计算机科学 2000Vol. 27№. 2

χ-演算与π-演算的语义比较研究

Research of Semantic Comparison between χ-calculus and π-calculus

徐 林 傅育熙

(上海交通大学计算机科学与工程系 上海 200030)

Abstract Through the comparison of syntactic structure, operational semantics and algebraic semantics between χ-calculus and π-calculus, this paper concludes that χ-calculus has more succinct syntactic structure, more explicit operational semantics, more intuitionistic algebraic semantics and more favorable algebraic property. And a translation from π-calculus to χ-calculus is presented.

Keywords Concurrency theory, Process algebra, Bisimulation

1. 引言

计算模型一直是计算机科学研究的重要问题。与逻辑学家不同,计算机科学家着重于模型的操作语义、代数语义与指称语义。λ-演算是顺序计算的经典模型。与顺序计算不同,我们对并行与并发计算实质的认识尚处于初级阶段。自 Milner 提出 CCS 以来,计算机科学家提出了许多计算模型。U. Engberg 和 M. Nielsen 研究过一并发计算模型[1],提出了 CHOCS 类语言[2]。但这些模型都存在不同程度的问题。为了找出一个可与 λ-演算相比的并发计算模型,Milner 等人随后提出了 π-演算[3],Sangiorgi 在其博士论文[4]及其后的文章中研究了高阶 π-演算。但所有这些语言都只能模拟 Lazy λ-演算的操作语义[5]。为了找到一个能完全模拟 λ-演算操作语义的并发计算模型,傅育熙提出了 χ-演算[6~8],该模型在概念上对 π-演算进行了简化。一阶 χ-演算可模拟 Call-by-name λ-演算的操作语义,高阶 χ-演算可模拟 λ-演算的操作语义。χ-演算的意义在于它既是对 π-演算在概念上的简化,又在表达能力上有所提高。

2. χ-演算与 π-演算的比较

π-演算是九十年代计算机并行理论领域最重要的并发计算模型,它是由 Milner、Parrow 和 Walker 在 Engberg 和 Nielsen 工作的基础上,简单地将通道内容限制为通道名而得到的。由于可以传递通道名,故此模型可以用来描述结构不断变化的并发系统(即系统中进程之间的连接关系可随着进程状态的不断变化而发生改变)。这种通过共享通道名来通讯,并且进程间可传递通道名的方法使 π-演算具有比 CCS 更强的表达能力,具有描述可移动性的能力。

χ-演算是一种全新的并发计算模型,它建立在 π-演算的基础之上,是对 π-演算研究工作的继续和发展,其基本思想与 π-演算相同。但与 π-演算相比,它有如下特点:1)取消了抽象名(abstract name),将它和局部名合二为一;2)将输入和输出操作前缀统一为形如 a[x]的前缀,这样在该演算中名的输入操作可通过名的受限方式来完成,即将 π-演算中的输入进程 a(x).P 表示成(x)a[x].P,因而 χ-演算的通讯具有对称的性质;3)χ-演算中通过局部化操作来描述通讯的代数语义,使互模拟这一重要概念的定义更简洁、证明更方便、性质更良好。在 χ-演算中由于取消了 π-演算的顺序操作进程 a(x).P,因此很大程度地提高了其演算的并发能力。χ-演算用信息交换的通讯机制代替了 π-演算中进程之间传值的通讯机制。

2.1 语法结构的比较

π-演算的语法结构。设 N 表示名(name)的集合,名由小写字母表示,$\overline{N}\triangleq\{\overline{a}\mid a\in N\}$,$a\in N\cup\overline{N}$,P,Q 表示 π-演算进程,x 表示名,则 π-演算进程可定义为:

$$P::=0\mid a(x).P\mid\overline{a}y.P\mid P+Q\mid P\mid Q\mid(x)P\mid[x=y]P$$

其意义的解释如下:

· 0 表示非活动进程,它不能与任何进程进行交互。

· a(x).P:这里 a 指某一通道的输入端,此进程的行为是先通过通道名 a 从通道的另一端接收名 y,再

徐 林 硕士生,研究方向:并发计算模型。傅育熙 教授,博士生导师,研究方向:并行理论、类型理论。

激活进程 P[y/x]，其中 x 为局部名，a(x).0 常简化为 a[x]。

· $\bar{a}y.P$：这里$\bar{a}$指某一通道的输出端，此进程的行为是先通过通道名$\bar{a}$向通道的另一端输出名 y，再激活进程 P，其中 y 为全局名。

· P+Q：这是进程的不确定计算形式，此进程将根据一定的诱因来激活进程 P 或 Q，它的行为仅为这个被激活的进程的行为。

· P|Q：这是一个并行操作形式的进程，此式中进程 P 和 Q 并行地存在，它们可以独立地同其他的进程进行交互操作，也可以彼此通过共享的通道进行通信。

· (x)P：该进程中 x 是进程 P 的局域名，它使得进程 P 不能通过通道 x 与其他进程进行通信。

· [x=y]P：这是一个条件进程，如果 x=y，它将激活进程 P，否则将不进行任何操作，成为非活动进程 0。

χ-演算语法结构和 π-演算的语法结构相似，χ-演算进程可定义为：

P::=0|a[x].P|P+Q|P|Q|(x)P|[x=y]P

其不同之处如下：

· a[x].P：这里 a 的形式为 m 或$\bar{m}$，其中名 m 和$\bar{m}$分别指某一通道的两端，不再区分输入端与输出端。此进程的行为是先通过通道名 a 同其他共享同一通道的进程进行一次通信，接收名 y，再激活进程 P[y/x]，或将名 x 传出，然后再激活进程 P。a[x].0 常简化为 a[x]。其中 x 为全局名。

· (x)P：该进程中 x 是进程 P 的局部名，它使得进程 P 不能通过通道 x 与其他进程进行通信。注意 χ-演算中只含有这一类局部名。

由以上定义可以分析比较得到：1)χ-演算统一了通道的输入端与输出端。χ-演算中，a[x].P 和$\bar{a}$[x].P 对应于 π-演算中的 a(x).P 和$\bar{a}$x.P，所不同的是，a[x].P 和$\bar{a}$[x].P 中的 x 均为全局名。2)在 π-演算中存在两类局部名，如 a(x).P 中的 x 不同于(x)$\bar{a}$x.P 中的 x。χ-演算没有前类局部名，输入和输出是对称的，在概念上也更简单。我们认为作为一种基本的并发计算模型，两类局部名似乎多了些。χ-演算之所以可以只含一类局部名，是基于对通讯这一基本概念的进一步认识之上的。传统认为通讯是一种传值操作，而我们认为通讯是一种信息交换机制。3)经过以上简化，并没有消弱 χ-演算的表达能力。π-演算中的 a(x).P 很容易由 χ-演算中的(x)a[x].P 来表示。这里值得指出的一点是：χ-演算是在 π-演算的基础上扩展而得，但这种扩展并不是在 π-演算中加入某种额外的操作子，而是对 π-演算的进一步简化。我们认为，如果简化了一个模型以后，还能够增强其表达能力，那么这种简化是必要的。

2.2 操作语义的比较

π-演算的操作语义。对于进程 P，$In(P)$表示它的局部名集合，$gn(P)$表示它的全域名集合，进程 P 的名集合 $n(P)$为集合 $In(p)$和 $gn(P)$的并。下面的规则定义了 π-演算的操作语义：

$$\text{OUTPUT-ACT}:\frac{}{\bar{x}y.P\xrightarrow{\bar{x}y}P}$$

$$\text{INPUT-ACT}:\frac{}{m(x).P\xrightarrow{m(y)}P[y/x]}\ y\notin fn((x)P)$$

$$\text{TAU-ACT}:\frac{}{\tau.P\xrightarrow{\tau}P}$$

$$\text{MATCH}:\frac{P\xrightarrow{a}P'}{[x=x]P\xrightarrow{a}P'}$$

$$\text{SUM}:\frac{P\xrightarrow{a}P'}{P+Q\xrightarrow{a}P'}$$

$$\text{PAR}:\frac{P\xrightarrow{a}P'}{P|Q\xrightarrow{a}P'|Q}\quad bn(a)\cap fn(Q)=\Phi$$

$$\text{COM}:\frac{P\xrightarrow{\bar{x}y}P'\quad Q\xrightarrow{x(z)}Q'}{P|Q\xrightarrow{\tau}P'|Q'[y/z]}$$

$$\text{CLOSE}:\frac{P\xrightarrow{x(w)}P'\quad Q\xrightarrow{\bar{x}(w)}Q'}{P|Q\xrightarrow{\tau}(w)P'|Q'}$$

$$\text{RES}:\frac{P\xrightarrow{a}P'\quad x\notin n(a)}{(x)P\xrightarrow{a}(x)P'}$$

$$\text{OPEN}:\frac{P\xrightarrow{\bar{x}y}P'\quad x\neq y\ w\notin fn((y)P')}{(y)P\xrightarrow{\bar{x}(w)}P'[w/y]}$$

χ-演算的操作语义。下面的规则定义了 χ-演算的操作语义。为了保证操作语义的完整性，我们引入了不完全通讯[y/x](update)，主要起一个辅助作用。其中 N 为名集，$\alpha\in N\cup\bar{N}$，$\mu\in\{\tau\}\cup\{\alpha[x],\alpha x|x\in N\}$，$\delta\in\{\tau\}\cup\{\alpha[x],\alpha x,[y/x]|x,y\in N\}$。为简单起见，我们省略了对称规则。

$$\text{Sequentialization}:\frac{}{m[x].P\xrightarrow{m[x]}P}S_0$$

$$\text{Composition}:\frac{Q\xrightarrow{\mu}Q'}{P|Q\xrightarrow{\mu}P|Q'}P_0$$

$$\frac{Q\xrightarrow{m[y/x]}Q'}{P|Q\xrightarrow{m[y/x]}P[y/x]|Q'}P_1$$

$$\frac{P\xrightarrow{ax}P'\quad Q\xrightarrow{\bar{a}[x]}Q'}{P|Q\xrightarrow{\tau}P'|Q'}C_0$$

$$\frac{P\xrightarrow{ax}P'\quad Q\xrightarrow{\bar{a}x}Q'\quad x\notin gn(P|Q)}{P|Q\xrightarrow{\tau}(x)P'|Q'}C_1$$

$$\text{Communication}:\frac{P\xrightarrow{a[x]}P'\ Q\xrightarrow{\bar{a}[x]}Q'}{P|Q\xrightarrow{\tau}(P'|Q')}C_2$$

$$\frac{P\xrightarrow{a[x]}P'\ Q\xrightarrow{\bar{a}[y]}Q'\ x\neq y}{P|Q\xrightarrow{\bar{y}/x}P'[y/x]|Q'[y/x]}C_3$$

$$\frac{P\xrightarrow{\delta}P'\ x\notin n(\delta)}{(x)P\xrightarrow{\delta}(x)P'}L_0$$

$$\frac{P\xrightarrow{a[x]}P'\ x\notin\{a,\bar{a}\}}{(x)P\xrightarrow{ay}P'[y/x]}L_1$$

$$\text{Localization}:\frac{P\xrightarrow{[y/x]}P'}{(x)P\xrightarrow{\tau}P'}L_2$$

$$\text{Summation}:\frac{P\xrightarrow{\delta}P'}{P+Q\xrightarrow{\delta}P'}Sum$$

$$\text{Condition}:\frac{P\xrightarrow{\delta}P'}{[x=x]P\xrightarrow{\delta}P'}C$$

由以上定义可以分析比较得到：χ-演算中以 τ 表示一次完全通讯的发生。如规则中的 C_0、C_1、C_2，以[y/x](update)表示一次不完全通讯的发生，如 C_3，再通过规则 L_2 使其成为完全通讯 τ。由于在这些通讯名中不存在局部名，所以这里不需要边侧条件。由此，我们可以看出 χ-演算中对通讯机制的刻画更直观、简洁、深刻。值得一提的是：我们认为并发计算模型的本质是对通讯机制的描述。χ-演算则比 π-演算更好地刻画了通讯机制的本质。即通讯就是将局部名进行替换，以达到信息交换的目的。并且这种交换或者由一个全局名替换一个局部名，或者将两个局部名统一为另一个局部名。直观上，可以认为通讯的结果是信息共享。由此，我们认为通讯机制是一种信息交换机制，从而有别于传统认为的传值操作机制。同时，我们可以引出通讯发生的两条充要条件：1）两个进程连接在同一通道的两个端口上。2）两个对应端口中的通讯名至少有一个是局部的。下面这个规则可形象地表示出 χ-演算中通讯的实质：

$$(x)(R|a[x].P|\bar{a}[y].Q)\xrightarrow{\tau}R[y/x]|p[y/x]|Q[y/x]$$

这里 $x\neq y$.

下面这个通讯在 π-演算中不可发生，但在 χ-演算中是可以的：

$$(y)(R|a[y].P)\xrightarrow{ax}R[x/y]|P[x/y]$$ 这里 $x\neq y$.

2.3 代数语义的比较

对并发计算模型代数语义的研究中，最重要的就是对互模拟(Bisimulation)这一重要概念的刻画。

π-演算中的互模拟定义。进程集上的二元关系 φ 是一个模拟当且仅当对进程 P 和 Q，如果 PφQ，那么以下三式成立：

1）如果 $P\xrightarrow{a}P'$ 且 a 为自由名，那么存在一个 Q'，满足 $Q\xrightarrow{a}Q'$ 且 $P'\varphi Q'$。

2）如果 $P\xrightarrow{x(y)}P'$ 且 $y\notin n(P,Q)$，那么存在一个 Q'，满足 $Q\xrightarrow{x(y)}Q'$ 且对任意名 w，$P'[w/y]\varphi Q'[w/y]$。

3）如果 $P\xrightarrow{\bar{x}(y)}P'$ 且 $y\notin n(P,Q)$，那么存在一个 Q'，满足 $Q\xrightarrow{\bar{x}(y)}Q$ 且 $P'\varphi Q'$。

关系 φ 是一个互模拟当且仅当 φ 与其逆均为模拟。进程集上的二元关系 ∽(互模拟)定义为：若 P∽Q 则存在一个互模拟 φ，满足 PφQ。

χ-演算中互模拟定义。进程集上的二元关系 φ 是一个局部模拟当且仅当对进程 P 和 Q，如果 PφQ，那么对任意进程 R 和名向量 $\vec{x}$，下式成立。

如果 $(\vec{x})(P|R)\overset{i}{\Rightarrow}P'$，那么存在一个 Q' 满足 $(\vec{x})(Q|R)\overset{i}{\Rightarrow}Q'$ 并且 $P'\varphi Q'$。

关系 φ 是一个局部互模拟当且仅当 φ 与其逆均为局部模拟。进程集上的二元关系 ≈(局部互模拟)定义为：若 P≈Q 则存在一个局部互模拟 φ，满足 PφQ。

由以上定义可以分析比较得到：χ-演算对互模拟这一重要概念的定义更容易理解。即考察两个 χ-进程 P 和 Q 是否互模拟，我们只需要将它们放入相同的环境中(以 R 和 $\vec{x}$ 来表示)，进一步观察这两个进程是否具有完全相同的通讯行为。但是，π-演算中互模拟的定义则显得过于复杂，考察起来也更加困难。研究其代数性质时，又不可避免地陷入了烦琐的替换等操作中，进而影响人们对其代数性质本质的认识。值得指出的是：χ-演算中的 ≈ 是一个同余关系，具有良好的代数性质。但是，π-演算中的 ∽ 不是同余关系，它对替换操作不封闭，进而不可在输入前缀操作下保持其原有的性质。

3. 由 π-演算到 χ-演算的翻译

我们已经指出 χ-演算具有比 π-演算更强的表达能力。这里我们给出一个简单的由 π-演算到 χ-演算的翻译。下面以 $a(x).P$，$\bar{a}x.P$，$(x)P$，$P+Q$，$P|Q$ 表示 π-进程，以 $\overline{a(x).P}$，$\overline{ax.P}$，$\overline{(x).P}$，$\overline{P+Q}$，$\overline{P|Q}$ 表示翻译后的 χ-进程。

$$\overline{a(x).P}\equiv(x)a[x].\bar{P}\qquad \overline{ax.P}\equiv a[x].\bar{P}$$

$$\overline{(x).P}\equiv(x)\bar{P}\qquad \bar{0}\equiv 0$$

$$\overline{P+Q}\equiv\bar{P}+\bar{Q}\qquad \overline{P|Q}\equiv\bar{P}|\bar{Q}$$

由此，我们可以看出：π-进程实际上是 χ-进程的一个子进程，π 语言也是 χ 语言的一个子语言。

结束语 通过以上各点的研究。我们发现：χ-演算虽然是在简化 π-演算的基础上获得的，但是它具有比 π-演算更加简洁的语法结构、更加明确的操作语义、更加直观的代数语义、更加良好的代数性质、更加强大的表达能力等等。因此，我们有理由认为：χ-演算比 π-演算更有资格充当"并发计算的 λ-演算"这一角色。

参 考 文 献

1 Engberg, U, Nielsen M. A Calculus of Communicating Systems with abel-Passing. [Report DAIMI PB-208]. Computer Science Department, University of Aarhus, 1986

2 Thmsen B A Theory of Higher Order Communicating Systems. Information and Computation, 1995, 116: 38～57

3 Milner R, et al. A Calculus of Mobile Process. Information and Computation, 1992, 100: 1～77

4 Sangirgi D Expressing Mobility in Process Algebras: First-Order and Higher Paradigms: [PhD Thesis] Department of Computer Science, University of Edinburgh, 1993

5 Milner R. Functions as Processes. Cambridge Univ. Journal of Mathematical Structures in Computer Science, 1992, 2

6 Fu Yuxi. The χ-Calculus. In, Proc of the Intl Conf on Advances in Parallel and Distributed Computing IEEE Computer Society Press, 1997. 74～81

7 Fu Yuxi, a Proof Theoretical Approach to Communications ICALP' 97, July 7th-11th, Bologna, Italy, Lecture Notes in Computer Science 1256, Springer Verlag, 1997. 325～335

8 Fu Yuxi. Variation on Mobile Processes. Theory Computer Science. Elsevier Science Publisher, 1998

9 Fu Yuxi. Bisimulation Lattice of χ-Processes. Submitted for publication, 1998

10 Fu Yuxi. Symmetric π-Calculus. Journal of Computer Science and Technology, 1997

11 Sangiorgi D. A Theory of Bisimulation for π-Calculus. LNCS 751, 1993. CONCUR' 93

12 Milner R. Communication and Concurrency, Prentice Hall, 1989

（上接第5页）

法有目的地进行。

4）人们意识到，大脑相当复杂，逻辑思维和形象思维同等重要，因此、具有归纳和演绎双重功能的混合型逻辑系统的运行机制会受到广泛的重视。

参 考 文 献

1 石纯一，黄昌宁、王家钦. 人工智能原理. 北京：清华大学出版社，1995

2 Rumelhart D E, et al. Learning Representation by Back-propagation Errors. *Nature*, 1986, 323(6188): 533～536

3 Holland J H. Adaptation in Natural and Artificial Systems. MIT Press, 1992

4 Bui T N, Moon B R. Genetic Algorithm and Graph Partitioning. IEEE Trans. on Computers, 1996, 45(7): 841～855

5 董聪. 广义遗传算法. 大自然探索，1998，17(1). 33～37

6 董聪，郭晓华. 智能计算中的热点问题. 计算机科学，1999，26(4)：5～9

7 董聪，郦正能，夏人伟，何庆芝. 多层前向网络研究进展及若干问题. 力学进展，1995，25(2)：185～196

8 Honik K. Approximation Capabilities of Multilayer Feedforward Networks. Neural Networks, 1991, 4: 551～557

9 董聪. 前向网络全局最优化问题研究. 中国科学基金，1997，(1)：23～29

10 董聪. 多层前向网络的逼近与泛化机制. 控制与决策，1998，13(增刊)：413～417

11 董聪. 多层前向网络的逼近机理与拓扑结构学习方法. 通讯学报，1998，19(3)：29～34

12 董聪，郭晓华. 广义遗传算法的逻辑结构及全局收敛性的证明. 计算机科学，1998，25(5)：37～42

13 董聪，郭晓华. 广义遗传算法的数学结构. 中国科学基金，1999，(2)：77～80

14 董聪. 大脑、知觉模型和计算机模拟. 科技导报，1997，(7)：7～10

15 Kirkpatrick S. Optimization by Simulated Annealing: Quantitative Studies. J. Statis. Phys. , 1984, 34: 975～986

16 Jacob F. The Possible and The Actual. University of Washington Press, Seattle, 1982

17 董聪，郭晓华，袁曾任. 基于广义遗传算法的全局优化方法. 计算机科学，1999，26(6)：7～10

18 李虎军. 桥梁诊断与遗传算法. 科学时报，1999-4-26

19 董聪，郭晓华. 基于知识的结构故障诊断理论. 计算机科学，1999，26(12)

20 张春霆. 生物信息学——重大科学意义与经济效益兼备的新学科. 中国科学基金，1999，(2)：65～68

21 董聪，刘西拉. 广义 BP 算法及网络容错性和泛化能力的研究. 控制与决策，1998，13(2)：120～124