ 上的极大项,则子句集$\Sigma$的模型个数为 $2^m-S$ 当且仅当$\Sigma$中含有 $S$ 个互不相同的子句,其中 $S\le 2^m$.

定理 3 实际给出了赋值和极大项的对应关系,对于每个赋值,如果它使得子句集为真当且仅当它的“非”形式的极大项不出现在子句集中.这样,如果要计算一个子句集的模型个数,首先要用扩展规则把原来的子句集扩展成与其等价的极大项组成的集合,再根据定理 3:如果扩展后的子句集里面有 $S$ 个互不相同的子句,则原来的子句集就含有 $2^m-S$ 个模型.

**定理 4**. 在命题逻辑中,基于扩展规则的模型计数方法是正确的和完备的.

显然,直接应用扩展规则最坏情况下的空间复杂性是呈指数级的.实际上,我们并不需要把所有的极大项都扩展出来,只需要计算它们的个数就足够了.如果极大项的个数是 $S$,则子句集含有 $2^m-S$ 个模型.包含排斥原理恰好能够帮助我们做到这一点.

**定理 5**[17]. 两个子句扩展出的极大项的集合不含交集当且仅当这两个子句含有互补对.

**定理 6(包含排斥原理)**. 集合 $A_1,A_2,\ldots,A_n$ 并集的元素个数可以用如下公式计算:

$$|A_1\cup A_2\cup\ldots\cup A_n|=\sum_{i=1}^{n}|A_i|-\sum_{1\le i<j\le n}|A_i\cap A_j|+\ldots+(-1)^{n+1}|A_1\cap A_2\cap\ldots\cap A_n|.$$

包含排斥原理的简单形式为大家所熟知:

$$|A\cup B|=|A|+|B|-|A\cap B|,$$

$$|A\cup B\cup C|=|A|+|B|+|C|-|A\cap B|-|A\cap C|-|B\cap C|+|A\cap B\cap C|.$$

我们在文献[17]中已经介绍了如何利用包含容斥原理计算出极大项的个数:给定一个子句集$\Sigma=\{C_1,C_2,\ldots,C_n\}$,$M$ 是其原子的集合且$|M|=m$.记 $P_i$ 为 $C_i$ 扩展出的所有极大项的集合,$i=1,\ldots,n$.令 $S$ 为$\Sigma$能够扩展出的所有极大项的集合的元素个数,则 $S=|P_1\cup P_2\cup\ldots\cup P_n|$,由包含排斥原理容易得到公式(1):

$$S=\sum_{i=1}^{n}|P_i|-\sum_{1\le i<j\le n}|P_i\cap P_j|+\sum_{1\le i<j<l\le n}|P_i\cap P_j\cap P_l|-\ldots+(-1)^{n+1}|P_1\cap P_2\cap\ldots\cap P_n| \quad (1)$$

其中,$|P_i|=2^{m-|C_i|}$.

$$|P_i\cap P_j|=\begin{cases}0, & C_i\text{和}C_j\text{中含有互补文字}\\ 2^{m-|C_i\cup C_j|}, & C_i\text{和}C_j\text{中不含有互补文字}\end{cases}.$$

例 1:利用公式(1)计算子句集$\Sigma=\{\neg A\vee B\vee\neg C,A\vee C,\neg A\}$的模型个数.

$\Sigma$能够扩展出的极大项的数目:$S=2^0+2^1+2^2-0-2^0-0+0=6$,

因此,$\Sigma$中的模型个数为 $2^3-6=2$.

利用公式(1)可以设计出直接利用扩展规则求解模型计数问题的算法,在附录中给出了算法 CER(counting models using extension rules)的具体流程.显然,该算法的效率依赖于公式(1)中需要计算的非零项的数目.一般情况下,算法 CER 的复杂性是呈指数级的,效率比较低.但有些情况下,算法 CER 的效率会很高.考虑子句集中任意两个子句都含有互补对的情况,在这种情况下,只需计算公式(1)的前 $n$ 项就可以判定子句集的可满足性了,因为从第 $n$+1 项开始所有的项都为 0.直观上考虑,这种情况下使用基于归结的方法效率会比较低,因为可能潜在地有很多归结需要做.尽管上面的情况非常特殊,但我们总可以通过知识编译的方法使它得到满足.我们在第 2.2 节中将详细介绍这种方法.

### 2.2 利用知识编译求解模型计数问题

简单地说,知识编译的过程就是把知识转换成易于推理的表示形式的过程,转换的结果被称作是目标语言(target language).在文献[18]中我们介绍了 EPCCL(each pair of clauses contains complementary literal(s))理论.

**定义 3**[18]. EPCCL 理论是其中任意两个子句都含有互补对的子句集.

我们已经证明了 EPCCL 理论是在“可满足可控制类”中的,它的可满足性可以在线性时间内用扩展规则得到判定.

**定理 7**[18]. 对任何一种 EPCCL 理论的可满足性判定可以在多项式时间内完成.

但一种理论是在“可满足性可控制”的类中,并不意味着它也在“模型计数可控制”的类中,下面给出一个例子来说明这个问题.

例 2:假定$\Sigma=\{C_1,C_2,\ldots,C_n\}$是一个难解的模型计数问题,但$\Sigma'=\{C_1\vee L,C_2\vee L,\ldots,C_n\vee L\}$的可满足性非常容易判定:任何含 $L$ 以正的方式出现的解释都是$\Sigma'$的模型.因而,$\Sigma'$在“可满足性可控制”的类中.但计算“$\Sigma'$的模型个数”就等于计算$\Sigma$的模型个数,是非常困难的.因此,$\Sigma'$不在“模型计数可控制”的类中.

定理 8 指出,EPCCL 理论是在“模型计数可控制”类中的.

**定理 8**. 对于一种 EPCCL 理论的模型计算过程可以在多项式时间内完成.

根据定理 8,就可以设计基于知识编译的模型计数算法,附录 B 中给出了算法 KCCER(knowledge compilation for counting models using extension rules)的具体流程.算法 KCCER 在计算模型个数时,首先需要调用算法 KCER(knowledge compilation using extension rules),将给定的命题集合通过知识编译转换为一个等价的 EPCCL 理论.再利用公式(1)计算出给定 EPCCL 理论的模型个数即可.算法 KCER 的核心思想是基于“桶删除(bucket elimination)”的思想.初始时,$\Sigma$中的每一个子句相当于一个桶(bucket),算法依次处理每一个桶,当处理一个桶的时候,其相应的子句被从$\Sigma_1$移到$\Sigma_2$,用扩展规则和删除规则来保证每一个含$\Sigma_1$中的一个子句和$\Sigma_2$中的一个子句的子句对含互补对,这样,$\Sigma_2$中的子句就可以作为编译结果的一部分被从$\Sigma_2$移到$\Sigma_3$中.所有的桶都被处理完之后算法也就终止,$\Sigma_3$作为其输出.需要指出的是,在利用算法 KCER 进行知识编译时,编译后的子句集比编译前的子句集规模要有所扩大,最坏情况下,这种扩大是呈指数级的,算法 KCER 的时间代价在最坏情况下也呈指数级.但由于知识编译实际上属于离线推理,因此,用户实际关心的是在线求解模型计数的时间.而且,知识编译所花费的时间可以通过对同一个知识库的多次询问得到补偿.

**定理 9**. 算法 KCCER 是正确和完备的.

## 3 加权模型计算

在第 2 节中我们介绍了利用扩展规则求解模型计算的一般方法,本节进一步介绍如何使用扩展规则求解加权模型计算问题.

**定义 4**. 设$\Sigma$为任意逻辑子句集合,$\Sigma$中的每个文字 $l$ 有一个权值,记做 $w(l)$,设$\omega$为$\Sigma$的一个模型,则$\omega$的权为$W(\omega)$,被计算为

$$W(\omega)=\prod_{\omega\models l} w(l)\ .$$

$\Sigma$的权为满足$\Sigma$的所有模型的权之和,记作 $WMC(\Sigma)$,被计算为

$$WMC(\Sigma)=\sum_{\omega|=\Sigma}W(\varpi).$$

显然,在第 2 节中讨论的模型计算可以被看作是加权模型计算的特例.这时,$\Sigma$中的每个文字 $l$ 的权值都是0.5.在很多文献中,若某个文字 $l$ 的权为 $w(l)$,则$\neg l$ 的权值为 $1-w(l)$,即 $w(l)+w(\neg l)=1$,本文并不作这种假设.

**命题 10**. 设$\Sigma$为子句集合,如果其中任意两个子句含有互补文字,则对$\Sigma$的任意赋值都不会使这两个子句同时为假.

**命题 11**. 设 $M$ 为原子集合,$M=\{l_1,\ldots,l_m\}$,设$\Pi=\{\Sigma_1,\ \Sigma_2,\ldots,\ \Sigma_n\}$,其中$\Pi$中的每个元素$\Sigma_i$为一个子句集合,$\Sigma_i$包含且只包含 $M$ 中的所有原子;设$\Pi$中权最大的子句集合为$\Sigma_T$,则

(1) $\Sigma_T|=T$;

(2) $WMC(\Sigma_T)=\prod_{i=1}^{m}[w(l_i)+w(\neg l_i)]$.

事实上,如果给定子句集合$\Sigma$,$M=\{l_1,\ldots,l_m\}$为$\Sigma$中出现的所有原子集合,则$\Sigma$的所有可能的赋值的权之和即为 $\prod_{i=1}^{m}[w(l_i)+w(\neg l_i)]$.我们用$\Sigma(l_k)$来表示在$\Sigma$中限定 $l_k$ 取值为真得到的新的子句集合,用 $WMC(\Sigma(l_k))$表示限定 $l_k$ 取值为真后所有可能的赋值的权和,则

$$WMC(\Sigma(l_k))=w(l_k)\times\prod_{i=1}^{k-1}[w(l_i)+w(\neg l_i)]\times\prod_{i=k+1}^{m}[w(l_i)+w(\neg l_i)].$$

类似地,可以计算出$\Sigma(\neg l_k)$,$\Sigma(l_k\wedge l_s)$的值.

对于任意子句 $C$,如果要使该子句为假,则必须使 $C$ 中所有的文字都赋值为假,因此很容易得到如下引理:

**引理 12**. 设$\Sigma$为子句集合,$C_1$ 和 $C_2$ 为任意两个不含任何互补文字的子句,其中,$C_1=l_1\vee\ldots\vee l_r$,$C_2=l'_1\vee\ldots\vee l'_s$,设 $M$ 为$\Sigma$中出现的所有原子集合,$|M|=m$,设 $L=\{l_1,\ldots,l_r\}\cup\{l'_1,\ldots,l'_s\}=\{l''_1,\ldots,l''_t\}$,则

(1) $W(\neg C_1)=WMC(\Sigma(\neg l_1\wedge\neg l_2\wedge\ldots\wedge\neg l_k))$;

(2) $W(\neg C_2)=WMC(\Sigma(\neg l'_1\wedge\neg l'_2\wedge\ldots\wedge\neg l'_s))$;

(3) $W(\neg C_1\wedge\neg C_2)=WMC(\Sigma(\neg l''_1\wedge\neg l''_2\wedge\ldots\wedge\neg l''_t))$.

这里,$W(\neg C_1)$是指所有使 $C_1$ 为假的赋值的权和,$W(\neg C_1\wedge\neg C_2)$表示同时使 $C_1$ 和 $C_2$ 为假的赋值的权和.

下面用一个例子来说明上述命题和引理.

例 3:给定子句集合$\Sigma=\{C_1:\neg p\vee q,C_2:\neg r\vee\neg q,C_3:\neg r\}$,$w(p)=w(\neg p)=0.5$;$w(q)=0.3$,$w(\neg q)=0.7$;$w(r)=0.8$,$w(\neg r)=0.2$.计算 $W(\neg C_1\wedge\neg C_2)$,$W(\neg C_1)$,$W(\neg C_1\wedge\neg C_3)$,$WMC(\neg C_2\wedge\neg C_3)$的值.

由于 $C_1$ 和 $C_2$ 中含有互补文字,因此,

$$W(\neg C_1\wedge\neg C_2)=0.$$

根据引理 11,

$$W(\neg C_1)=WMC(\Sigma(p\wedge\neg q))=w(p)\times w(\neg q)\times(w(r)+w(\neg r))=0.35,$$

$$W(\neg C_1\wedge\neg C_3)=WMC(\Sigma(p\wedge\neg q\wedge r))=0.28,$$

$$WMC(\neg C_2\wedge\neg C_3)=WMC(\Sigma(q\wedge r))=0.24.$$

给定一个子句集$\Sigma=\{C_1,C_2,\ldots,C_n\}$,$M$ 是其原子的集合且$|M|=m$.记 $P_i$ 为 $C_i$ 扩展出的所有极大项的集合,$i=1,\ldots,n$,由于$\Sigma$中所有可能的赋值的权和为 $WMC(\Sigma_T)$,基于扩展规则计算加权模型计数问题的思想就是计算出所有使$\Sigma$为假的模型权和 $WMC(\neg\Sigma)$,这样使$\Sigma$为真的模型权和 $WMC(\Sigma)$即为 $WMC(\Sigma_T)-W(\neg\Sigma)$.计算 $W(\neg\Sigma)$的基本思想类似于算法 CER,利用包含容斥原理,公式(2)给出了计算加权模型的一般方法.利用公式(2)就可以设计计算加权模型计数问题的算法 CWER(counting weighted models using extension rules),其具体流程详见附录 B.

$$WMC(\Sigma)=WMC(\Sigma_T)-W(\neg\Sigma)=\prod_{i=1}^{m}[w(l_i)+w(\neg l_i)]-WMC(\neg\Sigma)\quad(2)$$

其中,

$$WMC(\neg\Sigma)=W(\neg C_1\vee\neg C_2\vee\ldots\vee\neg C_n)$$
$$=\sum_i W(\neg C_i)-\sum_{1\le i<j\le n}W(\neg C_i\wedge\neg C_j)+\sum_{1\le i<j<l\le n}W(\neg C_i\wedge\neg C_j\wedge\neg C_l)+\ldots+$$
$$\sum_{1\le i<j<l\le n}W(\neg C_i\wedge\neg C_j\wedge\ldots\wedge\neg C_n).$$

例 4:考虑例 3 中的子句集,其加权模型数计算如下:

$$WMC(\Sigma)=WMC(\Sigma_T)-[W(\neg C_1)+W(\neg C_2)+W(\neg C_3)]+[W(\neg C_1\wedge\neg C_2)+W(\neg C_2\wedge\neg C_3)+W(\neg C_1\wedge\neg C_3)]-$$
$$W(\neg C_1\wedge\neg C_2\wedge\neg C_3)$$
$$=1-(0.5\times0.7+0.3\times0.8+0.8)+(0+0.5\times0.7\times0.8+0.3\times0.8)-0=0.13.$$

同样,我们可以利用知识编译的方法来求解加权模型计算问题.根据命题 11,当两个子句含有互补对时,不存在使这两个子句同时为假的赋值,因此,当给定一个含有 $n$ 个子句的 EPCCL 理论$\Sigma=\{(l_{11}\vee\ldots\vee l_{1n_1}),\ldots,(l_{n1}\vee\ldots\vee l_{nn_n})\}$时,公式(2)被简化为

$$WMC(\Sigma)=\prod_{i=1}^{m}[w(l_i)+w(\neg l_i)]-WMC(\neg l_{11}\wedge\ldots\wedge\neg l_{1n_1})-\ldots-WMC(\neg l_{n1}\wedge\ldots\wedge\neg l_{nn_n}) \quad (3)$$

根据公式(3),可给出基于知识编译求解加权模型计算问题的算法 KCCWER(knowledge compilation for counting weighted models using extension rules),该算法的详细流程见附录 B.

**定理 13**. 给定 EPCCL 理论,我们可以在多项式时间求解加权模型计算问题.

证明:公式(3)显然可以在多项式时间内求解,得证.

例 5:给定 EPCCL 理论$\Sigma=\{p\vee q,p\vee\neg q,\neg p\vee r\}$;$w(p)=w(\neg p)=0.5$;$w(q)=0.3$,$w(\neg q)=0.7$;$w(r)=0.8$,$w(\neg r)=0.2$.计算$WMC(\Sigma)$.

$$WMC(\Sigma_T)=(0.5+0.5)\times(0.3+0.7)\times(0.2+0.8)=1,$$
$$WMC(\Sigma(\neg p\wedge\neg q))=0.5\times0.7\times(0.2+0.8)=0.35,$$
$$WMC(\Sigma(\neg p\wedge q))=0.5\times0.3\times(0.2+0.8)=0.15,$$
$$WMC(\Sigma(p\wedge\neg r))=0.5\times(0.3+0.7)\times0.2=0.1,$$

因此,

$$WMC(\Sigma)=1-0.35-0.15-0.1=0.4.$$

## 4 实验结果

我们使用 Visual C++实现了上述 4 种算法:算法 CER、算法 CWER、算法 KCCER、算法 KCCWER,并将上述 4 种算法嵌入 JLU-WMCER 系统中.所有的实验都在一台 CPU 主频为 2.8G、内存 512M 的 PC 机上测试.实验 1 的目的在于说明基于扩展规则的模型计算方法和基于归结的模型计算方法由于互补因子大小的不同而导致的性能上的差异.我们用 Visual C++实现了文献[17]中介绍的 CDP 算法,由于 CDP 不能处理加权模型计算问题,因此只给出了算法 CER 和算法 CDP 的比较.问题样例由一个随机产生器产生,这个随机产生器有 3 个输入参数:变量的个数 $n$、子句的个数 $m$ 以及每个子句的最大长度 $k$.每个子句都是随机地从 $n$ 个变量中选不大于 $k$ 个得到的,并且每个变量为正或为负的概率都等于 0.5,子句的值为(0,1)之间的随机数.

事实上,目前几乎所有的模型计算方法都是在归结原理的基础上求解的,这里可以比较基于扩展规则的方法和基于归结原理的方法.考虑公式(1)和公式(2),当给定子句集中含有的互补文字较多时,显然,我们的算法这时效率一般会更高,因为这时要计算的项的值为 0;反之,如果互补文字较少,则这时基于归结的方法效率应该会更高,因为这时可选的归结较少.这样,假设所有的模型计算问题都在一个光谱上,光谱的一端(比如左端)是任意两个子句都含有互补对的情况,在光谱的另一端(右端)是任意两个子句都不含互补对的情况.在光谱的左端应该用基于扩展规则的方法效率更高;在光谱的右端用基于归结的方法效率更高.然而,实际的问题往往都在这两种极端情况之间,决定对于一个实际的问题哪种方法效率更高就变得尤其重要.为此引入如下定义:

**定义 5**. 给定一个子句集$\Sigma=\{C_1,C_2,\ldots,C_n\}$,互补因子(complementary factor)是含有互补对的子句对的个数

与所有子句对的个数的比值.即 $S/(n\times(n-1)/2)$,其中,$S$ 表示含有互补对的子句对的个数.

虽然很难用互补因子精确地描述出上述问题的复杂性,但直观上说,互补因子越高,问题离光谱的左端就越近,基于扩展规则的算法的效率就越高.在实验 1 中,为了只考察算法性能和互补因子的关系,我们固定了子句长度为 5,变量个数为 50,子句个数为 200,通过限定部分变量在子句中的正负形式来限定互补因子的大小,实验结果是 50 次实验的平均结果.图 1 显示了算法 CER 和算法 CDP 的性能随着互补因子的改变的变化曲线.从图中可以看出,当互补因子较大时,算法 CER 的性能要优于算法 CDP.反之,当互补因子较小时,算法 CDP 的性能要优于算法 CER.在计算加权模型计算问题时,CWER 的变化曲线与图 1 类似.

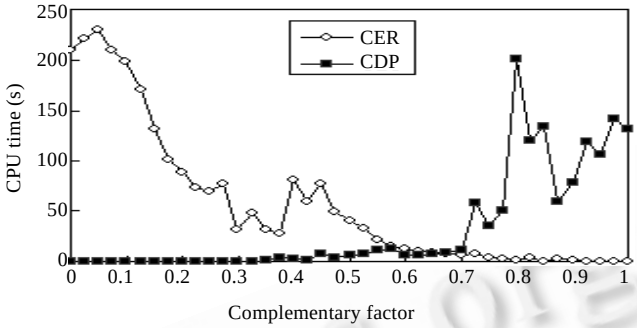


Fig.1 Experimental comparisons of CER and CDP

图 1 算法 CER 和算法 CDP 实验结果比较

实验 2 的目的在于比较基于扩展规则的模型计算方法和基于知识编译的模型计算方法,实验结果见表 1,时间单位均为秒.我们使用了知识编译问题的一些 Benchmark 问题对算法进行测试,并混合使用了一些随机生成的样例,问题样例同样由实验 1 中的随机生成器生成.图中的第 4 列表示的知识编译后生成的新子句集的大小,第 5 列为知识编译的时间,第 6 列和第 7 列为知识编译后计算模型和加权模型的时间,第 8 列和第 9 列给出了利用扩展规则直接进行模型计算和加权模型计算的时间.从表 1 中可以看出,经过知识编译后,无论是模型计算还是加权模型计算的时间都在 0.2s 以内,远小于直接使用扩展规则进行模型计算的时间.但需要注意的是,在知识编译的过程中依然需要花费较长时间,总的来说,编译时间和在线推理时间的总和大约等于直接进行模型计算的时间.不过,由于知识编译实际属于离线推理,用户实际关心的还是在线推理的时间.而且,知识编译所花的时间可以通过对同一个知识库的多次询问得到补偿.

**Table 1** Comparison of offline model counting and online model counting

**表 1** 离线模型计数求解方法和在线模型计数方法比较

| Problem | No. of variables | No. of clauses | No. of results | Compile time | Online time (MC) | Online time (WMC) | Offline time (CER) | Offline time (CWER) |
|---|---|---|---|---|---|---|---|---|
| Adder-3 | 13 | 43 | 169 | 0.13 | <0.01 | <0.01 | 0.12 | 0.12 |
| Adder-4 | 17 | 57 | 681 | 0.13 | <0.01 | <0.01 | 0.14 | 0.14 |
| Adder-5 | 21 | 71 | 2 729 | 0.21 | 0.07 | 0.03 | 0.26 | 0.27 |
| Two-Pipes | 15 | 54 | 208 | 0.13 | <0.01 | <0.01 | 0.31 | 0.29 |
| Three-Pipes | 21 | 82 | 483 | 13.12 | 0.03 | 0.02 | 11.19 | 11.17 |
| Four-Pipes | 27 | 110 | 767 | 101.04 | 0.05 | 0.04 | 79.27 | 82.11 |
| Pigeon-4-5 | 20 | 74 | 4 430 | 63.1 | 0.07 | 0.07 | 67.11 | 65.39 |
| Random problems | 30 | 100 | 8 862 | 31.19 | 0.1 | 0.09 | 33.02 | 32.99 |
| Random problems | 30 | 200 | 14 612 | 27.91 | 0.07 | 0.08 | 23.63 | 23.61 |
| Random problems | 50 | 300 | 127 093 | 130.27 | 0.13 | 0.13 | 112.813 | 114.732 |

虽然基于扩展规则的方法在互补因子较高时,其性能要优于一般的基于归结的方法,但在另外一些 Benchmark 问题中,由于互补因子较低,算法性能要差于基于归结的方法.例如,与 Cachet 系统比较,在 logistic 域问题中,算法 CER 的平均运行时间大约是 Cachet 的 3.7 倍;在 Grid 域算法 CWER 的运行时间大约是 Cachet-WMC 的 2.3 倍.因此在 JLU-ERWMC 系统中,在求解模型计算或加权模型计算时,并不直接调用算法 CER 和算

法 CWER,而是首先计算问题的互补因子.当互补因子超过某个阈值(比如 0.7)时,才调用算法 CER 或算法 CWER,否则调用 Cachet 或 Cachet-WMC 进行求解.表 2 给出嵌入 Cachet 和 Cachet-WMC 算法后 JLU-ERWMC 系统在另一些 Benchmark 问题中的实验结果.我们还在一些随机问题上测试了嵌入了 Cachet 系统后的 JLU-ERWMC.实验结果表明,JLU-ERWMC 的效率要远高于 CDP.例如子句数为 300,变量数为 50,JLU-ERWMC 的速度是 CDP 的 5.1 倍;当子句数为 500,变量数为 100 时,JLU-ERWMC 的速度是 CDP 的 7.3 倍.

**Table 2** Results of CER-Cachet and CWER-Cachet

**表 2** CER-Cachet 和 CWER-Cachet 实验结果

| Problems | CER-Cachet (s) | Problems | CWER-Cachet (s) |
|---|---|---|---|
| log-001 | 0.11 | log-002 | 11 |
| log-002 | 11 | log-003 | 14 |
| log-003 | 12 | log-004 | 57 |
| log-004 | 51 | Grid-002 | 3 |
| log-005 | 401 | Grid-003 | 24 |
| log-012 | 377 | Grid-004 | 57 |

## 5 总 结

本文在扩展规则的基础上设计了求解#SAT 问题的一般方法,提出了求解模型计数问题和加权模型计数问题的新的求解框架.与传统的模型计算方法需要找出哪些赋值是给定命题公式集合的模型不同,我们的方法需要找出哪些赋值不是命题公式的集合,可以看作是目前#SAT 问题求解方法的一种补方法.我们证明了该方法的完备性和有效性,并设计了基于扩展规则的#SAT 求解系统:JLU-ERWMC.实验结果表明,当互补因子较高时,基于扩展规则的方法一般要优于基于归结的方法;反之,当互补因子较低时,基于扩展规则的方法要差于基于归结的方法.

**致谢** 感谢 Henry Kautz 教授提供的 Cachet 和 Cachet-WMC 的源代码,使得我们可以将它们嵌入到系统中,使得系统的效率得到了很大的提升.感谢 Martin Davis 教授对本文中工作的讨论并提出了一些具体的建议.同时,也把这份感谢送给 Sharad Malik,Alvaro del Val,Hantao Zhang 以及 Joe Hurd 等人.感谢 Marquis 和 Mazure 等人提供的一些知识编译问题的 Benchmarks.

**References**:

[1] Zhang LT, Madign CF, Moskewicz MH, Malik S. Efficient conflict driven learning in a Boolean satisfiability solver. In: Pileggi LT, Kuehlmann A, eds. Proc. of the ACM/IEEE ICCAD 2001. New York: ACM Press, 2001. 279−285.

[2] Zecchina R, Mezard M, Parisi G. Analytic and algorithmic solution of random satisfiability problems. Science, 2002,297:812−815.

[3] Zecchina R, Monasson R, Kirkpatrick S, Selman B, Troyansky L. Determining computational complexity from characteristic ‘phase transitions’. Nature, 1999,400:133−137.

[4] Kautz H, Selman B. Unifying SAT-based and graph-based planning. In: Gottlob G, Walsh T, eds. Proc. of the Int’l Joint Conf. on Artificial Intelligence (IJCAI). Seattle: Morgan Kaufmann Publishers, 1999. 318−325.

[5] Kautz HA. Deconstructing planning as satisfiability. In: Zilberstein S, Koehler J, Koenig S, eds. Proc. of the 21st National Conf. on Artificial Intelligence (AAAI 2006). Menlo Park: AAAI Press, 2006. 178−183.

[6] Majercik SM, Littman ML. Contingent planning under uncertainty via stochastic satisfiability. Artificial Intelligence, 2003, 147(1-2):119−162.

[7] Palacios H, Bonet B, Darwiche A, Geffner H. Pruning conformant plans by counting models on compiled *d*-DNNF representations. In: Biundo S, Myers KL, Rajan K, eds. Proc. of the ICAPS 2005. Menlo Park: AAAI Press, 2005. 141−150.

[8] Bacchus F, Dalmao S, Pitassi T. Algorithms and complexity results for #SAT and Bayesian inference. In: Blum A, Randall D, Upfal E, eds. Proc. of the FOCS 2003. Washington: IEEE Computer Society, 2003. 340−351.

[9] Birnbaum E, Lozinskii E. The good old Davis-Putnam procedure helps counting models. Journal of Artificial Intelligence Research, 1999,10(1):457−477.

[10] Sang T, Bacchus F, Beame P, Kautz H, Pitassi T. Combining component caching and clause learning for effective model counting. In: Hans KB, Zhao XS, eds. Proc. of the SAT 2004. Berlin, Heidelberg: Springer-Verlag, 2004. 20−28.

[11] Sang T, Beame P, Kautz H. Heuristic for fast exact model counting. In: Hans KB, Zhao XS, eds. Proc. of the SAT 2005. Berlin, Heidelberg: Springer-Verlag, 2005. 226−240.

[12] Sang T, Beame P, Kautz H. Solving Bayesian networks by weighted model counting. In: Zilberstein S, Koehler J, Koenig S, eds. Proc. of the 20th National Conf. on Artificial Intelligence (AAAI 2005). Menlo Park: AAAI Press, 2005. 475−482.

[13] Wei W, Selman B. A new approach to model counting. In: Hans KB, Zhao XS, eds. Proc. of the SAT 2005. Berlin, Heidelberg: Springer-Verlag, 2005. 324−339.

[14] Chavira M, Darwiche A. Compiling Bayesian networks with local structure. In: Gottlob G, Walsh T, eds. Proc. of the Int'l Joint Conf. on Artificial Intelligence (IJCAI). Seattle: Morgan Kaufmann Publishers, 2005. 211−219.

[15] Chavira M, Darwiche A. On probabilistic inference by weighted model counting. Artificial Intelligence, 2008,172(6-7):772−799.

[16] Darwiche A. On the tractability of counting theory models and its application to belief revision and truth maintenance. Journal of Non-Classical Logics, 2001,11(1-2):11−34.

[17] Lin H, Sun JG, Zhang YM. Theorem proving based on extension rule. Journal of Automated Reasoning, 2003,31(1):11−21.

[18] Lin H, Sun JG. Knowledge compilation using extension rule. Journal of Automated Reasoning, 2004,32(2):93−102.

[19] Wu X, Sun JG, Lin H, Feng SS. Modal extension rule. Progress in Natural Science, 2005,15(6):550−558.

[20] Sun JG, Li Y, Zhu XJ, Lu S. A novel theorem proving algorithm based on extension rule. Journal of Computer Research and Development, 2009,46(1):9−14 (in Chinese with English abstract).

**附中文参考文献:**

[20] 孙吉贵,李莹,朱兴军,吕帅.一种新的基于扩展规则的定理证明算法.计算机研究与发展,2009,46(1):9−14.

## 附录 A. 相关定理证明

定理 3 的证明:当 $S=2^m$ 时,定理 3 可以被描述为:子句集$\Sigma$不可满足当且仅当$\Sigma$中含有 $2^m$ 个互不相同的子句.我们在文献[9]中的定理 2.1 中已经证明了这个结论.下面我们只需要讨论 $S<2^m$ 的情况.

充分性:在$\Sigma$中最多只含有 $2^m$ 个子句,从中抽取出任何一个子句得到一个新的子句集合$\Sigma_1$,根据文献[9]中的定理 2.1,$\Sigma_1$ 是可满足的.事实上,缺少的那个子句的"非"就是该子句集的一个模型,这是因为对于子句集中子句都是极大项的情况,一个子句只有和它的否定放在一起才是不可满足的.若子句集$\Sigma$中缺少子句 $C_i$,$\neg C_i$ 只有和 $C_i$ 放在一起才不可满足,而$\Sigma$当中恰好没有 $C_i$,因此$\Sigma$可满足,$\neg C_i$ 是$\Sigma$的一个模型.由于每个极大项的"非"都是子句集合的一个模型,同时$\Sigma$中最多有 $2^m$ 个模型,因此,若$\Sigma$中含有 $S$ 个子句,则$\Sigma$含有 $2^m-S$ 个模型.

必要性:显然,由 $m$ 个原子组成的子句共有 $2^m$ 个不同的赋值.对于子句集中的每个赋值,它的"非"是一个极大项.而对于$\Sigma$中的每个赋值,它只能使其"非"形式的子句为假.因此,如果一个赋值不是$\Sigma$的模型,则它的"非"形式的子句一定出现在$\Sigma$中.如果$\Sigma$中含有 $2^m-S$ 个模型,则剩下的 $S$ 个不可满足赋值的"非"形式子句一定都在$\Sigma$中,因此$\Sigma$中含有 $S$ 个子句.

定理 4 的证明:

正确性:如果一个子句集经过扩展后被判定为含有 $2^m-S$ 个模型,那么它扩展后得到的极大项的集合一定含有 $S$ 个互不相同的全子句.由定理 3,该极大项的集合一定含有 $2^m-S$ 个模型,再由扩展规则的等价性可知,原子句集一定含有 $2^m-S$ 个模型.

完备性:如果一个子句集含有 $2^m-S$ 个模型,那么它扩展后得到的极大项的集合也一定含有 $2^m-S$ 个模型.再由定理 3 可知,其中一定含有 $S$ 个互不相同的全子句.

定理 8 的证明:给定一个 EPCCL 理论,利用公式(1),我们可以在线性时间内计算出它的极大项个数,因为这时只需要计算公式(1)的前 $n$ 项.因为根据定理 5,从第 $n+1$ 项开始所有的项都为 0.因而根据定理 3,我们可以在多项式时间内计算出 EPCCL 理论的模型个数.

定理9的证明:算法KCCER包括两个过程:首先利用算法KCER将给定子句集编译为等价的EPCCL理论,然后利用算法CER来计算子句的模型个数.由于这两种算法都是正确且完备的,因此算法KCCER是正确和完备的.

命题10的证明:不失一般性,设$C_i$和$C_j$中为含有互补文字的两个子句,其中,$C_i$中包含正文字$l$,$C_j$中包含负文字$\neg l$.要使$C_i$为假,需要赋给$\neg l$真值;要使$C_j$为假,需要赋给$l$真值.因此,对于任意赋值都不会使$C_i$和$C_j$同时为假.

命题11的证明:根据定义4,由于一个子句集的权为它所有模型的权之和,因此当$\Sigma_T|=T$时,它的所有可能赋值都是$\Sigma_T$的模型,因而$\Sigma_T$必然是$\Pi$中权最大的子句集合.根据定义4,我们很容易计算出$WMC(\Sigma_T)$.需要注意的是,如果对于每个文字$l$,$w(l)+w(\neg l)=1$成立,则$WMC(\Sigma_T)=1$.

## 附录B. 算法流程

**算法CER.**

令子句集$\Sigma=\{C_1,C_2,\ldots,C_n\}$,$M$为其原子的集合,且$|M|=m$.

1. $i:=1$, $sum:=0$
2. **while** $i<=n$

   **Loop**

   对于所有的包含$i$个子句的集合$L$

   **Loop**

   $Union:=L$中所有子句文字的集合

   **If** $Union$中不含互补对 **then** $number:=2^{m-|Union|}$

   **Else** $Number:=0$

   **If** $i$ mod 2==1 **then** $sum:=sum+number$

   **Else** $Sum:=sum-number$

   **Endloop**

   $i:=i+1$

   **Endloop**
3. $num\_model:=2^m-sum$
4. **Return** $num\_model$

**算法KCCER.**

设给定子句集$\Sigma=\{C_1,C_2,\ldots,C_n\}$,$M$为其原子的集合,且$|M|=m$.

1. 调用算法KCER编译子句集$\Sigma$,将其转换为等价的EPCCL理论$\Sigma'=\{C_1',C_2',\ldots,C_n'\}$
2. $sum:=0$, $i:=1$;
3. **while** $i<=n$

   **Loop**

   $Union:=C_i$中所有子句文字的集合

   $Number:=2^{m-|Union|}$

   $sum:=sum+number$

   **Endloop**

   $i:=i+1$

4. *num_model*:=$2^m$−*sum*

5. **Return** *num_model*

**算法 KCER**.

1. 输入:令子句集$\Sigma_1=\{C_1,C_2,\ldots,C_n\}$,$\Sigma_2=\Sigma_3=\varnothing$

2. **While** $\Sigma_1\neq\varnothing$

**Loop**

从$\Sigma_1$中选出一个子句,比如$C_1$,把$C_1$加入到$\Sigma_2$中

**While** $i<\Sigma_1$中子句的数目

**Loop**

**While** $j<\Sigma_2$中子句的数目

**Loop**

**If** $C_i$ and $C_j$含互补对 **Then** skip

**Else if** $C_i$蕴含$C_j$ **Then** 从$\Sigma_2$中删除$C_j$

**Else if** $C_j$蕴含$C_i$ **Then** 从$\Sigma_1$中删除$C_i$

**Else** 用扩展规则扩展$C_j$

$j$:=$j$+1

**Endloop**

$i$:=$i$+1

**Endloop**

$\Sigma_3$:=$\Sigma_3\cap\Sigma_2$, $\Sigma_2$:=$\varnothing$

**Endloop**

3. 输出$\Sigma_3$是知识编译过程的结果

**算法 CWER**.

令子句集$\Sigma=\{C_1,C_2,\ldots,C_n\}$,$M$为其原子的集合,且$|M|=m$,设$W$为文字的权值集合.

1. $i$:=1, *sum*:=0

2. **While** $i<=n$

**Loop**

对于所有的包含$i$个子句的集合$L$

**Loop**

*Union*:=$L$中所有子句文字的集合

**If** *Union*中不含互补文字

**Then** 设 *Union*:=$\{l_1,l_2,\ldots,l_k\}$

*Weight*:=*WMC*($\Sigma(\neg l_1\wedge\neg l_2\wedge\ldots\wedge\neg l_k)$)

**Else** *weight*:=0

**If** $i$ mod 2=1 then *sum*:=*sum*+*weight*

**Else** *sum*:=*sum*−*weight*

**Endloop**

$i$:=$i$+1

Endloop

3. *total_weight*:=*WMC*($\Sigma_T$)−*sum*

4. Return *total_weight*

**算法 KCCWER**.

设给定子句集$\Sigma$={$C_1$,$C_2$,…,$C_n$},*M* 为其原子的集合,且|*M*|=*m*.

1. 编译子句集$\Sigma$,将其转换为等价的 EPCCL 理论$\Sigma'$={$C_1'$, $C_2'$ ,…, $C_n'$}

2. *sum*:=0, *i*:=1;

3. **while** *i*<=*n*

**Loop**

*Union*:=$C_i$ 中所有子句文字的集合

令 *Union*:={$l_1$,$l_2$,…,$l_k$}

*weight*:=*WMC*($\Sigma$($\neg l_1\wedge\neg l_2\wedge...\wedge\neg l_k$))

*sum*:=*sum*+*weight*

**Endloop**

*i*:=*i*+1

4. *total_weight*:=$2^m$−*sum*

5. Return *total_weight*

**殷明浩**(1979－),男,安徽安庆人,博士生,主要研究领域为智能规划,自动推理.

**孙吉贵**(1962－2008),男,博士,教授,博士生导师,CCF 高级会员,主要研究领域为智能规划,自动推理.

**林海**(1981－),男,博士生,主要研究领域为自动推理.