# 基于配对方法的自动定理证明*

陈玉泉 陆汝占 余 皓

(上海交通大学计算机科学与工程系 上海 200030)

**摘要** Peter B. Andrews 提出了自动定理证明的配对方法的理论和算法. 本文针对该算法的缺点,给出了一个无需回溯的实现算法,并得到一个高阶逻辑的自动定理证明系统.

**关键词** 自动定理证明,配对,归结,关联,高阶逻辑.

**中图分类号** TP18

从1956年Newell,Shaw和Simon发表他们的著名论文《The Logic Theory Machine》算起,自动定理证明这个研究领域已经有40多年的历史了. 随着Herbrand定理的提出,证明谓词演算中某一公式的恒假性,就可以转化为证明一个命题公式的恒假性.

1964年J. A. Robinson推出了简洁而漂亮的归结原理,直到70年代,归结方法在自动推理领域都占有统治地位,但后来这种统治地位受到了强有力的挑战. 1977年,W. W. Bledsoe发表的重要论文《非归结定理证明》,标志着自动推理进入了一个新的时代. 尽管在归结方法的策略研究中,已经有许多杰出的成果,诸如单元归结、锁归结、广义归结、有序归结、着色归结等等,但是由于所需证明的定理的多样性,使得归结方法的弊病在许多情况下显露出来,那就是使用分配律$[P \vee (Q \wedge R)] \leftrightarrow [(P \vee Q) \wedge (P \vee R)]$来得到合取范式,导致了证明的冗余和不清晰. 尤其在某些高阶逻辑的定理证明过程中表现很突出,公式$F(R \cup S) \leftrightarrow F(R) \cup F(S)$就是一个典型的例子,它化为合析范式后有多达24个子句.

配对方法是一种非归结的自动定理证明方法,这种方法不需象归结方法那样把待证公式的否定化成合取范式,而将待证公式的否定化成只含析取和合取的公式,然后表示成矩阵的形式. 矩阵每一行的各元素之间是析取的关系,而行和行之间是合取的关系,由于矩阵的元素可以还是矩阵,所以,不论多么复杂的公式,都能用这种方法直观地表示出来. 这种直观不仅有利于人的观察,对计算机的处理也是有好处的. 这种递归定义的数据表示,有利于用递归程序进行处理.

## 1 配对方法的基本理论

首先,介绍本文中用到的逻辑符号:~(非),∧(合取),∨(析取),→(蕴涵),↔(等价),

---

* 本文研究得到国家自然科学基金资助. 作者陈玉泉,1968年生,讲师,主要研究领域为推理技术与定理机械证明,汉语语义计算理论. 陈汝占,1940年生,教授,博士导师,主要研究领域为程序设计语言,推理技术与定理证明,自然语言语义理论与理解. 余皓,1973年生,硕士生,主要研究领域为推理技术与定理机械证明.

本文通讯联系人:陈玉泉,上海200030,上海交通大学计算机科学与工程系

本文1996-04-05收到修改稿

∀(全称量词),∃(存在量词).此外,用$\theta A$表示对表达式$A$作替换$\theta$后的结果.

其次,介绍配对方法的基本概念和基本定理,本节概念和定理属于Peter B. Andrews.[1~3]

**定义** 1.1. 一个合式公式(wff)为否定范式(negation normal form)当且仅当每个~均出现在原子上.如果该公式不含自由变元,则称为否定范句(negation normal sentence).

一个合式公式经下述等价变换即可化为等价的否定范式:

$\sim\sim M\leftrightarrow M$ $\sim[M\wedge N]\leftrightarrow[\sim M\vee\sim N]$

$\sim[M\vee N]\leftrightarrow[\sim M\wedge\sim N]$ $\sim\forall xM\leftrightarrow\exists x\sim M$

$\sim\exists xM\leftrightarrow\forall x\sim M$

**定义** 1.2. 一个合式公式称为全称的(universal)当且仅当公式中所有全称量词前的否定出现偶数次,所有存在量词前的否定出现奇数次.

给定一个待证公式$B$,令$C$为$B$的全称封闭式的否定的否定范式,将$C$斯科伦化后得全称否定范句$D$.

**定理** 1.3. 全称否定范句$D$无模型当且仅当$B$为永真公式.

**定义** 1.4. 一个合式公式$F$称为标准的(normal)当且仅当$F$中的自由变元与约束变元均不同名,所有约束变元也不同名.

我们可以通过对约束变元改名将公式$E$标准化(normalized)成$F$,显然$E\leftrightarrow F$.

**定义** 1.5. 设$D$为全称否定范句,定义$D$的扩充(amplification)集合如下:

称一个合式公式$R'$由$R$经量词复制(quantifier duplication)而得当且仅当$R'$为把$R$中型为$\forall xM$的子公式替换为$\forall xM\wedge\forall xM$后的结果.

如果有一个合式公式的序列$D_1,\ldots,D_n$,其中$D_{i+1}$为$D_i$经量词复制而得$(0<i<n)$,则称$D_n$由$D_1$经多次量词复制而得.

设$E$由$D$经多次量词复制而得,$F$为$E$的标准化公式,$G$为消除$F$的所有量词的结果,则称$G$为$D$的一个扩充.

下面举例说明公式$B$到公式$G$的演变:

定理:

$(B)\ \exists x\forall y(Px\leftrightarrow Py)\rightarrow(\exists xPx\leftrightarrow\forall yPy)$

否定$B$,消去↔和→:

$(C)\ \exists x\forall y((\sim Px\vee Py)\wedge(\sim Py\vee Px))\wedge((\exists xPx\wedge\exists y\sim Py)\vee(\forall yPy\wedge\forall x\sim Px))$

斯科伦化:

$(D)\ \forall y((\sim Pc\vee Py)\wedge(\sim Py\vee Pc))\wedge((Pd\wedge\sim Pe)\vee(\forall zPz\wedge\forall x\sim Px))$

量词复制:

$(E)\ \forall y((\sim Pc\vee Py)\wedge(\sim Py\vee Pc))\wedge\forall y((\sim Pc\vee Py)\wedge(\sim Py\vee Pc))\wedge((Pd\wedge\sim Pe)\vee(\forall zPz\wedge\forall x\sim Px))$

标准化:

$(F)\ \forall y((\sim Pc\vee Py)\wedge(\sim Py\vee Pc))\wedge\forall w((\sim Pc\vee Pw)\wedge(\sim Pw\vee Pc))\wedge((Pd\wedge\sim Pe)\vee(\forall zPz\wedge\forall x\sim Px))$

消去量词:

$(G)\ ((\sim Pc \vee Py) \wedge (\sim Py \vee Pc)) \wedge ((\sim Pc \vee Pw) \wedge (\sim Pw \vee Pc)) \wedge ((Pd \wedge \sim Pe) \vee (Pz \wedge \sim Px))$

化成矩阵形式:

$$\begin{bmatrix} \sim Pc & Py \\ \sim Py & Pc \\ \sim Pc & Pw \\ \sim Pw & Pc \\ \begin{bmatrix} Pd \\ \sim Pe \end{bmatrix} & \begin{bmatrix} Pz \\ \sim Px \end{bmatrix} \end{bmatrix}$$

**定义** 1.6. 一个替换是形如$\{t1/v1,\ldots,tn/vn\}$的一个有限集合,其中$vi$是变量名,$ti$是不同于$vi$的项,并且$vi$各不相同.

**定义** 1.7. 设$G$为不含量词的合式公式,$L(G)$为$G$中所有原子出现的集合,$L(G)$上的二元关系$\mu$定义如下:

$L\mu K$当且仅当存在替换$\theta$使得$\theta K = \sim\theta L$,称$\mu$为$G$的一个配对集(mating),称$K$和$L$为配对的原子出现.

**定义** 1.8. 递归定义通过$G$的垂直路径如下:

(1)$G$为原子$L$,则通过$G$的垂直路径为$\langle L\rangle$.

(2)$G=G_1 \vee G_2$,则每条通过$G_1$的垂直路径和通过$G_2$的垂直路径均为通过$G$的垂直路径.

(3)$G=G_1 \wedge G_2$,设$P_i$为某一条通过$G_i$的垂直路径,$i=1,2$,则序列$P_1$和$P_2$的连接为通过$G$的垂直路径.

**定义** 1.9. 设$\mu$为不含量词的否定范式$G$的一个配对集,称$\mu$为可接受路径的配对集(path-acceptable)当且仅当每条通过$G$的垂直路径都包含一对配对的原子出现.

**定义** 1.10. 称不出现在其它量词作用域内的量词为最外层量词(outermost quantifier).

**定理** 1.11. 设$D$是一阶逻辑的全称否定范句,$D$无模型当且仅当$D$的某个扩充$G$有可接受路径的配对集,其中$G$只复制$D$的最外层量词.

## 2 实现算法

**定义** 2.1. 称$(K,L)$为配对当且仅当存在替换$\theta$使得$\theta K = \sim\theta L$.

**定义** 2.2. 如果一条通过$G$的垂直路径$P$上含有配对$(K,L)$的元素$K$和$L$,则称配对$(K,L)$覆盖路径$P$.

**定义** 2.3. 一个配对能覆盖的路径的集合称为该配对的路径集. 一个配对集中所含配对的路径集的并集称为该配对集的路径集.

**实现算法**:

(1)输入待证公式$B$,否定$B$,消去↔和→,斯科伦化后得公式$D$,$F$的初值为$D$.

(2)消除$F$的量词得$G$,令集合$\mu$为空集.

(3)是否还有未检查过的通过 $G$ 的垂直路径? 若有,则执行第(4)步,否则执行第(5)步.

(4)将该路径上 $\mu$ 中没有的配对加入 $\mu$,记录其替换和路径集,而对 $\mu$ 中已有配对,将该路径加入其路径集,然后回到第(3)步.

(5)将替换为空集的配对从 $\mu$ 中删除,把这种配对覆盖的路径从全部路径中删除,得到尚需覆盖的路径集 $Q$.

(6)$Q$ 为空集吗? 如果是空集,则停机,此时所有通过 $G$ 的垂直路径均被覆盖,公式 $B$ 被证明了. 否则,对所有 $\mu$ 中元素 $M_i$,设 $M_i$ 的路径集为 $P_i$,置 $P_i$ 为 $P_i \cap Q$,如果 $P_i$ 为空集,则置 $\mu$ 为 $\mu-\{M_i\}$.

(7)检查 $\mu$ 中是否存在一个子集为 $G$ 的配对集,且其路径集为 $Q$,如果有,则停机,公式 $B$ 被证明了,否则复制 $F$ 的最外层量词并标准化,更新 $F$,转回执行第(2)步.

## 3 证明系统的说明

证明系统按功能可以分为 6 个模块:

(1)输入输出模块

本模块的功能包括 2 方面:①输入要证明的以字符串表示的公式;②输出证明的过程,中间结果(如矩阵、语法树、配对、替换等)和最后的执行结果.

(2)公式处理模块

本模块的功能是否定输入的公式、消去等价和蕴涵,把否定深入到原子,用斯科伦函数消去存在量词,缩小全称量词的作用域,将同名的约束变元改名. 将此时的公式分析成语法树,备份在盘交换区(由内存管理模块控制),这样既可节约内存,又为后面可能要进行的量词复制提供未消去量词的语法树.

(3)矩阵生成模块

本模块将语法树上的全称量词消去,然后转化成配对方法所要求的矩阵形式. 并对产生的矩阵进行优化,减少矩阵的垂直路径数,甚至在导出空行时,直接证明公式.

这个模块主要用了以下结果:

$$\begin{bmatrix} \begin{bmatrix} L1 \\ P1 \end{bmatrix} \vee \ldots \vee \begin{bmatrix} Ln \\ Pn \end{bmatrix} \\ \begin{bmatrix} L1' \\ \cdot \\ \cdot \\ Ln' \\ Q \end{bmatrix} \vee R \end{bmatrix} \leftrightarrow \begin{bmatrix} \begin{bmatrix} L1 \\ P1 \end{bmatrix} \vee \ldots \vee \begin{bmatrix} Ln \\ Pn \end{bmatrix} \\ R \end{bmatrix}$$

如果公式$([L1 \wedge P1] \vee \ldots \vee [Ln \wedge Pn]) \wedge ([L1' \wedge \ldots \wedge Ln' \wedge Q] \vee R)$中有 $Li' = \sim Li (i = 1, 2, \ldots, n)$,则等价于公式$([L1 \wedge P1] \vee \ldots \vee [Ln \wedge Pn]) \wedge R$.

(4)量词复制模块

根据配对方法的原理,只需复制最外层的量词. 本模块从盘交换区通过内存管理模块取出备份的未消去量词的语法树,用宽度优先遍历的方法找到最外层的全称量词,把以此量词

为根的子树 $T$ 变为以 2 个相同子树 $T$ 合取而成的子树 $T \wedge T$.

(5)内存管理模块

由于程序要在微机上运行,受到 DOS 操作系统和微机内存的限制,不可能把所有的数据都同时驻留在内存中,所以要用本模块在内存和作为盘交换区的文件之间进行数据的调入和调出. 对暂时不用的数据就调出,等要用时调入. 如前面的语法树,以及一些链表、堆栈等. 本模块节省的内存空间,使得很多较复杂的公式也能证明.

(6)配对证明模块

本模块的功能是通过对矩阵中所有垂直路径的一次遍历,把所有的配对及该配对的路径集、所需进行的替换等信息一起收集起来. 然后对此进行优化,优化的内容包括:将替换集为空集的配对从 $\mu$ 中删除,把这种配对覆盖的路径从全部路径中删除,得到尚需覆盖的路径集 $Q$,把不能覆盖 $Q$ 的任意一条路径的配对从 $\mu$ 中删除,将不属于 $Q$ 的路径从 $\mu$ 中配对的路径集中删除. 最后检查 $\mu$ 中是否存在一个子集为配对集,且其路径集为 $Q$,如果有,则停机,否则就起动量词复制模块.

## 4　与 Peter B. Andrews 的算法的比较

Peter B. Andrews 在文献[2]中提出自动定理证明的配对的思想,然后在文献[1]中提出配对方法. 几乎与 Peter B. Andrews 同时,Bibel W. 在文献[4～6]中提出了自动定理证明的关联(connections)方法.

关联方法将待证公式直接转化为矩阵形式,而不是配对方法中的将待证公式的否定转化为矩阵形式,定义了通过矩阵的水平路径、关联集和互补(complementary),分别类似于配对方法中的通过矩阵的垂直路径、配对集和可接受路径的配对集,最后得到定理:一个公式是永真的当且仅当它是互补的.

由此可见,关联方法与配对方法是基础相同的类似方法,实现的效率也在同一个数量级上.

在文献[1]中,Peter B. Andrews 提出如下算法:

(1)输入待证公式 $B$,否定 $B$,消去↔和→,斯科伦化后得公式 $D$.

(2)创建与空配对集相关的 $D$ 中潜在配对的集合 $\Phi(D)$.

(3)$F$ 的初值为 $D$.

(4)令 $\mu$ 为空集,创建与 $\mu$ 相关的 $F$ 中潜在配对的集合 $\Phi_\mu(F)$.

(5)$\mu$ 是否为可接受路径的配对集? 如果是,则公式 $B$ 被证明了,否则转第(6)步.

(6)选择一条没有配对的通过 $F$ 的垂直路径 $P$.

(7)$P$ 中是否有潜在的配对? 如果有,则转第(8)步,否则转第(9)步.

(8)将此配对加入 $\mu$,修改 $\Phi_\mu(F)$,转第(5)步.

(9)$F$ 中所有配对是否都检索过? 如果是,则转第(10)步,否则回溯前面的路径,寻找另外的配对.

(10)复制 $F$ 的最外层量词并标准化,更新 $F$,转回第(4)步.

在 Peter B. Andrews 的算法中,配对集 $\mu$ 不是对路径一次扫描产生的,而是在回溯的过程中逐步修改原来的配对集得到的. 通过对该算法的实现,我们发现该算法的回溯部分是

影响整个算法的效率的主要原因.因为回溯要求记录整个路径搜索的历史和每次配对所进行的变元替换的历史,在回溯时还要把进行过替换的变元恢复过来,由于该算法对矩阵进行重复搜索和记录大量的变元替换的历史,在实现中要消耗大量的时间和空间资源,使得程序的效率较低,经常因为内存不够而导致死机.

鉴于该算法存在的不足,针对回溯和反复替换与恢复的缺点,我们的算法有如下改进:

(1)仅对矩阵的全部路径进行一次搜索,不回溯.

(2)对路径上的所有配对,仅记录对变元的替换,不进行实际的替换,当然也就不必进行替换的恢复.

下面是一组例子:

№1 $[[M\vee N]\wedge R]\leftrightarrow[[M\wedge R]\vee[N\wedge R]]$

№2 $\forall x[Px\leftrightarrow\forall yPy]\leftrightarrow[\exists xPx\leftrightarrow\forall yPy]$

№3 $\exists x\forall y[Px\leftrightarrow Py]\leftrightarrow[\exists xPx\leftrightarrow\forall yPy]$

№4 $[R\leftrightarrow S]\rightarrow[R\rightarrow S]$

№5 $\sim\exists G\forall F\exists J[G(J)=F]$(集合论中 Cantor 定理)

№6 $\exists x\forall y[Px\leftrightarrow Py]\rightarrow[\exists xPx\leftrightarrow\forall yPy]$

№7 $\forall x[Px\leftrightarrow\exists yPy]\leftrightarrow[\forall xPx\leftrightarrow\exists yPy]$

№8 $\exists S\forall x[[Sx\vee Px]\wedge[\sim Sx\vee Qx]]\leftrightarrow\forall y[Py\vee Qy]$

各公式的实际运行情况如表1所示.

表1

| 公式序号 | 改进算法所需内存(字节) | 改进算法运行时间(1/100s) | Andrews 算法所需内存(字节) | Andrews 算法运行时间(1/100s) |
|---|---|---|---|---|
| № 1 | 2 681 | 6 | 6 311 | 6 |
| № 2 | 10 068 | 21 | 23 891 | 22 |
| № 3 | 9 815 | 22 | 20 651 | 22 |
| № 4 | 1 190 | 5 | 1 224 | 5 |
| № 5 | 1 433 | 6 | —— | 死机 |
| № 6 | 3 217 | 6 | 3 613 | 6 |
| № 7 | 10 068 | 22 | 23 648 | 22 |
| № 8 | 7 655 | 22 | 225 700 | 247 |

注:表中所需内存是指动态申请的内存空间的总和,是在80386DX33型微机上运行的结果.

表1的8个例子中,既有命题演算公式,也有一阶逻辑公式,还有高阶逻辑公式,特别对高阶逻辑公式(№ 5,№ 8)的证明,我们的算法更具优越性.

## 参考文献


1 Andrews Peter B. Theorem proving via general mating. J. ACM, April 1981,**28**:193~214.

2 Andrews Peter B. Refutation by matings. IEEE Trans. Comput., 1976,**C-25**:801~807.

3 Andrews Peter B, Bishop Matthew, Issar Sunil *et al.* TPS: a theorem proving system for classical type theory. Tech. Rept. 94-166, Department of Mathematics Carnegie Mellon University, 1994.

4 Bibel W. Matrices with connections. J. ACM, 1981,**28**:633~645.

5 Bibel W. Automated theorem proving. Vieweg Verlag, Braunschweig, 1982.

6 Bibel W. Matings in matrices. German Workshop on Artificial Intelligence 81, Informatik-Fachberichte 47,

Springer-Verlag, Berlin, 1981.

# AUTOMATIC THEOREM PROVING BASED ON MATINGS

CHEN Yuquan　LU Ruzhan　YU Hao

(*Department of Computer Science and Engineering　Shanghai Jiaotong University　Shanghai*　200030)

**Abstract**　This paper first introduces the theory and algorithm of mating method in automatic theorem proving advanced by Peter B. Andrews, and then gives an implementation algorithm without backtracking, finally obtains a theorem proving system for higher-order logic.

**Key words**　Automatic theorem proving, mating, resolution, connection, higher-order logic.

**Class number**　TP18