并发模型的分类与层次*)

The Classification and Hierarchy of Models for Concurrency

李文军 周晓聪 李师贤
(中山大学计算机科学系 广州510275)

Abstract In this paper we classify models for concurrency into practical models, concrete models and abstract models. We put emphasis on the abstract models for concurrency, which are foundations of concurrency and distributed theories. We give some orthogonal classifications of abstract models for concurrency, including both interleaving models (e. g. Hoare trace, labelled transition systems, synchronization trees) and true concurrency models (e. g. Mazurkiewicz trace, asynchronuou transition systems and event structures). At last we build a hierarchy structure of these models and introduce the categorical approach to the study of models for concurrency.

Keywords Models for concurrency, Transition systems, Trace languages, Event structures, Category theory

随着处理器性能的不断提高和网络技术的迅速发展，越来越多的计算机应用系统涉及并行与分布式计算，开发这类系统远比传统的串行系统困难，这除了并行与分布式计算本身固有的复杂性外，还有一个原因是人们对并发性的实质缺乏真正了解，因而对并发模型进行深入广泛的研究具有重要的现实意义。并发模型是一种描述与解释并发与分布式系统的形式化数学模型，在理论上可提供对并发系统及其行为的深刻理解，在实践中可为并发系统的设计与分析提供指导方法。探讨并发性的语义往往也集中在并发模型的研究，以此作为并发语言与系统的指称结构[1]。

1 概述

并发模型的主要目标是将并发行为的各种概念形式化，从而给出并发行为的精确定义，避免软件开发人员理解上的二义性。传统串行系统开发采用了类似方法，通过形式化定义语法和语义使程序设计语言的设计者、实现者与使用者三方达成共识。在并发程序中问题更加突出，例如 a 和 b 的并行执行是否等价于 a 和 b 以任意次序交错执行，所以并发系统开发人员更需要一种精确的数学模型。

正如向量空间、矩阵空间、拓扑学等几何数学模型作为计算机图形学的基础一样，并发模型的另一目标是为并发系统开发人员提供可靠的数学工具，用于说明、设计和验证并发系统，但许多传统工具都是针对串行系统设计的，对于并行与分布式系统远远不够。

1.1 并发模型的分类

并发模型最基本的概念有状态及其变迁、动作与事件、并行复合、不确定选择等。不同设计者的出发点不一样，对这些概念的理解与处理有差别，因而形成不同的并发模型。

1) 抽象并发模型、具体并发模型与实用并发模型

抽象并发模型主要用于并发性理论研究，所以模型中的概念尽量简单而且直接对应着容易理解与处理的数学对象，从而揭示并发性的本质。例如带标号变迁系统[1]、同步树[7]、Hoare 踪迹语言[4]、Mazurkiewicz 踪迹语言[6]、事件结构[12]、异步变迁系统[13]等都属于这类模型，抽象并发模型的另一作用是解释具体并发模型的语义。

具体并发模型强调实际应用，所以通常较抽象并发模型有更多、更琐碎的概念。例如 Petri 网[2]、CSP[4]、CCS[7]、π-演算[8]、CBS[9]等均属于这类模型。基于这类模型可构造实用的自动化软件工具，如爱丁堡大学基于 CCS 的"并发工作台"、牛津大学的"并发工厂"、"北卡罗莱纳州大学的"并发工作台"、各种 Petri 网建模与分析工具等。

基于具体并发模型派生出许多实用并发模型，这些模型有时又称并行计算模型，如 Occam、LINDA、

*)南京大学计算机软件新技术国家重点实验室基金资助课题。

MPI、BSP、PVM、PRAM、LogP、UNITY 等，这些模型一般建立在某个具体并发模型的概念框架中，但以更便于实际应用的程序设计语言、函数库或类库的形式出现。

实用并发模型更多地采用计算机领域的概念考虑与表达问题，兼顾计算机体系结构或软件体系结构（如模块化、重用等），而具体并发模型与抽象并发模型则独立于计算机的软、硬件体系结构，更关心并发性与分布性本身的性质。

这种分类对于学习与研究并发模型很有帮助，但某些模型的归类并不是绝对的。例如 Petri 网的许多变形（特别是许多高级网）已应用到广泛的领域，属于具体模型；而 C/E 网、出现网等基本网系统常用于理论研究，属于抽象模型。

2）*系统模型与行为模型*　状态是系统某一时刻的描述，变迁指系统根据某种方式从一个状态转换到另一状态。状态与变迁几乎是任何系统的最基本概念，并发与分布式系统也不例外。绝大多数并发模型中状态的变化都看作是瞬时发生的、不可分割的原子单位，其名称可能是变迁、动作、事件、标号等。

系统模型将系统状态显式地表达出来，这些状态有可能重复出现，例如 Petri 网、变迁系统属于系统模型；而行为模型则忽略这些信息，将注意力集中在系统的行为，这些行为用时间轴上的动作出现模式来描述，例如同步树、踪迹语言、事件结构属于行为模型。

一些系统模型将状态作为原子概念，不再刻画状态的内部结构，如变迁系统；而另一些系统模型则用更原始的概念定义状态，如 Petri 网。

3）*基于动作的模型与基于事件的模型*　如果并发模型中所有变迁都允许重复发生，则称这种状态变迁方式为动作；如果在整个进程执行过程中任何状态变迁最多只能发生一次，则称这种状态变迁方式为事件。Petri 网、CCS、Hoare 踪迹等是基于动作的模型；而事件结构、偏序多重集等则是基于事件的模型。基于事件的模型通常使用标注函数表示一个动作的多次发生。

动作与事件的区别源于对进程内部结构的探讨与进程外部行为的观察：基于动作的模型通常刻画了进程的内部结构，而基于事件的模型则记录了进程的执行过程。所以基于动作的模型更加简练，一个结构很小的基于动作的模型可能需要一个无穷结构的基于事件的模型才能表达。但基于事件的模型记录了一个进程的全部历史，模型中不存在循环，所以具有较好的可复合性，很容易将这些模型形成范畴，从而自然用到代数理论作为工具。

4）*交错模型与真并发模型*　交错模型中两个原子动作的并行执行定义为这两个动作的不确定交错，而真并发模型则将并发作为基本的概念，它并不将事件投影到线性时间轴上，$a|b$ 只表示不约束两个动作之间的任何次序关系，而不象 $a.b+b.a$ 表示两个动作的互斥。

建立交错模型的主要依据是从观察角度考虑，进程 $a|b$ 和进程 $a.b+b.a$ 是相同的，带来的优点是交错模型更容易用代数方法处理。虽然这种抽象适合于很多应用情况，但在对模型进行某些分析（例如检查安全性、公平性等系统性质）时往往带来状态空间爆炸问题。此外在哲学上考虑，交错模型假设存在一个记录所有动作发生时间的全局时钟，但在分布式系统中这是不实际的，因为网络与相对论等原因带来的延迟使得不同观察者看到不同的时序，因而有些文献中所谓分布式计算模型即指真并发模型。

我们认为交错模型与真并发模型的划分应针对抽象并发模型这一层次，其本质在于模型中是否用不确定性取代并发性，而在具体并发模型这一层次是不合适的。例如 R. Milner 用带标号变迁系统描述 CCS 的交错语义时，CCS 是一个交错模型；而 U. Montanari 采用扩展 C/E 网给出 CCS 的真并发语义时，它又是一个真并发模型[1]。

由于在交错模型中并发被看作是原子动作的不确定交错，即动作之间的因果关系或独立关系被抽象掉，所以交错模型比真并发模型更抽象，更适合作为并发系统外部行为规格说明的语义基础，而真并发模型则更适合作为并发系统实现的规格说明的语义基础，特别是当实现需要关注系统性能时。

5）*线性时间模型与分支时间模型*　这两者对计算中不确定选择的处理手法不同。线性时间模型，如 Hoare 踪迹、Mazurkiewicz 踪迹、偏序多重集，用可能的执行集或可能的计算路径来表达系统的所有不确定行为；而分支时间模型，如变迁系统、同步树、Petri 网、事件结构则显式表达了计算过程中不确定选择的更详细信息，这些信息对于描述系统的安全性质（如是否有死锁）是必需的。

在线性时间模型中不确定选择结构被抽象掉，所以线性时间模型比分支时间模型更抽象。如果涉及系统安全方面的性质，则应使用分支时间模型；如果只考虑活性方面的性质，则使用线性时间模型就足够了。

6）*其他分类*　从考察并发系统的侧重点出发，还可分为内涵模型和外延模型，内涵模型如带标号变迁系统、Petri 网，显式描述并发系统的内部结构，包括系统状态及动作所引起的状态变化。外延模型如踪迹语言、事件结构，记录并发系统外部行为的发生。内涵模型通常比外延模型表达了更多的信息，并且所建造的系统要小于外延模型。

从研究方法出发，还可分为基于代数的模型、基于逻辑的模型、基于自动机的模型等。

1.2 并发模型的抽象层次

建模与抽象是计算机科学中两个密切相关的核心概念，在并发系统的分析、设计、实现与验证中也不例外，我们应选择合适的抽象层次来建立并发系统的模型。由于有太多的因素要考虑，所以用单一抽象层次的描述方法来处理复杂系统是远远不足的。

众多不同的并发模型的提出、研究与应用正是适应了这种需求，我们认为这些不同模型之间最本质的区别是处理并发性的抽象层次不同，例如真并发模型与交错模型并不存在哪个更好的问题，这是两个不同抽象层次的模型，各自适用于不同的环境，由用户根据系统开发的出发点决定。

两个并发系统的行为关系一直是人们关心的重要课题。CSP、CCS、π-演算等模型利用双模拟研究进程之间的直接等价、强等价、观察等价以及相应的等效关系，并且提供一些消隐机制（如CSP的隐藏算子、CCS的限制算子）用于建立同一模型框架内的不同抽象层次。但如果要冲破单一模型框架的约束，就必须研究不同并发模型之间的关系。

并发系统的规格说明与实现处于两个不同的抽象层次，验证实现相对于规格说明的正确性、并发系统的优化以及规格说明的逐步求精都需要研究不同并发模型之间的关系。当规格说明与实现采用同一模型（如CCS或带标号变迁系统）描述时，上述问题退化为研究两个CCS行为表达式或两个变迁系统之间的行为关系，但这要求一个广谱的单一并发模型能够表达用户需求的各种抽象层次，从传统串行系统的开发经验来看这是十分困难的。因而对不同的并发模型进行分类，形成一种比较完整的并发模型抽象层次，不仅具有深远的理论意义，而且有很大的实际应用价值。

2 交错模型

2.1 带标号变迁系统

带标号变迁系统为变迁系统扩充了标号，使之适合描述并发与分布式程序的结构化操作语义[1]，成为理论计算机科学中最常用、最基础的模型。例如带标号变迁系统为CSP、CCS、π-演算等模型建立了操作语义，并且用于定义进程之间的行为等价性。本文中带标号变迁系统也简称为变迁系统。

变迁系统是一个四元组$(Q,q_0,L,\vdash\!\!\rightarrow)$，其中$Q$是可数状态集，$q_0\in Q$是初态，$L$是可数标号集，$\vdash\!\!\rightarrow\subseteq Q\times L\times Q$是变迁集。变迁$(q,l,q')$通常记为$q\xrightarrow{l}q'$，它是进程的一个原子动作，动作名字为$l$，这种记号很容易扩充到标号组成的串。如果$\exists\, s\in L^*\cdot q_0\xrightarrow{s}q$，则称状态$q$为可达的，变迁的依次发生描绘了进程的一条计算路径，如果任意两个标号相同的变迁的前、后状态不完全相同，则称该变迁系统为扩展的。

如果变迁系统$T=(Q,q_0,L,\vdash\!\!\rightarrow)$中的任一状态$q\in Q$都是可达的，且任一标号$l\in L$都存在$a\xrightarrow{l}a'$，则称$T$是可达的；如果$\forall\, q\in Q,s\in L^*\cdot q\xrightarrow{s}q\Rightarrow s=\varepsilon$，则称$T$是无环的；空变迁$\tau$是不出现于任何标号集的特殊符号，引入空变迁后可将变迁系统上的偏函数扩展为全函数，简化了问题的表述。

给定两个变迁系统$T=(Q,q_0,L,\vdash\!\!\rightarrow)$和$T'=(Q',q'_0,L',\vdash\!\!\rightarrow')$，二元关系$R\subseteq Q\times Q'$称为模拟当且仅当$aRb$且$(a,l,a')\in\vdash\!\!\rightarrow$时存在$(b,l,b')\in\vdash\!\!\rightarrow'$使得$a'Rb'$。如果$R$与$R^{-1}$都是模拟关系，则称$R$为双模拟。如果存在一个双模拟关系$R$且$q_0Rq'_0$，则称变迁系统$T$和$T'$为双相似的，记为$T\sim T'$。

对于进程$a.b.Nil+b.a.Nil$和$a.Nil|b.Nil$，它们对应的带标号变迁系统是相同的，如图1所示。这正是交错模型与真并发模型的区别所在。

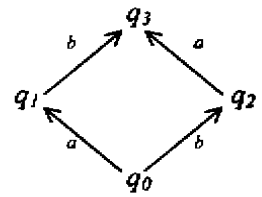

图1

2.2 同步树

R. Milner提出CCS模型时曾利用同步树给出CCS的操作语义[2]，将进程上的各种算子映射为同步树上的操作。同步树的特点是直观、易于理解，同时也有相应的线性表示方法。同步树摒弃了变迁系统中的循环结构，可由变迁系统展开得到。我们将同步树看作变迁系统的特例，定义为满足如下条件的变迁系统$(Q,q_0,L,\vdash\!\!\rightarrow)$：

(1)连通性：$\forall\, q\in Q.\exists\, s\in L^*\cdot q_0\xrightarrow{s}q$；

(2)无环性：$\forall\, q\in Q,s\in L^*\cdot q\xrightarrow{s}q\Rightarrow s=\varepsilon$；

(3)单亲性：$\forall\, q,q',q''\in Q,\forall\, a,b\in L.\ q'\xrightarrow{a}q\wedge q''\xrightarrow{b}q\Rightarrow a=b\wedge q'=q''$。

2.3 Hoare踪迹语言

踪迹语言源于形式语言与自动机的研究，其目标是通过顺序的观察记录描述并发系统的非顺序行为。典型的踪迹语言有C. A. R. Hoare提出的踪迹语言[4]

和 A. Mazurkiewicz 提出的踪迹语言[6]，前者是交错模型而后者是真并发模型。

Hoare 踪迹语言定义为二元组(Σ, L)，其中 Σ 是标号集，即语言的字母表；L 是 Σ^* 的非空子集，满足前缀封闭性，即$\forall s\in\Sigma^*, a\in\Sigma. sa\in L\Rightarrow s\in L$。

Hoare 踪迹语言既用于描述 CSP 的系统规范，也用于定义 CSP 中进程行为的语义。踪迹语言摒弃了同步树中的不确定选择结构，因而变迁系统与同步树是分支时间模型，而 Hoare 踪迹语言则是线性时间模型。作为线性时间模型，Hoare 踪迹语言无法表达一些重要的系统安全性特征。例如根据踪迹 ab 无法区别可能出现的两种情况：一种是还能进一步计算，一种是已进入死锁。

3 真并发模型

真并发模型显式地给出两个动作之间的并发性。在一些模型中动作之间的独立关系或因果依赖关系作为原始概念，如 Mazurkiewicz 踪迹语言；而一些模型则以更原始的概念定义动作之间的并发性，如 Petri 网，在 P/T 系统中两个变迁可能在某些标识下是并发的，而在另一些标识下却是冲突的，即这种独立关系或冲突关系应视为动作某次发生的性质，而不是动作本身的性质。因此，许多真并发模型是基于事件的模型。

3.1 异步变迁系统

异步变迁系统是在带标号变迁系统基础上扩充的一种真并发模型。在带标号变迁系统中，一个变迁表示系统中的一个动作，即可观察的系统行为；而异步变迁系统则将一个变迁看作一个事件的发生，事件之间可定义独立关系。

异步变迁系统是五元组$(Q, q_0, E, I, \vdash)$，其中$(Q, q_0, E, \vdash)$是带标号变迁系统，而独立关系 $I\in E\times E$ 是对称且反自反的，满足：

(1)$e\in E\Rightarrow\exists\ q, q'\in Q. q\xrightarrow{e}q'$；

(2)$q\xrightarrow{e}q'\wedge q\xrightarrow{e}q''\Rightarrow q'=q''$；

(3)$e_1 I e_2\wedge q\xrightarrow{e1}q_1\wedge q\xrightarrow{e_2}q_2\Rightarrow\exists\ q'\in Q. q_1\xrightarrow{e2}q'\wedge q_2\xrightarrow{e1}q'$；

(4)$e_1 I e_2\wedge q\xrightarrow{e1}q_1\wedge q_1\xrightarrow{e2}q'\Rightarrow\exists\ q_2\in Q. q\xrightarrow{e2}q_2\wedge q_2\xrightarrow{e1}q'$。

其中，公理(1)表明每一事件都作为变迁出现，公理(2)说明每一事件发生后导致状态变迁是唯一的，公理(3)和(4)给出了独立关系的性质。

3.2 事件结构

事件结构抽取了 Petri 网事件之间最基本的两种关系[12]：一是因果依赖关系，即变迁之间的顺序关系；一是表达不确定性的选择关系，包括前、后冲突关系和冲撞关系(冲撞可引入补位置转化为冲突)。事件结构是一种基于事件的分支时间模型。

事件结构定义为三元组$(E, \leqslant, \#)$，其中 E 是事件集，因果依赖关系$\leqslant$是事件集 E 上的偏序，冲突关系 $\#$ 是事件集 E 上的对称且反自反二元关系，满足：

(1)事件集 E 是可数集；

(2)对任意 $e\in E$，$\{e'\mid e'\leqslant e\}$是有限集；

(3)$\forall e, e', e''\in E. e\#e'\wedge e'\leqslant e''\Rightarrow e\#e''$。

有限性约束(2)表明只考虑离散进程，公理(3)表明冲突关系具有继承性。

如果 $e\leqslant e'$则称 e 导致 e'，又称 e'因果依赖于 e。如果事件 $e, e'\in E$ 满足$\neg(e\leqslant e'\vee e'\leqslant e\vee e\#e')$则称 e 和 e'是并发的，记为 $e\,co\,e'$。

事件结构计算过程中的状态定义为格局 $D(E, \leqslant, \#)$，由满足以下条件的 $X\subseteq E$ 组成：

·无冲突：$\forall e, e'\in X. \neg(e\#e')$，即 $X\times X\cap\#=\varnothing$；

·封闭性：$\forall e\in X, e'\in E. e'\leqslant e\Rightarrow e'\in X$，

记 $D^0(E, \leqslant, \#)$为所有有穷格局的集合，两个事件之间的因果依赖、冲突和并发关系也可用有穷格局来定义：

·$e\leqslant e'\Leftrightarrow\forall\ x\in D^0(E, \leqslant, \#). e'\in x\Rightarrow e\in x$；

·$e\#e'\Leftrightarrow\forall\ x\in D^0(E, \leqslant, \#). e\in x\Rightarrow e'\notin x$；

·$e\,co\,e'\Leftrightarrow\exists\ x, x'\in D^0(E, \leqslant, \#). e\in x\wedge e\notin x'\wedge e'\in x'\wedge e'\notin x\wedge x\cup x'\in D^0(E, \leqslant, \#)$；

定义主格局$\lceil e\rceil=\{e'\in E\mid e'\leqslant e\}$；对于任意格局 $X, e\in X\Rightarrow\lceil e\rceil\subseteq X$。格局 x 到 x'之间的转换定义为 $e\notin x\wedge x'=x\cup\{e\}$，记为 $x\xrightarrow{e}x'$。如果事件结构中的事件具有字母表 Σ 上的标号，则成为带标号事件结构$(E, \leqslant, \#, lab, \Sigma)$，其中 $lab: \Sigma\leftarrow E$ 是标注函数。

3.3 Mazurkiewicz 踪迹语言

70年代末提出的 Mazurkiewicz 踪迹语言是嵌入了独立关系的最简单并发模型[6]，它定义为三元组(Σ, L, I)，其中 Σ 是字母表，L 是 Σ^* 的非空子集，独立关系 $I\subseteq L\times L$ 是对称且反自反的，满足：

(1)前缀封闭性：$\forall s\in\Sigma^*, a\in\Sigma. sa\in L\Rightarrow s\in L$；

(2)独立关系封闭性：$\forall s, t\in\Sigma^*, \forall a, b\in\Sigma. sabt\in L\wedge aIb\Rightarrow sbat\in L$；

(3)一致性：$\forall s\in\Sigma^*, \forall a, b\in\Sigma. sa\in L\wedge sb\in L\wedge aIb\Rightarrow sab\in L$。

Mazurkiewicz 踪迹语言字母表中的元素应理解为事件而不是动作，每个事件可通过标注函数与标号建立联系。独立关系是事件之间而不是动作或标号之间的关系，串则应理解为按时间发生的事件序列。公理(2)和(3)表明当 a 和 b 相互独立时，$sabt$ 和 $sbat$ 代表

相同的计算，记为 $sabt \equiv sbat$。由等价关系≡定义的等价类称为踪迹。串之间的拟序 $s \prec t$ 定义为 $\exists u \in \Sigma^* . su \equiv t$。≺关于等价关系≡的商≺/≡是踪迹上的偏序。

4 抽象并发模型的层次

各种并发模型很好地体现了不同的抽象层次，例如：几乎所有的模型都忽略了事件发生的具体时间和地点，其中的动作都是不可分割的原子单位。带标号变迁系统等交错模型将动作之间的并发关系抽象为不确定选择；同步树排除了变迁系统中的循环结构；Hoare 踪迹语言进一步摒弃同步树中的不确定选择结构，成为一个线性时间模型。真并发模型保留并发关系为基本概念，Mazurkiewicz 踪迹语言是 Hoare 踪迹语言的扩展，带标号事件结构则是同步树的扩展，异步变迁系统是带标号变迁系统的扩展。

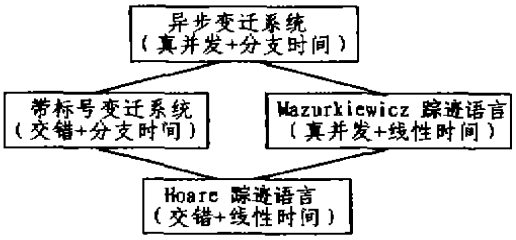

图2 采用二维正交坐标分类的并发模型层次实例

任给分类角度 α，如果从该角度看模型 A 是模型 B 的抽象，则定义偏序关系 $A \prec_\alpha B$，表示模型 B 比 A 在这一方面包含更多的信息量，或模型 A 忽略了不关心的 B 中这一方面的信息。从不同角度对并发模型进行分类相当于提供了多维正交坐标 $(d_1, d_2, \cdots, d_n)$，对应每一个 d_i 存在一种偏序关系 $\prec_i$，从而不同的并发模型形成一个有限层次完全格。

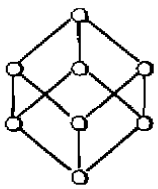

图3 采用三维正交坐标分类的并发模型层次结构

例如 $n=2$，$d_1=$（交错≺真并发），$d_2=$（线性时间≺分支时间），则以此为分类依据将形成如图2所示的并发模型层次；$n=3$ 时通常形成如图3所示的层次结构。分类依据的正交坐标不仅可使用本文介绍的各种分类角度，而且还可包括模型对优先级的表达能力、对时间与概率的表达能力等。这种分类层次也可借用面向对象程序设计中的继承机制组织起来，但其中必须允许使用多重继承和重复继承。

5 范畴论在并发模型研究中的应用

对并发模型及其关系的研究可借助范畴论作为形式化数学工具。范畴论提供了一种与集合论不同的思维方式，从一个更抽象的层次观察对象、集中考察对象之间的关系而不是对象的内部结构[5]。直观上看，各类并发模型分别形成相应的范畴，同一模型内部的关系表现为范畴中的射，而不同模型之间的关系则表现为范畴间的函子。因而范畴论可作为研究不同模型之间关系的一个强有力工具。

5.1 基本思路

类似 Petri 网研究的特殊网论和通用网论[2]，利用范畴论研究并发模型可分为两类。第一类是继续以单个并发模型为研究对象，为该模型构造合适的射从而形成一个范畴，在这个范畴中进程演算的各种算子成为泛构造。

第二类则以所有的并发模型为研究对象，主要考察不同模型之间的关系，为不同模型之间的转换提供理论依据[3,6-7]。例如可用反射与共反射这两种伴随来表达两个数学表示形式完全不同的模型之间是否具有嵌入关系，或给出不同模型按各自方式定义的并行复合的统一表示。在代数语义学研究中也有类似方法，即利用范畴论为基础对初始语义、终结语义、格语义等进行模型分类[1]。

5.2 带标号变迁系统范畴

建立变迁系统范畴的首要问题是定义合适的射。射的定义应保持初态和变迁：变迁系统 $T=(Q, q_0, L, \vdash)$ 到 $T'=(Q', q'_0, L', \vdash')$ 的映射定义为二元组 (δ, σ)，其中全函数 $\delta: Q' \leftarrow Q$ 满足 $\delta(q_0)=q'_0$，偏函数 $\sigma: L' \leftarrow L$ 满足对任意 $(q, l, q') \in \vdash$，如果 $\sigma(l)$ 有定义则 $(\delta(q), \sigma(l), \delta(q')) \in \vdash'$，否则 $\delta(q)=\delta(q')$。以所有变迁系统为对象，上述映射为射形成的范畴记为 **LTS**。类似可建立同步树范畴 **ST** 和 Hoare 踪迹语言范畴 **HTL**。

建立范畴 **LTS** 后，变迁系统上的并行复合、不确定选择、限制、重标号、前缀等算子可看作范畴 **LTS** 中的积、共积、笛卡尔提升、共笛卡尔提升等泛构造[12]。

5.3 并发模型范畴之间的关系

显然范畴 **ST** 是 **LTS** 的完全子范畴，包含函子 $st2lts$：**LTS**←**ST** 是一个完全函子。如果将变迁系统中所有可能发生的变迁序列作为状态，变迁序列的增长作为新的变迁，则可将一个变迁系统展开为同步树，据此定义的映射 $lts2st$ 可扩展为 **ST**←**LTS** 的函子，它是包含函子 $st2lts$ 的右伴，从而范畴 **ST** 是 **LTS** 的共反射

子范畴、右伴 *lts2st* 也称共反射子。根据范畴论的结论，同步树 T 与其展开 $lts2st(T)$ 是同构的[5]。

任给一个 Hoare 踪迹语言，以串为状态、串的增长为变迁可映射为相应的同步树，从而定义函子 *htl2st*：**ST←HTL**；任给一个同步树，可将所有可能出现的变迁序列对应的标号串作为踪迹语言，从而形成函子 *st2htl*：**HTL←ST**。函子 *st2htl* 和 *htl2st* 形成了从 **ST** 到 **HTL** 的伴随，该伴随具有左伴 *st2htl* 和右伴 *htl2st*。

由此可见，不同并发模型之间的转换可利用反射和共反射这两种伴随来描述。反射和共反射与包含函子密切相关，在交错模型中从范畴 **HTL** 到 **ST** 到 **LTS** 依次存在包含函子说明了这些模型之间的抽象层次关系。每一包含函子具有的左伴（反射）或右伴（共反射）分别对应着不确定选择结构的抽象与循环结构的抽象。

结束语 对并发模型的分类从不同的正交坐标出发，形成一个完整的分类体系。这些研究成果至少为我们带来两点启示。首先，借鉴门捷列夫对化学元素周期表的研究方法进一步完善并发模型的谱系。我们可找出一些新的并发模型，例如建立一种交错的、线性时间的系统模型；也可扩充新的正交坐标，例如为各类并发模型引入优先级。更重要的是通过分类真正掌握众多不同并发模型的特性与区别，从而在实际应用中根据需要正确选择使用某一成熟的模型或提出新的模型。

其次，S. Smolka 将并发研究工作分为模型、逻辑、语言、算法与方法论、工具以及应用六大领域[11]，而并发系统开发方法学的匮乏是严重阻碍并发应用发展的瓶颈。我们的下一步工作将在并发模型分类体系的基础上提出一种基于模型变换的逐步求精并发系统开发与验证方法。

主要参考文献

1 陆汝钤. 计算机语言的形式语义. 科学出版社，1992

2 袁崇义. Petri 网原理. 电子工业出版社，1998

3 Degano P. , et al. On Relating Some Models for Concurrency. In: Proc. of TAPSOFT'93, Lecture Notes in Computer Science, Vol. 668, Springer, 1993. 15～30

4 Hoare C. Communicating Sequential Processes. Prentice Hall International, 1985

5 MacLane S. , Categories for the Working Mathematics. Graduate Texts in Mathematics, Springer, 1971

6 Mazurkiewicz A. Introduction to Trace Theory. In: Diekert V, Rozenberg G. eds, Book of Traces. World Scientific, 1994. 3～41

7 Milner R. A Calculus of Communicating Systems. Lecture Notes in Computer Science, 1980, 92

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

9 Prasad K. A Calculus of Broadcast Systems. In: Proc. of TAPSOFT' 91, Vol. 1: CAAP, Lecture Notes in Computer Science, Vol. 493, Springer, 1991

10 Sassone V, et al. Models for Concurrency. Towards a Classification. Theoretical Computer Science, 1996, 170: 297～348

11 Smolka S. First Draft of Concurrency Working Group Report. ACM SDCR workshop, MIT, June 1996

12 Winskel G. , Nielsen M. Models for Concurrency. In: Abramsky S, et al. , eds. Handbook of Logic in Computer Science. Oxford University Press, 1995. 1～148

（上接第42页）

```
动作:insert-policy-reservation(g,p)
  I:(n:reservation#,g:agency,p:policy-type,A:assured,a:
    arrival-date,d:depature-date)
  O:(policy-reservation)
  If policy-exists(p)且 legal-assured(A) then
      得到 arrival-date a;
      得到 reservation# n;
      录入投保单(n,p,g,a,d)
  end if
  If  投保单(n,p,g,a,d)录入成功 then
      policy-reservation:插入(n,p,g,A,a,d)
  end if
```

图8 插入投保单动作

结论 本文基于保险信息管理系统开发的工程实践，探讨了把面向对象的方法用于 MIS 的概念模型设计。本文重点对对象行为进行了比较透彻的分析。这里对对象行为引入了动作抽象和事务抽象。本文所采用的方法是半形式化的，用这种方法构造出的模型易理解、易实现、易修改。

参考文献

1 Brodie M L, et al. On Conceptual Modelling. Springer-Verlag, 1986

2 Hardgrave B C, et al. Comparing Object-Oriented and Extended-Entity-Relationship Data Models. Journal of Database Management, 1995, 6(3)

3 Liu Ling, et al. The Building Blocks for Specifying Communication of Complex Objects, An Activity-Driver Approach. ACM Transactions on Database Systems, 1996, 21(2)

4 Wang Shouhong. Object-oriented task analysis. Information & Management, 1995, 29: 331～341

5 Coad P, Yourdon E 著，邵维忠，杨芙清等译，面向对象的设计. 北京大学出版社，1994