重写技术 自动定理证明

计算机科学1992 Vol.19 No.2

(15)

基于重写技术的自动定理证明*)

79-80,24 张 健（中国科学院软件研究所，北京100080） TP18

摘 要

重写技术是处理等式理论的一种有效方法，它已成功地应用到带等词的一阶谓词逻辑中定理的自动证明。本文介绍基于重写的定理证明方法的基本思想，以及几种具体的证明技术，最后将这类方法与经典的归结证明方法加以比较。

近几年来，基于重写技术的定理证明方法越来越受到重视。人们发现，重写技术不仅可用于处理等式理论，还可用来进行一般的定理证明，特别是它能有效地处理带等词的一阶逻辑中的命题。本文将介绍这类定理证明方法的基本思想以及几种具体的做法。

一、项重写系统及其完备化

项重写系统是描述项代数上不确定计算的一种模型，关于它的详细介绍，可参看〔2〕，〔3〕，〔8〕等文献。简单地说，一个项重写系统由有限条重写规则构成，每条规则的左边是一个较复杂的项，右边是一个较简单的项。根据这些规则，可以逐步地将复杂的项归约（或叫重写）成与之等价的简单的项。若一个项已经是最简单形式，不能再归约了，则称为不可归约项。一个项的范式是指与它等价的一个不可归约项。项重写系统的计算行为表现在，对任意的项，通过一步步的重写（即归约），求出该项的范式。

要做到这一点，项重写系统必须具备"完备性"。只有在完备的重写系统中，才能对任意的项，在有限步内将它归约为范式；而且这样的范式是唯一的，与所采用重写规则的顺序无关。

完备化过程[10]的目的就是从给定的等式公理集合出发，试图得到一个与之等价的具有完备性的项重写系统，当然这一点并不总是能做到。该过程的基本操作有如下三种：（1）将等式定向为重写规则，使规则左部的项比右部大（即复杂）。（2）在两条规则上进行迭加，得到新的等式。迭加操作实际上是找一个项t，使它经两条规则归约后得到不同的项t_1和t_2，于是有等式$t_1=t_2$。项对$\langle t_1, t_2\rangle$称为临界对。（3）将新等式定向为新的规则之后，利用它对已有的其它等式和规则进行化简，化简的办法是对两个项进行重写。

上述三种操作构成"重写证明法"中的基本步骤。特别地，迭加操作是明显的等式推理规则。

二、等式定理的证明

等式逻辑在计算机科学和数学的很多领域中占有重要地位。例如，很多抽象代数系统和形式逻辑系统的公理可用一组等式描述。在软件的形式化规格说明技术中，一种重要方法是用项代数上的等式来定义抽象数据类型。在这些场合下，一件很基本的事就是，给定一个等式集合E，判断任意的等式$s=t$是否是E的定理。

判定等式定理的一个直接方法是，先用完备化过程将E转化为一个完备的重写系统R。利用R求出s和t关于R的范式，若这两个

*)八六三计划资助课题

范式相同，则s=t是E的定理，否则不是。

但是，完备化过程并不总是能得到一个完备的重写系统，它有可能失败。即使这样，也能证明一些定理。比如说，在执行完备化过程中，产生了临界对$\langle t_1, t_2\rangle$，那么$t_1=t_2$就是一条定理。

Hsiang和Rusinowitch[7]提出了一种完备的等式定理证明方法，从理论上讲，它能证明所有的等式定理。该方法如下：对于要证明的等式s=t，先将其两端Skolem化，用常数代替全称变量，得到项$\overline{s}$和$\overline{t}$。假定等式不成立，将$\overline{s}\neq\overline{t}$加入到等式公理集合E中，再对这些等式和不等式执行一个类似于完备化过程的过程。在该过程中，新产生的规则不仅用来化简其它等式和规则，也用来化简不等式；由不等式可以推出不等式；另外还可能用到等式x=x(自反公理)。如果在过程执行时，某个不等式的两端被化简为相同的项，则得出矛盾，于是原来的等式被证明为定理。

三、基于重写的定理证明方法

重写技术和完备化过程不仅可用于等式定理的证明，还可推广到（带等词）一阶逻辑中的定理证明。其基本思想是，将“证明”p是定理这件事转化为“证明p=true”成立。下面我们介绍一些具体的证明方法。

1. Hsiang 方法

在J. Hsiang的方法[5],[6]中，先将要证的公式Φ取反，Skolem化，变成合取范式$C_1\wedge C_2\wedge\cdots\wedge C_n$，再按下述规则将每个$C_i$变成布尔环上的多项式$M_i$：

$$x\vee y\longrightarrow x*y+x+y$$
$$x\wedge y\longrightarrow x*y$$
$$x\supset y\longrightarrow x*y+x+1$$
$$x\equiv y\longrightarrow x+y+1$$
$$\neg x\longrightarrow x+1$$

（这里，+、*分别表示逻辑运算“异或”、“与”，而0、1分别表示false、true。）

假定Φ不成立，即$\neg\Phi=M_1*M_2*\cdots*M_n=1$，亦即，每个$M_i=1$，于是得到一组重写规则$\{M_i\rightarrow 1 \mid i=1, 2, \cdots n\}$。将这组规则与下面的布尔环上的重写系统合在一起作为输入。

$$x+0\longrightarrow x$$
$$x+x\longrightarrow 0$$
$$x*0\longrightarrow 0$$
$$x*1\longrightarrow x$$
$$x*x\longrightarrow x$$
$$x*(y+z)\longrightarrow (x*y)+(x*z)$$

对这组输入规则，执行完备化过程。如果在此过程中产生规则1→0，则得到矛盾，定理得证。如果该过程终止时还没有得出矛盾，则说明Φ不是定理。当然该过程也可能不终止。

2. Gröbner 基方法

Gröbner基（Gröbner basis）算法[1]本来是用于研究多项式理想的。Kapur和Narendran[9]将它用于定理证明。同Hsiang方法一样，该方法也是先将一阶逻辑公式变成布尔环上的多项式，再变成重写规则。所不同的是，Hsiang方法中每条规则右部都是1（或0），而在这里，规则右端可为一个比左端小的多项式。例如，它将子句$q(x, y)\vee\neg q(y, x)$变成规则$q(x, y)*q(y, x)\rightarrow q(y, x)$，而不是$q(x, y)*q(y, x)+q(y, x)\rightarrow 0$。另外，这里用的推理规则是Buchberger的Gröbner基方法，而不是Knuth和Bendix的迭加操作。

3. 等式子句迭加法

L. Fribourg[4]将重写和迭加结合到经典的归结证明法中，得到所谓的“等式子句迭加法”（SEC）。它将子句变为等式子句（E-子句）；同上面一样，也将关系谓词看成布尔函数（其值为true或false）；推理规则采用“子句迭加法”。欲证明一个定理，如 f(x, x, y)=x，则将它取反，Skolem化，得到E-子句：E(f(a, a, b), a)=false，加入原来的前提集合。若从这组子句集推出□（矛盾），则定理得证，本方法中要用到

（转第24页）

识工程提供一种新的工具。用ALT类型理论可以将软件开发过程形式化，建立一种“开发演算”；ALT类型理论支持时态逻辑，模糊逻辑，多值逻辑，尤其在描述变换式软件开发方法，保证变换正确性方面，ALT类型理论已经取得了实际的成果。

诚然，类型理论及其方法亦有自身的缺点。由于类型理论方法属于形式化方法的一种，因而形式化方法所带有的诸如固有复杂性、表达繁琐、低效等局限性和缺点在类型理论中也打下了深深的烙印。

Goguen教授在[1]中指出今后几年值得重视的研究方向是软件开发方法和程序证明方法。我们认为，类型理论及其程序设计方法学是沿着这两个方向推动软件科学前进的可取途径之一。（参考文献略）

（接第80页）

自反公理：E (x, x) = true。

4、基于条件重写的子句迭加法

Zhang和Kapur[11]将条件重写规则引入定理证明过程中，单位子句（即，只含一个文字）变为无条件规则，非单位子句（若干个文字之或）变为条件规则。如，将子句 ¬q (a, a) 和q (x, y) ∨ ¬q (y, x) 分别变成规则 q (a, a) →0，以及 q(x, y) →1, if q (y, x) 。

这种方法中的推理规则也称为子句迭加。与Fribourg 方法所不同的是，这里的迭加是在两个子句的最大文字之间进行，而不是在最右文字之间进行。另外，这里不用自反公理。

小结 由上面可看出，重写证明法的一般过程是：将待证明的定理取反，Skolem化，变成一定的规范形式；再对它和公理集进行推理，直至得出矛盾。在以上几种具体方法中，Hsiang方法简单直观，影响较大，对它的研究较多，得到了很多有效的策略，而且用Hsiang方法得到的证明也较简洁。

四、与归结法之比较

重写证明法的大致过程与归结法相似，其基本操作也很相似。迭加可看成是归结再加上调解 (paramodulation)，临界对相当于归结式。但一般说来，迭加和重写比经典方法中的归结和删除策略从概念上讲要更广泛。可以证明，有一组子句集合，它的基于重写的否证过程比基于归结的过程要短(即，所含的步骤更少）。

当然，从时间耗费上看，重写法并不一定优越，因为每步迭加操作中要进行一致化和归约，而每步归结只需要一致化。

完备化过程、归结原理以及Gröbner基算法具有很多相似的特征，可置于一个统一的框架之中[1]。实际上，归结法和重写法可以互相借签对方的很多有益的东西。

五、 结束语

重写技术为定理证明提供了一条很有潜力的途径，用基于重写的证明方法已经解决了不少难度颇高的问题。（可参看 Journal of Automated Reasoning, Association of Automated Reasoning Newsletter 以及 Journal of Symbolic Computation1991年的专辑。）并且出现了一些实际的证明系统，如 RRL (Rewrite Rule Laboratory)、LP (Larch Prover)、HIPER (High Performance Rewriting)等等。在一些使用经典方法的系统（如OTTER）中也加进了项重写的一些机制。目前，人们正探讨更有效的重写证明策略，使重写方法能证明更多更难的定理。（参考文献略）