第7卷第6期 软件学报 Vol. 7, No. 6
1996年6月 JOURNAL OF SOFTWARE Jun. 1996

机器定理证明的反向归约方法*

李爱中 黄厚宽
(北方交通大学计算机系 北京 100044)

乔佩利
(哈尔滨理工大学计算机系 哈尔滨 150040)

摘要 基于代数和递归函数理论,本文定义了代数递归谓词.代数递归谓词是一类广泛的谓词.基于数学归纳法,作者给出了证明代数递归谓词永真性的反向归约方法及相应的算法Reduction.由于采用反向归约方式来完成定理证明,从根本上消除了正向组合式定理证明所产生的组合爆炸,因而极大地提高了定理证明的效率.

关键词 自动定理证明,代数,递归,反向归约,数学归纳法.

推理规则是自动定理证明ATP(automated theorem proving)的关键.归结原理(Resolution Principle) $\sim P \vee C_1, P \vee C_2 \models C_1 \vee C_2$;[1]蕴涵规则 $A, A \rightarrow B \models B$;数学归纳法 $P(0), \forall a[P(a) \rightarrow P(a+1)] \models \forall xP(x)$都是重要的推理规则.由于它们从2个前提而推导出1个结论,我们称之为组合式规则.但是,反过来考查上述3个推理规则.由 $C_1 \vee C_2$ 根据归结原理可以找到多对 $\sim P \vee C_1$ 和 $P \vee C_2$,由B根据蕴涵规则也可以找到多对 A 和 $A \rightarrow B$;特别是当 $C_1 \vee C_2$ 为定理时,$\sim P \vee C_1$ 和 $P \vee C_2$ 未必均为定理;当 B 为定理时,A 和 $A \rightarrow B$ 未必均为定理.也就是说,归结原理和蕴涵规则作为推理规则不是可逆的,故此我们称之为纯粹的组合式推理规则.但是数学归纳法则不然,从 $\forall xP(x)$ 能很容易地找到唯一一对 $P(0)$ 和 $\forall a[P(a) \rightarrow P(a+1)]$,并且当 $\forall xP(x)$ 为定理时,$P(0)$ 和 $\forall a[P(a) \rightarrow P(a+1)]$ 必为2个定理.换句话说,数学归纳法把证明 $\forall xP(x)$ 转化成2个子定理 $P(0)$ 和 $\forall a[P(a) \rightarrow P(a+1)]$ 的证明.我们称数学归纳法是归约式规则,因为它能把1个定理归约为2个子定理.

对于纯粹的组合式规则虽能采用正向推理策略,从已知不断组合,直至生成结论为止.这个组合过程难免产生组合爆炸,而组合爆炸正是目前阻碍自动定理证明发展的根本原因.对于归约式规则,可以采用反向推理策略,从结论开始,不断分解或归约,直至所有子目标均为已知为止.这样可以完全避免无用的子目标的产生,因而可以从根本上避免组合爆炸,进而提高定理证明的效率.

如果1个定理有2个或2个以上证明过程,我们称此定理是歧义的(Ambiguous).如果1个逻辑系统中存在歧义定理,我们称此逻辑系统是歧义的.和形式语言的歧义性类似,1个

---

* 本文研究得到国家自然科学基金资助.作者**李爱中**,1963年生,副教授,上要研究领域为机器学习和自动推理.**黄厚宽**,1940年生,教授,主要研究领域为人工智能,专家系统.**乔佩利**,1951年生,副教授,主要研究领域为机器学习和自动推理.

本文通讯联系人:李爱中,北京 100044,北方交通大学计算机系

本文1995－03－30收到修改稿

逻辑系统可能是先天性歧义的,故此根本不可能在该系统内实现反向归约式推理,而只能在该系统内实现正向组合式推理,从而组合爆炸是不可避免的.

使用蕴涵规则或归结原理的一阶逻辑系统是歧义的.1 个一阶谓词逻辑系统是否是(先天)歧义的是不可判定的.故此,若使用蕴涵规则或归结原理,歧义是不可避免的.而蕴涵规则和归结原理带来的歧义性能否被 1 条或几条新的推理规则所消除是难以判定的;若能,则可能在一阶逻辑上实现高效的定理证明;反之,组合爆炸是不可避免的.由于先天歧义性的存在,在一阶逻辑中研究定理证明存在着危险.

因此,为了避免组合爆炸,本文只使用数学归纳法作为唯一的 1 条推理规则.在一阶逻辑中,由于函数和谓词本身并未严格定义,函数和谓词并不一定是完全可计算的或完全可判定的,难以直接使用数学归纳法.为此,作者定义了一类函数和谓词,并在其上使用数学归纳法来完成谓词永真性的证明.

作者在定义代数递归函数和代数递归谓词时,借鉴了吴文俊先生的几何定理证明方法[2]中的代数思想[3],并结合了递归函数理论[4,5]中递归的概念,因而定义了一个广泛的函数类和谓词类,并在此基础上进行了自动定理证明的反向归约式方法的研究.

1　背景知识

为了便于理解,先介绍文中所用的关于代数和递归函数的一些概念和结论.

1.1　原始递归函数和原始递归谓词

原始递归函数(谓词),从本质上几乎包括全部常用的函数(谓词),但并非全部 Turing 可计算函数(谓词).在原始递归谓词之上讨论自动定理证明是有实用意义的,这可以从下面的定义及其解释中得到.[4,5]

原始递归函数集由下列规则定义:

(1)零函数 $O(x)=0$,后继函数 $S(x)=x+1$ 和投影函数 $P_j^n(x_1,\cdots,x_n)=x_j(1\leqslant j\leqslant n)$ 均是原始递归函数.

(2)如果 n 元函数 $g_1(x_1,\cdots,x_n),\cdots,g_m(x_1,\cdots,x_n)$ 和 m 元函数 $h(x_1,\cdots,x_m)$ 均为原始递归函数,那么 n 元函数

$$f(x_1,\cdots,x_n)=h(g_1(x_1,\cdots,x_n),\cdots,g_m(x_1,\cdots,x_n))$$

也是原始递归函数.

(3)如果 $n-1$ 元函数 $g(x_2,\cdots,x_n)$ 和 $n+1$ 元函数 $h(x_1,\cdots,x_{n+1})$ 均为原始递归函数,那么 n 元函数

$$\begin{cases} f(0,x_2,\cdots,x_n)=g(x_2,\cdots,x_n) \\ f(x_1+1,x_2,\cdots,x_n)=h(x_1,\cdots,x_n,f(x_1,\cdots,x_n)) \end{cases}$$

也是原始递归函数.

谓词 $P(x_1,\cdots,x_n)$ 是原始递归谓词,如果其特征函数 $f(x_1,\cdots,x_n)$ 是原始递归函数,谓词 $P(x_1,\cdots,x_n)$ 和其特征函数 $f(x_1,\cdots,x_n)$ 之间有下列关系:

$$P(x_1,\cdots,x_n)\text{为真 iff } f(x_1,\cdots,x_n)=0$$

1.2　代数方程和归结引理

一元多项式 $P(X)=a_0x^n+\cdots+a_n$，其中 n 为正整数，$a_0,\cdots,a_n$ 为整数，$a_0\neq 0$.

n 元多项式 $P(x_1,\cdots,x_n)=P_0(x_1,\cdots,x_{n-1})x_n^m+\cdots+P_m(x_1,\cdots,x_{n-1})$，其中 m 为正整数，$P_0(x_1,\cdots,x_{n-1}),\cdots,P_m(x_1,\cdots,x_{n-1})$ 均为 $n-1$ 元多项式，$P_0(x_1,\cdots,x_{n-1})\neq 0$.

设 ξ 是 $p(x)=a_0x^n+\cdots+a_n=0$ 的任意一个根，那么 $|\xi|\leqslant\max\{1,\frac{1}{|a_0|}\sum_{i=1}^{n}|a_i|\}$. 因此，一元代数方程是否有自然数根是可判定的，若有，则可以求出其最小的自然根.

设 $p_1(x_1,\cdots,x_k,Y)=p_{10}(x_1,\cdots,x_k)Y^m+\cdots+p_{1m}(x_1,\cdots,x_k)$，$p_2(x_1,\cdots,x_k,Y)=p_{20}(x_1,\cdots,x_k)Y^n+\cdots+p_{2n}(x_1,\cdots,x_k)$ 为 2 个多项式，$m>0,n>0$，$p_{10}(x_1,\cdots,x_k)\neq 0$，$p_{20}(x_1,\cdots,x_k)\neq 0$，那么方程 $p_1(x_1,\cdots,x_n,Y)=0$ 和 $p_2(x_1,\cdots,x_n,Y)=0$ 在复数域上对于 Y 有公共根的充分必要条件是结式如下，而 $R(p_1,p_2)$ 是关于 $x_1,\cdots,x_k$ 的多项式.[3]

$$R(p_1,p_2)=\begin{vmatrix} p_{10} & \cdots & p_{1m} & & \\ & p_{10} & \cdots & p_{1m} & \\ & \cdots & \cdots & & \\ & & p_{10} & \cdots & p_{1m} \\ p_{20} & \cdots & p_{2n} & & \\ & p_{20} & \cdots & p_{2n} & \\ & \cdots & \cdots & & \\ & & p_{20} & \cdots & p_{2n} \end{vmatrix}=0$$

利用结式的上述性质，不难证明下列归结引理.

归结引理. 在自然数域内，如果 $\forall x_1\cdots\forall x_k\exists y(p_1(x_1,\cdots,x_k,y)=0\wedge p_2(x_1,\cdots,x_k,y)=0)$，那么存在多项式 $p_3(x_1,\cdots,x_k)=0$.

证明：设 $p_1(x_1,\cdots,x_k,Y)=p_{10}(x_1,\cdots,x_k)Y^m+\cdots+p_{1m}(x_1,\cdots,x_k)$，$p_2(x_1,\cdots,x_k,Y)=p_{20}(x_1,\cdots,x_k)Y^n+\cdots+p_{2n}(x_1,\cdots,x_k)$. 根据 m 和 n 的不同情况直接构造出 $p_3(x_1,\cdots,x_n)$.

1° $m>0,n>0$，规定 $p_3(x_1,\cdots,x_k)=R(p_1,p_2)$；

2° $m=0,n>0$，规定 $p_3(x_1,\cdots,x_k)=R(p_{10}^2+p_2^2,p_2)$；

3° $m>0,n=0$，规定 $p_3(x_1,\cdots,x_k)=R(p_1,p_{20}^2+p_1^2)$；

4° $m=0,n=0$，规定 $p_3(x_1,\cdots,x_k)=p_{10}^2+p_{20}^2$

显然，$p_3(x_1,\cdots,x_k)$ 满足此引理. □

此引理和归结原理很类似，但是此引理只是在代数方程上进行归结. 我们称 $p_3(x_1,\cdots,x_k)$ 为从 $p_1(x_1,\cdots,x_n,Y)=0$ 和 $p_2(x_1,\cdots,x_n,Y)=0$ 中消去 Y 的结果.

2　代数递归函数和代数递归谓词

代数递归函数由下列规则定义：

如果 $g(x_1,\cdots,x_{n-1})$ 是 1 个 $n-1$ 元代数递归函数，$p(x_1,\cdots,x_{n+2})$ 是 1 个 $n+2$ 元多项式，那么 n 元代数递归函数 $f(x_1,\cdots,x_n)$ 由下式定义：

$$\begin{cases} f(0,x_2,\cdots,x_n)=g(x_2,\cdots,x_n) \\ f(x_1+1,x_2,\cdots,x_n)\text{是关于 } t \text{ 的一元代数方程 } p(x_1,\cdots,x_n,f(x_1,\cdots,x_n),t)=0 \\ \text{的最小自然根} \end{cases}$$

当 $n=1$ 时
$$\begin{cases} f(0)\text{为一个已知常数} \\ f(x+1)\text{是关于 } t \text{ 的一元代数方程 } p(x,f(x),t)=0 \text{ 的最小自然根} \end{cases}$$

显然，代数递归函数中含有部分函数和全函数，下文仅涉及到全函数.

关于代数递归函数，我们有下列定理.

定理 1. $O(x),S(x)$ 和 $p_j^n(x_1,\cdots,x_n)$ $(1\leqslant j\leqslant n)$ 均是代数递归函数.

定理 2. 如果 $g_1(x_1,\cdots,x_n),\cdots,g_m(x_1,\cdots,x_n)$和 $h(x_1,\cdots,x_m)$均为代数递归函数且均为全函数,那么 $h(g_1(x_1,\cdots,x_n),\cdots,g_m(x_1,\cdots,x_n))$也是代数递归函数,而且是全函数.

证明:完整的证明可以对 n 和 m 实施数学归纳法而得到. 这里仅证 $n=1,m=1$ 的情形.

由于 h,g 均为代数递归函数,故此存在 2 个多项式 p_1 和 p_2 具有下列性质:

$$p_1(x_1,h(x_1),h(x_1+1))=0 \tag{1}$$

$$p_2(x,g(x),g(x+1))=0 \tag{2}$$

从(1)把 x_1 代入 $g(x)$得　$p_1(g(x),h(g(x)),h(g(x)+1))=0$ (3)

从(3)、(2)消去 $g(x)$得　$p_3(x,g(x+1),h(g(x)),h(g(x)+1))=0$ (4)

从(4)、(2)消去 $g(x+1)$得　$p_4(x,g(x),h(g(x)),h(g(x)+1))=0$ (5)

从(5)、(3)消去 $h(g(x)+1)$得　$p_5(x,g(x),h(g(x)))=0$ (6)

从(6)对 x 进行改名并代入 $x+1$ 得　$p_5(x+1,g(x+1),h(g(x+1)))=0$ (7)

改写(7)为　$p_6(x,g(x+1),h(g(x+1)))=0$ (8)

从(8)、(2)消去 $g(x+1)$得　$p_7(x,g(x),h(g(x+1)))=0$ (9)

由(9)、(6)消去 $g(x)$得　$p_8(x,h(g(x+1)),h(g(x+1)))=0$

因此,显然 $h(g(x+1))$为 $p_8(x,g(x)),t)=0$ 关于 t 的最小自然根.

故此 $h(g(x))$为代数递归函数,显然也是全函数. □

定理 3. 如果 $g(x_2,\cdots,x_n)$和 $h(x_1,\cdots,x_{n+1})$都是代数递归函数,而且是全函数,那么由下式定义的函数

$$\begin{cases} f(0,x_2,\cdots,x_n)=g(x_2,\cdots,x_n) \\ f(x_1+1,x_2,\cdots,x_n)=h(x_1,\cdots,x_n,f(x_1,\cdots,x_n)) \end{cases}$$

也是代数递归函数而且是全函数.

证明:由于 h 是代数递归函数,故此存在多项式 p_1 满足下式:

$$p_1(x_1,\cdots,x_{n+1},h(x_1,\cdots,x_{n+1}),h(x_1+1,x_2,\cdots,x_{n+1}))=0 \tag{1}$$

将(1)中 x_{n+1}代入 $f(x_1,\cdots,x_n)$得:

$$p_1(x_1,\cdots,x_n,f(x_1,\cdots,x_n),h(x_1\cdots,x_n,f(x_1,\cdots,x_n)),h(x_1+1,x_2,\cdots,x_n,f(x_1,\cdots,x_n)))=0$$

即

$$p_1(x_1,\cdots,x_n,f(x_1,\cdots,x_n),f(x_1+1,x_2,\cdots,x_n),h(x_1+1,x_2,\cdots,x_n,f(x_1,\cdots,x_n)))=0 \tag{2}$$

从(1)先对 x_1 改名后代入 x_1-1,然后将 x_{n+1}代入 $f(x_1,\cdots,x_n)$,并改写成为:

$$p_2(x_1,\cdots,x_n,f(x_1,\cdots,x_n),h(x_1-1,x_2,\cdots,x_n,f(x_1,\cdots,x_n)),f(x_1+1,x_2,\cdots,x_n))=0 \tag{3}$$

从(2)、(3)消去 $f(x_1,\cdots,x_n)$得:

$$p_3(x_1,\cdots,x_n,f(x_1+1,x_2\cdots,x_n),h(x_1+1,x_2,\cdots,x_n),f(x_1,\cdots,x_n)),h(x_1-1,x_2,\cdots,x_n,f(x_1,\cdots,x_n)))=0 \tag{4}$$

从(2)、(3)消去 $f(x_1+1,x_2,\cdots,x_n)$得:

$$p_4(x_1,\cdots,x_n,f(x_1,\cdots,x_n),h(x_1+1,x_2,\cdots,x_n,f(x_1,\cdots,x_n)),h(x_1-1,x_2,\cdots,x_n,f(x_1,\cdots,x_n)))=0 \tag{5}$$

从(4)、(5)消去 $h(x_1-1,x_2,\cdots,x_n,f(x_1,\cdots,x_n))$得:

$$p_5(x_1,\cdots,x_n,f(x_1,\cdots,x_n),f(x_1+1,x_2,\cdots,x_n),h(x_1+1,x_2,\cdots,x_n,f(x_1,\cdots,x_n)))=0 \tag{6}$$

从(6)、(2)消去 $h(x_1+1,x_2,\cdots,x_n,f(x_1,\cdots,x_n))$得:

$$p_6(x_1,\cdots,x_n,f(x_1,\cdots,x_n),f(x_1+1,x_2,\cdots,x_n))=0$$

显然 $f(x_1+1,x_2,\cdots,x_n)$为 $p_6(x_1,\cdots,x_n,f(x_1,\cdots,x_n),t)=0$ 关于 t 的最小自然根.

故此,$f(x_1,\cdots,x_n)$为代数递归函数,显然也是全函数. □

一个谓词是代数递归谓词,如果其特征函数是代数递归函数而且是全函数.

显然,代数递归谓词对于逻辑连接词¬,∨,∧和→是封闭的.

3 代数递归谓词的永真性证明的反向归约方法

Boyer 和 Moore 给出了基于数学归纳法的一种启发式的反向归约方法,并建造了 BMTP 系统.[6]但是 BMTP 的运行效率并不理想.启发式带来了不完备性,也未能从根本上避免组合爆炸.[7]这里作者给出了一种不用启发式的算法.

证明谓词 $p(x_1,\cdots,x_n)$是永真的,即证明其特征函数 $f(x_1,\cdots,x_n)\equiv 0$. 对于代数递归谓词 $p(x_1,\cdots,x_n)$其特征函数 $f(x_1,\cdots,x_n)$具有下列形式:

$$\begin{cases} f(0,x_2,\cdots,x_n)=g(x_2,\cdots,x_n) \\ f(x_1+1,x_2,\cdots,x_n)\text{为 } p(x_1,\cdots,x_n,f(x_1,\cdots,x_n),t)=0 \text{ 关于 } t \text{ 的最小自然根} \end{cases}$$

根据数学归纳法,证明 $f(x_1,\cdots,x_n)\equiv 0$ 可由 2 步完成:

1° 证明 $f(0,x_2,\cdots,x_n)=g(x_2,\cdots,x_n)\equiv 0$;

2° 假设 $f(x_1,\cdots,x_n)\equiv 0$,证明 $f(x_1+1,x_2,\cdots,x_n)\equiv 0$.

如果能证明,$p(x_1,\cdots,x_n,0,0)\equiv 0$,即完成了归纳过程.

故此,我们给出了下列的证明 $f(x_1,\cdots,x_n)\equiv 0$ 的反向归约方法:

1° 证明 $f(0,x_2,\cdots,x_n)=g(x_2,\cdots,x_n)\equiv 0$;

2° 证明 $p(x_1,\cdots,x_n,0,0)\equiv 0$.

基于上述论述,我们给出了证明 $f(x_1,\cdots,x_n)\equiv 0$ 的算法 *Reduction*(*f*):

Reduction(*f*): *Boolean*;

if $n=1\wedge f(0)=0\wedge$ *proof*($p(x,0,0)$) then return(True) else return(False);

if $n>1\wedge$ *Reduction*(g) $\wedge$ *proof*($p(x_1,\cdots,x_n,0,0)$)

then return(True)

else return(False);

其中 *proof*()为一个证明多项式恒等于 0 的算法.

假设多项式 $f(x_1,\cdots,x_n)$,$(n>1)$具有下列形式[8]:

$$\begin{cases} f(0,x_2,\cdots,x_n)=g(x_2,\cdots,x_n) \\ f(x_1+1,x_2,\cdots,x_n)=f(x_1,\cdots,x_n)+h(x_1,\cdots,x_n) \end{cases}$$

其中 $h(x_1,\cdots,x_n)$ 中 x_1的次数比 $f(x_1,\cdots,x_n)$ 中 x_1的次数少 1,或 $h(x_1,\cdots,x_n)$退化为 $h(x_2,\cdots,x_n)$. $f(x_1)=a_0x_1^m+\cdots+a_m$ (当 $n=1$ 时).

proof(*f*): *Boolean*;

if $n=1\wedge a_0^2+\cdots+a_m^2=0$ then return(True)

else return(False);

if $n>1\wedge f=0$ then return(True);

if $n>1\wedge$ *proof*(g) $\wedge$ *proof*(h)

then return(True)

else return(False);

例如:$\begin{cases} f(0)=1 \\ f(x+1)=2f(x) \end{cases}$

证明:$f(x)=2^x$,即证明　$f(x)-2^x\equiv 0$

令　$F(x)=f(x)-2^x$,则 $F(x+1)-2F(x)=0$, $F(0)=0$,

故$\forall\ {}_xF(x)=0$,即$\forall\ {}_xf(x)=2^x$. □

4 结　论

为了避免采用纯粹的组合式推理规则而导致的组合爆炸,作者研究了基于数学归纳法的反向归约式推理方法.结合代数和递归函数理论,定义了代数递归谓词,给出了证明代数递归谓词永真性的反向归约方法及相应的多项式复杂性算法.更重要的是代数递归函数和代数递归谓词均可以从示例中得到归纳.

致谢　感谢刘叙华教授、石纯一教授和洪家荣教授对作者工作的支持和指教.

参考文献

1　刘叙华.基于归结方法的自动推理.北京:科学出版社,1994.
2　吴文俊.几何定理机器证明的基本原理.北京:科学出版社,1984.
3　《数学手册》编写组.数学手册.北京:高等教育出版社,1979.
4　Peter R. Rekursive Funktionen. Akademiai Kiado, Budapest. 1951.(莫绍揆译.递归函数论.北京:科学出版社,1958.)
5　莫绍揆.递归论.北京:科学出版社,1987.
6　Boyer R S, Moore J S. A computational logic. New York: Academic Press, 1979.
7　石纯一等.人工智能原理.北京:清华大学出版社,1993.
8　李爱中.模型发现和智能决策支持系统工具研究[博士论文].哈尔滨工业大学,1991.

BACKWARD REDUCTION METHOD FOR AUTOMATED THEOREM PROVING

Li Aizhong　　Huang Houkuan

(*Department of Computer Science　Northern Jiaotong University　Beijing*　100044)

Qiao Peili

(*Department of Computer Science　Harbin University of Science and Technology　Harbin*　150040)

Abstract　Based on algebra and recursive function theory, a key concept of algebraic recursive predicates is defined in this paper. Based on mathematical induction, a backward reduction method and its corresponding algorithm reduction are given for proving the universal truth of algebraic recursive predicates. Because the method is reduction based, the efficiency of theorem-proving is improved greatly.

Key words　Automated theorem proving, algebra, recursion, backward reduction, mathematical induction.