计算机科学 1997 Vol. 24 №. 6

线性逻辑式程序设计:理论与应用

Linear Logic Programming:Theory and Applications

倪德明　姚卿达

(中山大学软件研究所　广州 510275)

摘　要　线性逻辑作为一种动作逻辑,有很强的表达能力和很好的可构造性,线性逻辑式程序设计语言为表达操作语义、并发规范等提供了新的工具和框架,Girard 对线性逻辑所作的"定义"概念的扩充对状态变迁系统中模拟和双模拟提供了证明论框架,本文综述了这方面的成果。

关键词　证明论,并发变迁系统,线性逻辑,程序设计

近年来,许多研究人员使用 Gentzen 矢列系统作为研究(线性)逻辑式程序设计的框架,在此框架下,矢列用来表示计算的状态,无 CUT 证明(cut free proofs)的构造过程则反映计算的动态特征。所以线性逻辑式程序设计语言的研究往往包括证明论(而非模型论)性质研究和表达能力(或应用)研究。

1　证明论研究及其成果

逻辑式程序设计语言的设计要求"目标驱动"搜索策略必须是完备的。对矢列系统而言,"目标驱动"证明搜索及其在计算模型上的解释有下述特点:

• 对象系统必须有 CUT 消去定理(cut-elimination theorem),

• 证明树中的矢列 $\Gamma\rightarrow G$ 表示解释器处于这样的一种状态:它的下一步求解目标是由程序 Γ 得到目标 G。初始求解目标为终矢列(end sequent),即待证矢列,构造过程由底向上(由树根到树叶)。

• 构造过程中右部引入规则优先使用,仅当矢列右部为原子公式时才使用左部引入规则,如经典逻辑中,矢列 $p_1\vee p_2\rightarrow p_1\vee p_2$ 根据右部规则优先使用的原则得到证明树片断:

$$\dfrac{\dfrac{p_1\rightarrow p_1 \quad p_2\rightarrow p_2}{p_1\vee p_2\rightarrow p_1}\vee L}{p_1\vee p_2\rightarrow p_1,p_2}WR(i=1,2)$$
$$\dfrac{}{p_1\vee p_2\rightarrow p_1\vee p_2}\vee R$$

因而经典逻辑的"目标驱动"搜索是不完备的。

• 能够定义向后链(back-chaining)推理规则以替代所有的左引入规则而不失去搜索的完备性。如 Horn 子句逻辑中,直觉主义中的左引入规则

$$\dfrac{\Gamma\rightarrow A,G \quad \Gamma,B\rightarrow G}{\Gamma,A\Rightarrow B\rightarrow G}\Rightarrow L$$ 替换为:

$$\dfrac{\Gamma\rightarrow\theta A}{\Gamma,A\Rightarrow B\rightarrow G}BC$$

其中 θ 为一替换满足 $G=\theta B$。

完备性的要求指任何可证矢列都能由"目标驱动"策略构造出它的一个证明,由于 $p_1\vee p_2\rightarrow p_1\vee p_2$ 在经典中可证,因而经典逻辑没有完备的"目标驱动"搜索策略。

那么什么样的子逻辑才有完备的"目标驱动"证明搜索策略呢?

Miller 等[33]首先在直觉主义逻辑中对"目标驱动"搜索这一要求进行了形式化,从证明论角度提出了一致证明(uniform proofs)的概念:一个目标驱动证明或一致证明指一无 CUT 证明,其中每个矢列的出现如果其右部为非原子公式,则该矢列出现必为右部引入规则的结论。一个抽象逻辑式程序设计语言可定义为一个一致证明为完备的逻辑系统,即对该系统中的每个证明都有一个一致证明在规则置换的意义上与之等价。然后通过对证明规则可置换性的分析发现,Horn 子句逻辑的目标驱动搜索在直觉主义可证性的含义上是完备的[33],并证明:由蕴含联结词、合取联结词、全称联结词及零元联结词"真"生成的直觉主义逻辑的子逻辑(称为 hereditary Harrop formula)仍有完备的"目标驱动"证明搜索。

由于直觉主义逻辑中的矢列是单结论的,即只有一个待证目标,因而一致证明的概念及其证明规则可置换性的分析技术很容易推广到直觉主义线性逻辑上(仍然是单结论的),Hodas 和 Miller 基于此

设计了线性逻辑式程序设计语言 Lolli[24,22]，实际上是 λProlog 的一个模块式扩展：程序子句可被使用一次或任意次。线性逻辑的联结词可提供对使用次数的控制。

对线性逻辑而言，矢列结论有多个公式，因而待证目标为一多重集，集中每个元素的可证性可能是互相关联的。在此情况下，目标驱动和一致证明的概念需要扩展或修改。近年的研究表明，至少有两种成功的方法，一是由 Harland 和 Pym[42,25,27]提出的，它要求在每步证明构造中，待证目标中至少有一个目标是可归约的，可以看出，依此种要求设计的逻辑式语言不能表示并发。另一种要求多个待证目标可同时归约，也即待证目标可以任意顺序归约。Miller 首先用此方法给 π-演算作了一个线性逻辑规格说明[35]。

很重要的一点是：Miller 后来证明，通过选择线性逻辑的一个合适的完全联结词集，整个线性逻辑可用作逻辑式程序设计，线性逻辑的这种描述方法，称为 Forum[36]。

2 线性逻辑式程序设计语言及应用

下述语言使用线性逻辑的有自然操作语义的子逻辑作为语言构架：

· LO (Linear Objects)[3,6]，由 Andreoli 和 Pareschi 设计，可以看成是 Horn 子句的扩充--原子公式扩充为由乘性析取联结的多重集。在 LO 中，向后链(backchaining)推理表现为多重集的重写(multiset rewriting)。该语言用于支持面向对象和有协作进程的程序设计。

· ACL[28,29]由 Kobayashi 和 Yonezawa 设计，是一异步进程演算系统，其中 asend 和 read 原语体现为线性逻辑的互补联结词。

· Lincoln 和 Saraswat 用线性逻辑对并发约束逻辑式程序设计作了扩充[32,44]。

· LIP[38,39]，由倪德明和孙永强设计，可看着 LO 的扩充—逻辑上增加了动态模块传输机制，主要用作过程式和逻辑式程序设计合成的平台和抽象机。

这些语言都使用线性逻辑的一个较小子集，用公式的多重集反映状态，对象结构或进程结构。使用多重集重写来反映继承，同步或变迁。

当然证明论的分析并不是纯粹的关于矢列演算的技术处理，主要是为了扩充逻辑式程序设计语言，使之有更强的表达能力，线性逻辑式语言在许多领域中支持和提供新的规范和技术：

并发：许多语言设计支持(至少部分地支持)并发规范的说明[28,29,32,35,44]。

面向对象程序设计：描述状态和继承是 LO 的早期目标。

操作语义：Forum 和 LIP 成功地用于描述过程式语言(如 C)的操作语义。[39,36,12]。Chirimar 用 Forum 描述了流水线 RISC 处理器的操作语义[12]。

自然语言处理：Lolli 为英语相关子句的处理提供了一种描述式方法[23]。

对象逻辑证明系统：Lolli 可用于改善经典的对对象级自然推理系统的直觉主义规范说明[27]及对象级矢列系统的规范说明[36]。

CLP 和 IP 的合成框架：LLP 支持基于 Horn 子句的并发逻辑式程序设计和过程式程序设计的合成及人-机交互作用[38,40]。

3 并发变迁系统说明

上述应用领域可看成广义的状态变迁系统，线性逻辑方法的基本特点是：变迁系统规范对应一线性逻辑理论。证明树构造过程反映变迁的动态特征，而证明规则的分析在于保证搜索呈"目标驱动"的形式以便有计算意义上的解释并保证其完备性。所以线性逻辑及其证明构造策略模拟了对象级变迁系统，说明了对象系统的操作语义，遗憾的是对对象级系统的一些元性质并未提供更多的说明。

Girard[19]在线性逻辑中引入了"定义"机制，"定义"可看作是一组具有一定形式的非逻辑公理。Schroeder-Heister[45]证明：如果对"定义"的结构加以一定的限制，则由"定义"可引入附加的左部引入规则和右部引入规则而 CUT 消去定理(cut-elimination theorem)仍成立。直觉上，定义机制的引入在逻辑式程序设计中所做的扩充是允许子句体中负原子公式的出现，并且可以通过反驳(refutation)去证明它。

Raymond McDowell 等考虑了下述形式的非逻辑公理：

假定 $p_i(i=1,\cdots,n)$ 为一组谓词，H_i 中仅出现联结词 $1,\top,\otimes,\&,\multimap,\forall,\exists$，且 H_i 中每个自由出现的变元均在 $\bar{t}_i$ 的某个项中自由出现，$\bar{t}_i$ 中每个自由变元均出现于 $\bar{x}_i$ 中，"定义"为下述形式：

计算机科学 1997 Vol. 24№. 6

并发系统的操作时序逻辑描述和验证*)

The Specification and Verification of the Temporal Logic of Action(TLA)for Concurrent Systems

蒋 慧 张兴元 王元元

(中国人民解放军通信工程学院计算机教研室 南京 210016)

摘 要 The Temporal Logic of Action(TLA)is a kind of temporal logic used to specify and to verify concurrent system, it is still in the tradition of assertional methods for reasoning about. In this method, the algorithm of the concurrent system and its properties is expressed by a TLA formula. So, the proof of the assertions which specify the concurrent system's specification and its properties are all in the system of a underlying logic system. TLA is a simply and a relatively compelete temporal logic. This paper introduces the TLA, and describes how it is used to specify and verify concurrent system.

关键词 Temporal logic, Temporal logic of action, Safety, Liveness, Fairness

1. 引言

近年来,为了在开发一个复杂系统的过程中尽量提高系统的正确性,减少开发过程中重复、繁琐的工作,国内外许多学者都从逻辑的角度进行研究,用逻辑系统对并发系统程序进行描述、验证,以期通过逻辑系统的简洁和严密来保证大型并发系统的正确性和精确性。

用时序逻辑来证明并发程序的思想最早是由 Pnueli 在 1977 年提出的,它是一种变迁公理方法(transition-axiom method)。在这种方法中,通过抽象程序和时序逻辑的组合来描述一个并发系统。随着将近二十多年的研究,这种方法经过不断的发展和演化,已经比较完善和成熟,并且逐步从理论的研究转为实践的应用。其中较为成熟和流行的时序逻辑系统有三个流派:Unify logic[Chandy and Misra 1988]、Manna & Pnueli[1991]的 MPTL 和 Lamport 的 TLA[1992]。Lamport 的 TLA 具有前两种系统所不具有的一些优点,本文就其在并发系统的描述和验证中的应用作一简单的讨论。

2. 操作的时序逻辑(TLA)

TLA(*Temporal Logic of Action*)是操作逻辑(logic of actions)和标准的时序逻辑的结合。TLA 的语义及部分规则见图 1 所示。

$\forall \overrightarrow{x_1}[p_1(\overrightarrow{t_1}) \leftarrow H_1]\cdots\forall \overrightarrow{x_n}[p_n(\overrightarrow{t_n}) \leftarrow H_n]$ $(n\geqslant 0)$

则由[45],可定义下述 DR(definitional reflection)规则,而系统的 CUT 消去定理仍成立:

$$\frac{\{\theta\Gamma,\theta H_i\rightarrow\theta C \mid \text{where } \theta \text{ is the mgu of A and } p_i(\overrightarrow{t_i})\}}{\Gamma,A\rightarrow C}\mathrm{DR}$$

Raymond McDowell 等指出[37],上述定义形式能够描述抽象状态变迁系统及 CCS 的线性逻辑规范,并且证明,在不使用 DR 规则的情况下,则状态 p 到状态 q 的可达性对应于矢列 $q\rightarrow p$(或其线性逻辑描述形式)的可推出性。假如使用 DR 规则,模拟和双模拟体现为可推出性。这些成果说明:合适的可算的线性逻辑子集不仅能用于描述抽象的状态变迁系统,而且能用于描述和判断如模拟和双模拟这样的元性质。

致谢:作者感谢 Dale Miller 教授提供的材料以及 P. Lincoln 所维护的线性逻辑 mailinglist,上海交通大学孙永强教授对我们在线性逻辑方面的研究工作给予很多支持。参考文献共 48 篇(略)。

---

*)国家自然科学基金资助项目,项目号:69572041。蒋 慧 硕士生,主要研究方向:协议验证、人工智能及其在网络中的应用。张兴元 副教授,主要研究方向:协议验证和网络。王元元 教授,主要研究方向:数理逻辑、人工智能。