一个移动进程演算的互模拟同余定义框架*)

陈韬略 李 斌 胡 昊 吕 建
(南京大学计算机软件新技术国家重点实验室 南京 210093)

摘 要 并发计算模型是理论计算机科学研究的重要领域之一。以π演算为代表的移动进程演算是目前并发理论的研究热点。互模拟等价定义是移动进程演算研究中的核心概念和问题,而传名机制使得移动进程演算中的互模拟同余关系更加复杂和有趣。本文在分析了常见的互模拟同余定义的基础上,通过抽取定义的核心要素,提出了一个三维的互模拟同余定义模型,从而将一般文献中常见的互模拟定义纳入到一个统一的框架中来,加深了我们对移动进程演算中互模拟概念的理解;同时本文利用这个模型,系统分析了各种互模拟之间的关系。模型的优点在于它的普适性和开放性。

关键词 并发理论,移动进程演算,互模拟,通用框架

A Bisimulation Congruence Definition Framework for Mobile Process Calculi

CHEN Tao-Lue LI Bin HU Hao LU Jian
(State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing, 210093)

Abstract Concurrent computation model is one of the most important fields in theoretical computer science. Mobile process calculus, notably π calculus, is one of the hottest fields in concurrent theory. Bisimulation equivalence is the key concept and problem in the research of this kind of calculus. However, the name-passing mechanism makes the definition of bisimulation congruence more intricacy and interesting. Based on our understanding of common bisimulation congruence definitions, a very generic framework is proposed for analysis and systematic study of such definitions. Most of common bisimulation congruence can be embedded in this uniform model, which provides some insight into bisimulation congruence in mobile process calculus; based on this model, the relations between these congruence relations are systematically studied and can be specified in a uniform and clear way. The main merits of this framework lie in its generality and openness.

Keywords Concurrent theory, Mobile process calculus, Bisimulation, General framework

1. 引言

计算模型一直是计算机科学研究的重要问题之一。计算机科学家对于顺序计算的研究已经处于比较成熟的阶段,主要体现为提出并深入研究了各种各样的顺序计算模型,并严格证明了它们的等价性。这其中,在程序设计语言及相关领域的研究中尤以λ演算[1]为代表。与顺序计算相比,对并发计算的认识还处于相当肤浅的阶段。事实上,顺序计算可以看成是并发计算的特例。相比之下,并发计算是一个相当广阔的领域,其复杂性要大得多。然而,随着计算机技术和网络技术的发展,并发已成为新一代计算范型的本质特征,对并发本质的把握有着重要的理论和现实意义。目前,对并发理论的研究主要集中在并发系统的形式模型和形式语义上[7]。研究者从不同角度出发相继提出了各种各样反映并发本质的模型,如Petri 网[16]、进程代数[6,9]、事件结构[21]、模态逻辑[2]、线性逻辑[5]等。这其中,以CCS[9]为代表的进程演算因其概念简洁,可用的数学工具丰富,在并发系统的规约、分析、设计和验证等方面得到了广泛的应用,已经成为并发计算的经典模型之一;同时也是并发理论特别是进程代数研究的奠基性工作。

进入 20 世纪 90 年代以来,随着分布计算技术的发展,在理论和实践中需要我们对移动这一概念加以描述。由R. Milner, J. Parrow 和 D. Walker 等人提出的π演算[11]正是为了适应这一需求而引入的。它是在CCS的基础上通过允许进程之间传递通道名而得到的。在π演算中,通信将导致参与通信的进程之间的拓扑结构发生变化,从而导致通道连接的动态变化,因而可以用来描述通讯拓扑结构可动态改变的并发系统(如移动电话系统)。事实上,π演算是一大类进程演算,即所谓传名进程演算的代表。传名进程演算又称移动进程演算,在这类演算中,移动通过从一个进程向另一个进程传送通道名获得。通过传名使得收到通道名的进程能够利用它与第三方进程进行通讯,从而赋予系统动态创建通道的能力,使"移动"能够以一种隐式的方法表达出来。正是这种传名机制使得以π演算为代表的移动进程演算具有强大的表达能力,典型的例子是能够翻译λ演算[10]和高阶进程演算[17];然而同时这也导致了移动进程演算语义研究的复杂性。

在移动进程演算的研究中,进程的等价性问题一直是个核心问题:即在何意义下,两个语法形式不同的进程表达式可以认为是行为等价的。在进程演算中常用的等价关系有互模拟等价、测试等价、踪迹等价等,这其中以互模拟等价最有影响。互模拟的概念曾被广泛用于数理逻辑和博奕论中,它最初

*)基金项目:国家重点基础研究发展规划 973 项目(No. 2002CB312002); 国家自然科学基金(No. 60273034); 国家 863 高科技项目(No. 2001AA113110, No. 2002AA116010); 江苏省教育厅自然科学基金(No. 01KJB520010)。

是由 Park[13]和 Milner[9]引入到并发理论的研究中,并产生了巨大的影响。一般而言,互模拟等价较为精细地刻画了行为等价这一直觉概念,且具有相对较好的代数性质,无论是对理论研究还是实际应用都比较适合,因而在进程演算的研究中占有显著的地位。在传统的非传名进程演算(如 pure CCS)中互模拟的定义是十分简洁和清晰的,然而在移动进程演算中,传名机制使得互模拟的定义显得十分复杂和有趣。如果简单地将传统的定义移植到传名演算中来,则所得到的定义将和直觉所理解的进程等价相距甚远,因而是不合适的。一般而言,我们希望所定义的互模拟等价关系不仅仅是一个等价关系,而且要是一个同余关系。所谓同余关系,简而言之就是对于演算所有的操作子封闭。这是因为,在直觉上,进程行为实际体现为与环境的交互事件流。两个进程被认为是行为等价的,当且仅当它们与任何环境交互都不能体现出差异,即仍然是行为等价的。形式上,这种思想体现为将两个互模拟等价的进程表达式置于相同的上下文(context)中仍然得到两个互模拟等价的进程表达式。基于此,我们希望所定义的互模拟等价关系对于演算所有的操作子封闭。

在移动进程演算的研究中,根据不同的需要,研究者提出了各种各样的互模拟同余关系。如迟互同余(late congruence[11])、早互同余(early congruence[11])、开互同余(open congruence[18])和接口互同余(barbed congruence[12])等等。本文的目的,在于从一个新的角度来研究这些同余关系,并比较它们之间的关系。具体而言,本文提出了一个三维的互模拟同余定义框架,并在此基础上系统分析了带有等名测试和不等名测试的 π 演算的各种常见互模拟同余关系的定义,指出了开互同余的定义实际上可以看成是一种宽泛的定义风格的特殊化,从而加深了我们对互模拟定义的理解,为进一步的深入研究打下了坚实的基础。值得指出的是,本文的工作主要针对 π 演算进行,这主要是利用它的典型性以更好地说明问题。事实上,本文的思想和方法可以适用于其它以传名机制为核心的移动进程演算,如 χ 演算[3],各种异步进程演算[26,27]中。

本文第 2 部分简要引入 π 演算,包括语法和语义,并给出常见的四种同余关系(迟互同余、早互同余、开互同余和接口互同余)的定义,包括强、弱两种版本;第 3 部分引入一个三维的互模拟同余定义框架,并分析了从框架中导出的各种互模拟同余关系;第 4 部分利用我们提出的框架分析和讨论各个互模拟同余之间的关系;最后总结全文,并讨论进一步工作。

2. π 演算和互模拟定义

2.1 π 演算的语法和语义

在本部分,我们将给出 π 演算的语法和语义,并讨论相关的性质。

名字是 π 演算中基本的概念,主要用以表示通道名。记 N 为可数无限的名字集合。一般地,N={a,b,c…};用 $\overline{N}$ 表示对偶名的集合 $\{\overline{x}\mid x\in N\}$。

π 演算的语法由下列 BNF 给出:

$$P::=0\mid\pi.P\mid P|P\mid P+P\mid(x)P\mid[x=y]P\mid[x\neq y]P\mid A(\vec{y})$$

$$\pi::=a(x)\mid\overline{a}x\mid\tau$$

其中,$a,x,y\in N$。

其具体含义可以参见文[11],在此不赘述。我们记 Π 为所有进程表达式组成的集合,它随着上下文含义有所不同。值得指出的是,我们引入了不等名测试(mismatch)$[x\neq y]P$ 的构造。一般地,用 $\pi^{=}$ 表示只含有等名测试的 π 演算,而 $\pi^{\neq}$ 表示整个演算。记 Π 是进程表达式的集合,根据上下文它可以代表不同的进程集合。

遵循进程代数研究的传统,在 $a(x).P$ 和 $(x)P$ 中,x 是受限名,这就引入了我们通常所谓的受限名和自由名的区别。在本文中,我们分别以 bn(P) 和 fn(P) 表示进程 P 的受限名和自由名。我们将使用通常的 α-换名协议,即将进程中的受限名换成一个新名,而不改变进程的语法。一般地,记 $\vec{x}$ 是名字 $x_1,\cdots,x_n$ 的序列,$|\vec{x}|$ 是 $\vec{x}$ 的长度,相应地,$(\vec{x})p$ 表示 $(x_1)\cdots(x_n)P$。

替换 σ 是 $N\to N$ 的函数,一般地,我们记为 $\{y_1/x_1,\cdots,y_n/x_n\}$ 简记 $\{\vec{y}/\vec{x}\}$;$P\sigma$ 表示将 P 中的所有自由名按 σ 替换后得到的进程。其详细定义可见文[11],在此不赘述。

上下文(context)是指一个不完全的进程。部分上下文以 BNF 定义如下:

$$C[]::=[]\mid\pi.C[]\mid P|C[]\mid(x)C[]\mid[x=y]C[]$$

全上下文(full context)定义为:

$$C[]::=[]\mid\pi.C[]\mid P|C[]\mid(x)C[]\mid[x=y]C[]\mid P+C[]\mid[x\neq y]C[]$$

π 演算的迁移语义由如下的结构化操作语义所定义。其中,α 取值于 $\{a(x),\overline{a}x\mid a,x\in N\}\cup\{\tau\}$:

$$pre\ \frac{-}{\pi.P\xrightarrow{\pi}P}$$

$$par\ \frac{P\xrightarrow{\alpha}P'}{P|Q\xrightarrow{\alpha}P'|Q}\,bn(\alpha)\cap fn(Q)\neq\phi$$

$$res\ \frac{P\xrightarrow{\alpha}P'}{(x)P\xrightarrow{\alpha}(x)P'}\,x\notin n(\alpha)$$

$$com\ \frac{P\xrightarrow{a(x)}P',\ Q\xrightarrow{\overline{a}y}Q'}{P|Q\xrightarrow{\tau}P'\{y/x\}|Q'}$$

$$sum\ \frac{P\xrightarrow{\alpha}P'}{P+Q\xrightarrow{\alpha}P'+Q}\qquad open\ \frac{P\xrightarrow{\overline{a}x}P'}{(x)P\xrightarrow{\overline{a}(x)}P'}\,x\neq a$$

$$close\ \frac{P\xrightarrow{a(x)}P',\ Q\xrightarrow{\overline{a}(x)}Q'}{P|Q\xrightarrow{\tau}(x)(P'|Q')}\qquad match\ \frac{P\xrightarrow{\alpha}P'}{[x=x]P\xrightarrow{\alpha}P'}$$

$$mismatch\ \frac{P\xrightarrow{\alpha}P'}{[x\neq y]P\xrightarrow{\alpha}P'}$$

$$id\ \frac{P\{\vec{y}/\vec{x}\}\xrightarrow{\alpha}P'}{A(\vec{y})\xrightarrow{\alpha}P'}\,A(\vec{x})\Leftarrow P$$

2.2 互模拟定义

在纯 CCS 中,我们能够以一种清晰和简明的方法给出互模拟的定义:

定义 1[9] 对称二元关系 $R\subseteq\Pi\times\Pi$ 称为强互模拟,如果对于任意 $(P,Q)\in R$,蕴涵:若 $P\xrightarrow{\alpha}P'$,则存在 Q',使得 $Q\xrightarrow{\alpha}Q'$,且 $(P',Q')\in R$。

P 和 Q 称为强互模拟,记为 $P\sim Q$,若存在一个强互模拟 R,使得 $(P,Q)\in R$。

如果我们将此定义简单移植到 π 演算中,则可以得到如下的定义。在本文中,我们称之为基互模拟。

定义 2(基互模拟) 对称二元关系 $R\subseteq\Pi\times\Pi$ 称为强基互模拟,如果对于任意 $(P,Q)\in R$,蕴涵:若 $P\xrightarrow{\pi}P'$,则存在

Q'，使得 $Q \xrightarrow{\pi} Q'$，且 $(P',Q') \in R$。

P 和 Q 称为强互模拟，记为 $P \sim_g Q$，若存在一个强基互模拟 R，使得 $(P,Q) \in R$。遗憾的是，这样得到的互模拟关系不是一个同余关系，它甚至对于并行算子 | 都不封闭，这是我们所不能接受的。为此研究者相继提出了修正定义，陈述如下：

定义 3(迟强互模拟[11]) 对称二元关系 $R \subseteq \Pi \times \Pi$ 称为强迟互模拟，如果对于任意 $(P,Q) \in R$，蕴涵：

(i)若 $P \xrightarrow{a(x)} P'$，则存在 Q'，使得 $Q \xrightarrow{a(x)} Q'$，且对任意 $z \in N$，$(P'\{z/x\}, Q'\{z/x\}) \in R$；

(ii)若 $P \xrightarrow{\pi} P'$，且 π 非输入算子，则存在 Q'，使得 $Q \xrightarrow{\pi} Q'$，且 $(P',Q') \in R$

P 和 Q 称为强迟互模拟，记为 $P \dot{\sim}_l Q$，若存在一个强迟互模拟 R，使得 $(P,Q) \in R$。

P 和 Q 称为强迟互同余，记为 $P \sim_l Q$，若对任意替换 σ，$P\sigma \dot{\sim}_l Q\sigma$。

定义 4(强早互模拟[11]) 对称二元关系 $R \subseteq \Pi \times \Pi$ 称为强早互模拟，如果对于任意 $(P,Q) \in R$，蕴涵：

(i)若 $P \xrightarrow{a(x)} P'$，则对任意 $z \in N$，存在 Q'，使得 $Q \xrightarrow{a(x)} Q'$，且 $(P'\{z/x\}, Q'\{z/x\}) \in R$；

(ii)若 $P \xrightarrow{\pi} P'$，且 π 非输入算子，则存在 Q'，使得 $Q \xrightarrow{\pi} Q'$，且 $(P',Q') \in R$

P 和 Q 称为强早互模拟，记为 $P \dot{\sim}_e Q$，若存在一个强早互模拟 R，使得 $(P,Q) \in R$。

P 和 Q 称为强早互同余，记为 $P \sim_e Q$，若对任意替换 σ，$P\sigma \dot{\sim}_e Q\sigma$。

为了给出开互模拟的定义，需要区别(distinction)等一系列相关概念，具体请见文[18]，在此不赘述。

定义 5(开互模拟[18]) 区别加标项上的对称二元关系族 $R=\{R^D\}$ 称为开互模拟，若对于任意 R^D 和任意 $\sigma \rhd D$，$(P,Q) \in R^D$ 蕴涵：

(i)$P\sigma \xrightarrow{\bar{a}(x)} P'$，且 $x \notin fn(P\sigma, Q\sigma, D\sigma)$，则存在 Q'，使得 $Q\sigma \xrightarrow{\bar{a}(x)} Q'$ 且 $(P',Q') \in R^{D\sigma \cup \{x\} \times fn(P\sigma,Q\sigma)}$；

(ii)$P\sigma \xrightarrow{\pi} P'$，且 π 非受限输出算子，则存在 Q'，使得 $Q\sigma \xrightarrow{\pi} Q'$，且 $(P',Q') \in R^{D\sigma}$。

P 和 Q 称为强开互同余，记为 $P \sim_o Q$，若存在一个强开互模拟族 R^D，使得 $(P,Q) \in R^\phi$。

需要指出的是，引入区别加标的目的在于合理地处理限制算子(restriction)带来的问题，如果只考虑不含限制的子语言，则定义 4 中的(i)(ii)可以统一的加以处理，整个定义将是十分简洁的，具体可见文[14,18]。

为了给出接口互模拟的定义，首先给出接口观察的定义。

定义 6 对任意进程 P，名字 a，称 a 是进程 P 的直接接口，记为 $P \downarrow_a$，当且仅当存在 P'，使得 $P \xrightarrow{a(x)} P'$ 或者 $P \xrightarrow{\bar{a}x} P'$。

定义 7 对称二元关系 $R \subseteq \Pi \times \Pi$ 称为强接口互模拟，如果对于任意 $(P,Q) \in R$，蕴涵：

(i)对任意名字 a，$P \downarrow_a$ 蕴涵 $Q \downarrow_a$；

(ii)$P \xrightarrow{\tau} P'$，则存在 Q'，使得 $Q \xrightarrow{\tau} Q'$，且 $(P',Q') \in R$

P 和 Q 称为强接口互模拟，记为 $P \dot{\sim}_b Q$，若存在一个强接口互模拟 R，使得 $(P,Q) \in R$。

P 和 Q 称为强接口互同余，记为 $P \sim_b Q$，若对任意上下文 $C[\,]$，$C[P] \dot{\sim}_b C[Q]$。

以上给出了常见的四种进程间同余关系 $\sim_l, \sim_e, \sim_o, \sim_b$ 的定义，它们的形式是所谓的强版本的。事实上，在进程演算的研究中，更有实际意义的是所谓弱版本的等价关系，二者的区别简而言之是是否将进程内部动作 τ 和进程间的通信动作(输入，输出动作)同等看待。为了定义弱互模拟，首先给出一些符号的定义。用 $\Rightarrow$ 表示 $(\xrightarrow{\tau})^*$；用 $\stackrel{\alpha}{\Rightarrow}$ 表示 $\Rightarrow \xrightarrow{\alpha} \Rightarrow$；当 $\alpha \neq \tau$ 时，$\stackrel{\hat{\alpha}}{\Rightarrow}$ 表示 $\stackrel{\alpha}{\Rightarrow}$，否则表示 $\Rightarrow$。

同样地，在 CCS 中，弱互模拟的定义是十分简洁的。

定义 8 对称二元关系 $R \subseteq \Pi \times \Pi$ 称为弱互模拟，如果对于任意 $(P,Q) \in R$，蕴涵：若 $P \xrightarrow{\alpha} P'$，则存在 Q'，使得 $Q \stackrel{\hat{\alpha}}{\Rightarrow} Q'$，且 $(P',Q') \in R$。

P 和 Q 称为弱互模拟，记为 $P \approx Q$，若存在一个弱互模拟 R，使得 $(P,Q) \in R$。

遗憾的是，上述定义的弱互模拟还不是一个同余关系，它对选择算子不封闭。一个经典的解决方法是引入观察同余的概念[9]。在本文中，为了术语的统一，称之为弱互同余。

定义 9 P 和 Q 称为弱互同余，记为 $P \simeq Q$，若

(i)$P \xrightarrow{\tau} P'$，则存在 Q'，使得 $Q \stackrel{\tau}{\Rightarrow} Q'$，且 $P' \approx Q'$；

(ii)$Q \xrightarrow{\tau} Q'$，则存在 P'，使得 $P \stackrel{\tau}{\Rightarrow} P'$，且 $P' \approx Q'$。

对于 π 演算，弱互同余的定义将更为复杂。一般的方法是首先定义一个对于部分上下文封闭的弱型同余关系，继而采用定义 8 的方法，以得到对于全上下文封闭的同余关系。作为一个例子，我们给出弱迟互同余的定义。对于其它的定义，囿于篇幅，在此不详细列出，请参考文[14]。

定义 10 对称二元关系 $R \subseteq \Pi \times \Pi$ 称为弱迟互模拟，如果对于任意 $(P,Q) \in R$，蕴涵：

(i)若 $P \xrightarrow{a(x)} P'$，则存在 Q''，使得 $Q \Rightarrow \xrightarrow{a(x)} Q''$，且对任意 $z \in N$，存在 Q'，满足 $Q''\{z/x\} \Rightarrow Q'$ 且 $(P'\{z/x\}, Q') \in R$；

(ii)若 $P \xrightarrow{\pi} P'$，且 π 非输入算子，则存在 Q'，使得 $Q \stackrel{\hat{\pi}}{\Rightarrow} Q'$，且 $(P',Q') \in R$

P 和 Q 称为弱迟互模拟，记为 $P \dot{\approx}_l Q$，若存在一个弱迟互模拟 R，使得 $(P,Q) \in R$。

P 和 Q 称为弱迟互等价，记为 $P \approx_l Q$，若对任意替换 σ，$P\sigma \dot{\approx}_l Q\sigma$。

P 和 Q 称为弱迟互同余，记为 $P \simeq_l Q$，若

(i)$P \xrightarrow{\tau} P'$，则存在 Q'，使得 $Q \stackrel{\tau}{\Rightarrow} Q'$，且 $P' \approx_l Q'$

(ii)$Q \xrightarrow{\tau} Q'$，则存在 P'，使得 $P \stackrel{\tau}{\Rightarrow} P'$，且 $P' \approx_l Q'$

3. 三维定义空间

在本节中，我们通过对移动进程演算中互模拟同余关系定义的分析，提出了一个三维的定义模型，以期对互模拟同余关系的定义有一个统一而深刻的理解，并有助于澄清各种互模拟之间的关系。使用该模型，可以从 3 个角度选取互模拟同余的定义参数，从而得到相应的定义。我们的主要贡献在于，将开互同余的定义方法一般化为一种定义风范，从而能够很好地解释 π^* 中弱开互同余研究中所遇到的问题。

3.1 定义参数

我们从三个角度选取定义参数，分别为进程自身规约的要求、对进程内部动作的不同看待、上下文对于进程规约的影响。下面详细阐述之。

(1) 进程自身规约要求。参考第 2 节中各种定义，主要考虑四种情况，即迟、早、基和接口的情况。其中，迟、早两种情况是受传值 CCS(value-passing CCS)启发引入的，可以参考文[11]；基情况是将纯 CCS 的定义简单移植所得到的，对其本身的研究并无实际意义；接口情况最初是由 Milner 和 Sangiorgi 引入的[12]，它对于研究各种进程演算之间的关系具有重要的意义。

• 迟情况(late)。进程所执行的输入动作 a(x)是一般的、抽象的输入动作。定义迁移时不对输入前缀进行参数实例化，而将之推迟到真正需要的时候，即只在定义互模拟关系或需要进行数据传送时才进行输入参变量的实例化。

• 早情况(early)。进程所执行的输入动作经过实例化后的基本动作，即在定义迁移时就对输入前缀进行参数实例化。

• 基情况(ground)。不考虑参数的实例化，而仅仅考察抽象动作之间的匹配，这在本质上忽略了传名的影响，而将它简单地看成是纯演算。

• 接口情况(barbed)。基本思想在于两进程能够模拟相互的通信，而忽略其它更为细节、具体的动作(如输入输出动作)之间的模拟，同时进程保持在同一通道上通信的能力。

(2) 对进程内部动作的不同观点。

• 强观点(strong)。将进程内部动作(τ 动作)和其它各种动作(如输入动作、输出动作、受限输出动作)同等看待，在模拟过程中需要逐一匹配。

• 弱观点(weak)。对一个外部观察者而言，他不关心表示进程内部通信事件的 T 动作，而只关注外部可见的行为动作。换言之，在互模拟的匹配和比较中忽略表示内部通讯的 τ 动作。

(3)上下文对于进程规约的影响。一般而言，在移动进程演算中，单纯通过(1)中的四种方法得到的互模拟关系(为叙述方便，记为 R)都不是同余关系，这就需要利用上下文 C[]，选取一个满足同余条件的子集。在直觉上，将进程置于上下文中可以看成是进程与环境的交互。据此，根据在进程规约的不同阶段使用上下文，可以有两种不同的定义同余关系的方法。

• 闭风格(close)。只要求进程在模拟规约的第一步与环境发生交互(对上下文封闭)。形式地，对于进程 P,Q，定义 PR_cQ 当且仅当对任意上下文 C[]，$C[P]RC[Q]$，换言之，即在互模拟关系 R 的定义之外给出同余关系 R_c 的定义。

• 开风格(open)。开风格的提出是受 Sangiorgi 的开互模拟[18]的启发，与闭风格相比，它要求进程在模拟规约的每一步都要与环境发生交互(对上下文封闭)。形式地，对于进程 P,Q，定义 PR_cQ 当且仅当 $R_c \subseteq R$，且对任意上下文 C[]，满足 PR_cQ 当且仅当 $C[P]R_cC[Q]$，这实际上是在互模拟关系 R 的内部给出了同余关系 R_c 的定义。

上述(1)(2)(3)三个方面构成了我们的互模拟同余关系的三维定义空间，如图 1 所示。其中每一方面表示为空间中的一维，而每一维上的不同参数则表示了该方向上的各种选择。空间中的三维是相互正交的，亦即上述三个方面是相互独立的三个因素，可以分别对其进行单独探讨，且每个方向的一种参数从逻辑上都可以和其他方向的各种参数进行组合。

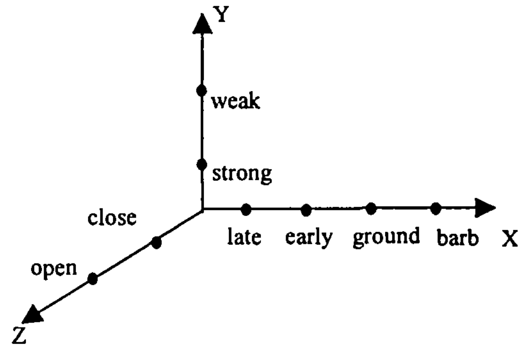

图 1 互模拟同余的三维定义空间

3.2 参数组合及相关定义

上述的三维模型给出了移动进程演算中同余关系定义的系统方法。在本节中，介绍不同的参数组合对应的各个定义。由于选择三维设计空间上的参数即可给出所需的定义，本文以 XX-YY-ZZ 的方式来表示定义，其中 XX 代表 X 方向上的 late(l)、early(e)、ground(g)或 barb(b)；YY 代表 Y 方向上的 weak(w)或 strong(s)；ZZ 代表 Z 方向上的 open(o)或 close (c)。

这样，我们实际上就可以得到 16 种定义。值得强调的是，在这里所陈述的只是定义的基本思想和框架。由于 π 演算的复杂性，在具体的定义时还必须进行某些修正，以使得定义在直觉上与人们对行为等价的理解相吻合，同时在数学上也是良定义的。为简便计，本文用“ * ”,“ # ”表示该方向上的任意参数选择。需要说明的是，这些同余关系大部分已经在相关文献中出现，本文的目的并不在于提出新的同余关系，而在于对它们有一个系统的理解和研究。基于此，我们无需列出它们的具体定义，而是在下表中列出一些主要的研究文献。

表 1 π^-,strong

	late	early	ground	barbed
Close	[11]	[11]		[12]
open			[18]	[19]

表 2 π^-,weak

	late	early	ground	barbed
Close	[11]	[11]		[12]
open			[18]	[19]

表 3 $\pi^{\neq}$,strong

	late	early	ground	barbed
Close	[15]	[15]		
open			[7]	

表 4 $\pi^{\neq}$,weak

	late	early	ground	barbed
Close	[8]	[8]		
open	[4]	[4]	[4]	

上述表中的空格，有些事实上没有研究的必要，有些是因为目前尚无文献系统加以正式研究，对此我们将在第 4 部分加以阐述。

4 互模拟同余之间的关系

在本节中，我们将从一个较为抽象的角度阐述 16 种互模

拟同余之间的关系。由于对于 π^- 和 $\pi^{\neq}$，所得的结论有较大差异，必要时将分别阐述之。如果不指明具体的演算，则结论对于 π^- 和 $\pi^{\neq}$ 皆成立。本节所陈述的定理，有一部分在一般文献中已有阐述，对此将略去其证明。

定理 1　$*$-s-# $\subset$ $*$-w-#，其中 $* \in \{l,e,g,b\}$，且 $\# \in \{c,o\}$

此定理说明，强版本的互模拟同余严格包含于相应的弱版本定义中。

定理 2　$*$-#-o $\subset$ $*$-#-c，其中 $* \in \{l,e,g,b\}$，且 $\# \in \{s,w\}$

此定理说明，用开风格定义的互模拟同余严格包含于相应的闭风格定义的互模拟同余。

定理 3　l-$*$-# $\subseteq$ e-$*$-# $\subseteq$ g-$*$-# $\subseteq$ b-$*$-#，其中 $* \in \{s,w\}$，且 $\# \in \{c,o\}$

定理 4　对于 π^-，e-s-c = g-s-c = b-s-c

e-s-c 和 b-s-c 的等价性是 π 演算中的重要而有趣的结果。其详细证明可见文[20]。对于相应的弱版本的结论，文[17]证明了对于 π 演算进程表达式的一个子集，即有限映像进程(finite-image process)，结论是成立的；文[19]进一步证明了，如果在 π 演算中允许无限和(infinite sum)，则结论也是成立的；然而，对于 π 演算本身，这个问题仍然是个开问题，然而，一般研究者相信它们的确是等价的。这也是一般文献不研究 g-s-c 或 g-w-c 的原因。对于 $\pi^{\neq}$，虽然没有文献给出详细的证明，但我们发现，对于 π^- 的相关证明可以无困难地移植到 $\pi^{\neq}$ 的情况中来，故相关结论依然成立。

定理 5　对于 π^-，l-$*$-o = e-$*$-o = g-$*$-o，其中，$* \in \{s,w\}$

定理 5 的证明可以由定义直接证得，它也告诉我们，为什么一般文献[18,25]中只对 π^- 中的开互模拟(包括强弱两个版本)加以研究，这两种同余关系实际上就是本文的 g-s-o 和 g-w-o。

开互模拟的定义原先只是针 π^- 对进行的。将其推广到 $\pi^{\neq}$ 中来是一个很有趣的问题。李舟军在其博士论文[7]中对此进行了研究，对于强版本，其定义可以用 π^- 类似的定义方法给出，也就是说，有如下结论：

定理 6　对于 $\pi^{\neq}$，l-s-o = e-s-o = g-s-o

然而，遗憾的是，文[7]对于弱版本的解决是有问题的。其原因在于，事实上，我们有：

定理 7　对于 $\pi^{\neq}$，l-w-o $\subset$ e-w-o $\subset$ g-w-o

Fu 和 Yang 在文[4]中对此进行了详细的研究。从上文的分析，可以很清楚地看到，本质上讲，对于 π^- 也应该对开风格的早迟同余进行研究，只不过它们与开基同余相同，故不需要详细进行。而 mismatch 操作子使得 l-w-o，e-w-o，g-w-o 可以严格区分开来，从而使得有必要对它们进行详细的讨论。从这个角度看，使得文[4]所引入的早开同余和迟开同余成为自然。

对于 b-$*$-o 的情况，一般地有如下结论：

定理 8　对于 π^-，以下性质成立：

(i) g-$*$-o $\subset$ l-$*$-c $\cap$ b-$*$-o

(ii) l-$*$-c $\not\subset$ b-$*$-o 且 b-$*$-o $\not\subset$ l-$*$-c

综合定理 2～定理 8，我们可以将互模拟同余之间的关系表示如下，其中→表示集合包含关系 $\subseteq$：

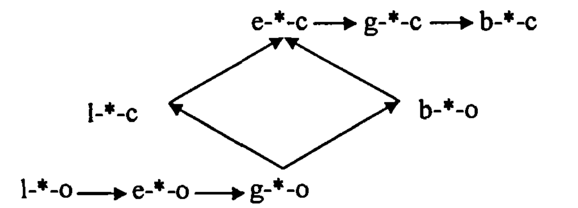

图 2　互模拟同余关系图

结论　互模拟等价是进程演算研究中的核心概念和问题。传名机制使得移动进程演算中的互模拟同余关系相对而言更加复杂和有趣。不同的研究者出于不同的目的提出了各种各样的互模拟定义。本文在分析了这些定义的基础上，通过抽取定义的核心要素，提出了一个三维的互模拟同余定义模型，从而将一般文献中常见的互模拟定义纳入到一个统一的框架中来，加深了我们对移动进程演算中互模拟概念的理解。本文的主要贡献在于：(1)将开同余的定义方式一般化，明确提出了“开”和“闭”两种定义风格，从而解释了弱开同余在带有 mismatch 操作子的 π 演算中的推广中所遇到的问题，使得 Fu 和 Yang 引入的早、迟开同余成为自然；(2)通过统一的定义，系统分析了各种互模拟之间的关系，使得相关文献中的有关结论和我们的一些结果能够以一种统一、清晰的方式加以表达，为进一步的研究打下了坚实的基础。

本文所引入的模型的优点在于普适性和可扩充性。本文的结论虽然主要针对 π 演算进行，但事实上其方法适用于其它的移动进程演算，如 χ 演算，异步 π 演算等，都可以遵循这样的方法对其互模拟定义加以研究；此外对于移动环境演算(MA[22])及其变例(如 SA[23]ROAM[24]等)，目前所提出的行为等价实际上主要为上下文等价(contextual equivalence[22])和接口同余[23]，用本文的术语，它们本质上是“开”和“闭”两种风格的定义，用这样的观点去看待其定义将使得我们的研究更为方便。本文所引入的是一个开放的框架，其参数定义，尤其是框架中的 x 轴具有较强的可扩充性。事实上，研究者可以根据自身的需要，定义新的规约(转移)要求，继而从另两个角度对定义进行扩充。

进一步的工作在于利用本文提出的思想和框架指导对各种移动进程演算互模拟的研究，包括公理化问题，增强我们对并发模型的理解，并将之用于实践中。

参考文献

1　Church A. The calculi of lambda-conversion. Annals of Mathematics Studies, no. 6, Princeton University Press, 1941

2　Emorson E A. Temporal and modal logic. In: Jan Leeuwen, ed. Handbook of Theoretical computer science, Volume B (Formal Models and semantics), The MIT Press, Cambridge, 1990. 789～840

3　Fu Y. The χ calculus. In: Proc. of the Intl. Conf. on Advances in Parallel and Distributed Computing, March 19th — 21th, ShangHai, IEEE computer Society Press, 1997. 74～81

4　Fu Y, Yang Z. Tau laws for pi calculus, appear in Theoretical Computer Science

5　Girard J Y. Linear logic. Theoretical computer science, 1987, 50: 1～102

6　Hoare C A R. Communicating Sequential Process. Prentice Hall, 1985

(下转第 27 页)

事。

3.4 其它算法

除了上面提到的算法，还有一些论文也讨论了随机流网络的可靠度问题，例如：Doulliez，Jamoulle[1]采用 Ford-Falkerson 流分析方法，将系统的状态空间分解为三种集合：accepted 状态，unaccepted 状态，unspecified 状态的不相交集合。每个 unspecified 状态用递归法继续分解，直至再也没有 unspecified 状态集了。accepted 状态的不相交集合的概率可以直接计算出来，它们的和就是系统的水平为 d 的可靠度。Fishman[2]用 Monte Carlo 抽样法从有限个样本中分离出点估计和区间估计。Clancy 等[3]则用逼近法估计系统的可靠度。先用 Doulliez，Jamoulle 的分解方法逐层分解系统的状态空间直到某一指定的水平，然后采用 Monte Carlo 拟合，用 unspecified 状态的不相交集合的分布来近似系统的可靠度。

结论 本文在随机流网络的可靠度问题的数学模型的基础上，着重介绍了在这个研究方向上的主要研究成果。可以看到，MPs，MCs 是解决这类问题的简单而有效的方法，但是也有其局限性，如上面提到的，这些方法的前提是系统的所有 MPs，MCs 已知。而对二态网络来说，求解 MPs，MCs 是 NP-难的。另外，本文提到的算法都是针对弧容量有限定的网络，现实世界的系统的节点也是有容量限定的，甚至会失效的。目前对这类问题的研究结果还非常少。文[7]提出的解决办法是基于概念"The failure of a node implies the failure of links incident from it"的，只适用于二态系统。因此可对这类问题进行深入的研究。

参 考 文 献

1 Doulliez P, Jamoulle J. Transportation Networks with Random Arc Capacities. RAIRO, Recherche Operationnelle, (Operations Research), 1972, 3: 45～60

2 Fishman G S. Monte Carlo Estimation of the Maximum Flow Distribution with Discrete Stochastic Arc Capacity Levels. Naval Research Logistrics Quarterly, 1989, 36: 829～849

3 Clancy D P, Gross G, Wu F F. Probabilistic Flows for Reliability Evaluation of Multiarea Power System Interconnections. Electrical Power and Energy Systems, 1983, 5: 100～114

4 Lee S H. Performance Indexes of a Telecommunication Network. IEEE Trans. Reliability, 1980, R-29: 24～26

5 Abraham J A. An Improved Algorithm for Network Reliability. IEEE Trans. Reliability, 1979, 28: 58～61

6 Aggarwal K K, Chopra Y C, Bajwa J S. Capacity Consideration in Reliability Analysis of Communication System. IEEE Trans. Reliability, 1982, 31: 177～180

7 Aggarwal K K, Gapta J S, Misbra K B. A Simple Method for Reliability Evaluation of a Communication System. IEEE Trans. Communications, 1975, COM-23: 563～566

8 Block H W, Savits T H. A Decomposition for Multistate Momotone Systems. J. Appl. Prob., 1982, 19: 391～402

9 Davio M, Deschamps J D, Thaysc A. Discrete and Swithing Functions. Georgi Publishing Company, McGraw-Hill International Book Company, 1978

10 Xue J. On Multistate System Analysis. IEEE Trans. Reliability, 1985, R-34: 329～337

11 Lin J S, Jane C C, Yuan J. On Reliability Evaluation of a Capacitated-Flow Network in Terms of Minimal Pathsets. Network, 1995, 25: 131～138

12 Lin Y K. A Simple Algorithm for Reliability Evaluation of a Stochastic-Flow Network with Node Failure. Computers & Operations Research, 2001, 28: 1277～1285

13 Jane C C, Lin J S, Yuan J. Reliability Evaluation of a Limited-Flow Network in Terms of Minimal Cutsets. IEEE Trans. Reliability, 1993, 42(3): 354～361

14 Lin Y K. On Reliability Evaluation of a Stochastic-Flow Network in Terms of Minimal Cuts. J. Chinese Institute of Industrial Engineers, 2001, 18(3): 49～54

(上接第15页)

7 Li Z. Theories and algorithm for verification of bisimulation equivalence in value-passing CCS and the π-calculus: [PhD Thesis]. Changsha Institute of Technology, Changsha, 1999

8 Lin H. Complete inference systems for weak bisimulation equivalences in the π-calculus. In: P. D. Mosses, M. Nielsen, M. I. Schwarzbach, eds. Proc. of TAPSOFT'95, volume 915 of LNCS, Springer, 1995. 187～201

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

10 Milner R. Functions as process. Journal of Mathem. Structures in Computer Science, 1992, 2(2): 119～141

11 Milner R, Parrow J, Walker D. A Calculus of Mobile Process, part Ⅰ/Ⅱ. Journal of Information and Computation, 1992, 100: 1～77

12 Milner R, Sangiorgi D. Barbed bisimulation. In: W. Kuich, ed. Proc. of ICALP'92, volume 623 of LNCS, Springer, 1992. 685～695

13 Park D. Concurrency and automata on infinite sequence. volume 154 of LNCS, Springer-Verlag, 1987. 561～572

14 Parrow J. An introduction to the π-calculus. Chapter to appear in Handbook of Process Algebra, Elesvier

15 Parrow J, Sangiorgi D. Algebraic theories for name-passing calculi. Journal of Information and Computation, 1995, 120(2): 174～197

16 Reisig W. Petri nets: an introduction. Springer-Verlag, 1985

17 Sangiorgi D. Expressing Mobility in Process Algebra: First-Order and Higher-Order Paradigms: [PhD Thesis]. LFCS, University of Edinburg, 1993

18 Sangiorgi D. A theory of bisimulation for the π-calculus. Acta Informatica, 1996, 33: 69～97

19 Sangiorgi D, Walker D. On barbed equivalence in π-calculus. CONCUR'01, 2001

20 Sangiorgi D, Walker D. The pi -calculus: a theory of mobile processes. Cambridge University Press, 2001

21 Winskel G. Event Structure. Volume 255 of LNCS. Springer-Verlag, 1987. 325～392

22 Cardelli L, Gordon A. Mobile ambients. Theoretical Computer Science, 2000, 240: 177～213

23 Levi F, Sangiorgi D. Controlling interference in ambients. In: Proc. POPL'00, Boston, Massachusetts, 2000. 352～364

24 Guan X, Yang Y, You J. Making ambients more robust. In: Proc. Intl Conf. on Software: Theory and Practice, Beijing, China, 2000. 377～384

25 Fu Y. An open problem of mobile processes. Journal of Computer (in Chinese), 2001, 24(7)

26 Honda K, Toloro M. On asynchronous communication semantics. object-based concurrent computing, volume 612 of LNCS, Springer-Verlag, 1991. 21～51

27 Merro M, Sangiorgi D. on asynchrony in name-passing calculi, ICALP'98, volume 1443 of LNCS, Springer-Verlag, 1998. 856～867