第39卷 增刊(2)
2000年 12月

中山大学学报(自然科学版)
ACTA SCIENTIARUM NATURALIUM
UNIVERSITATIS SUNYATSENI

Vol.39 Suppl.(2)
Dec. 2000

文章编号:0529-6579(2000)S2-0165-07

并发模型的范畴论方法*

李文军,周晓聪,李师贤
(中山大学计算机科学系,广州 510275)

摘 要:范畴论是一种通用概念框架,可作为软件工程的理论基础.介绍范畴论不同于集合论的独特思维方式,并探讨将这些概念与思想作为数学工具应用到语义模型研究中的一般方法,最后给出利用范畴论研究并发模型以及模型间关系的应用实例.

关键词:并发模型;范畴论
中图分类号:TP301 **文献标识码**:A

范畴是群、环、域等抽象数学结构的进一步抽象,其研究重点在于对象之间的关系而不是对象的内部结构.软件工程形式化方法的许多概念在范畴论中可得到更清晰的解释,因而范畴论被视为计算机科学中强有力的数学工具,广泛应用在函数式语言设计、多态类型系统、并发模型等领域.

1 范畴论思维方式

范畴用集合论术语定义,但范畴Set并不是要用范畴论术语重新定义集合,而只是将一个众所周知的领域表示为范畴.范畴论的各种泛构造给出抽象的规格说明,而在集合论中的相应解释可看作一种具体的实现.范畴论考虑问题的方式与传统集合论有很大差别.

(1) 范畴论注重对象间的关系而不是对象的内部结构,因而比集合论更适合建立较高抽象层次的模型.例如集合论定义笛卡尔积时明确给出结果集合的元素组成,但范畴论则从结果对象与其他对象之间的关系来定义两个对象的积,结果对象被当作黑盒,其元素组成被抽象掉.这种区别还体现在均衡、幂等其他泛构造.由于这一原因,集合论中普遍采用的归纳法很少用于范畴论的证明.

(2) 范畴论采用比集合论更简便的表达方式研究函数的性质.例如集合论以函数的定义域与值域的特性定义满射与内射,但在范畴论中射$f:A \leftarrow B$是单射定义为$\forall g, h:B \leftarrow C, f \cdot g = f \cdot h \Rightarrow g = h$.证明两个单射$f:A \leftarrow B$和$g:B \leftarrow C$的复合仍是单射可采用更简便的推导:$\forall h, i:C \leftarrow D, f \cdot g \cdot h = f \cdot g \cdot i \Rightarrow g \cdot h = g \cdot i \Rightarrow h = i$.

(3) 范畴论比集合论更多地利用了对称性.范畴论中的许多概念都是成对出现的,如

* 基金项目:高等学校博士学科点专项科研基金资助项目(99-018-411703)
收稿日期:2000-03-13; 作者简介:李文军(1966~),男,副教授.

积和共积、单射和满射等．对偶原则简化了这些对称概念的性质证明：如果陈述 S 对所有范畴成立，则对偶陈述 S^{op} 对所有范畴也成立．例如已知单射的复合仍是单射，由对偶原则直接可得满射的复合仍是满射．

(4) 范畴论是一种“强类型”的语言．函数 $f(x) = x^2$ 会因为函数的类型不同而在范畴论中作为不同的射来研究，如 $f_1: N^2 \leftarrow N$、$f_2: N^2 \leftarrow I$、$f_3: I^2 \leftarrow I$ 等．交换图为类型及类型转换提供了直观的表达方式．

(5) 范畴论支持不同的抽象层次．以范畴为对象、范畴之间的函子为射可得到一个关于范畴的范畴；以两个范畴之间的函子为对象、这些函子之间的自然变换为射可得到一个函子范畴．而这些不同抽象层次的范畴均可采用类似的方法来研究．

2 范畴论在语义模型的应用途径

范畴论作为一种数学工具广泛应用于理论计算机科学，有必要探讨借助范畴论研究不同目标系统的思路、步骤等普遍规律．通过总结范畴论在函数式语言、类型系统、并发模型等领域的应用，我们得出结论：建立范畴模型实际上是一个逐步求精的过程，在这一过程中范畴模型被不断细化，同时越来越多的目标系统特性被刻划出来．

2.1 给出目标系统最基本的范畴论解释

最简单的应用是给出目标系统基本现象的范畴论解释，即从范畴论观点看待这些现象．这步工作将目标系统的现象映射到范畴定义中的对象、射、复合等，并验证满足范畴定义中的约束（特别是复合的结合律）．这是建立目标系统范畴模型的第一步，即使这样简单的应用也有利于抽取不同现象以至不同领域之间的共性．例如范畴逻辑给出了逻辑的范畴论解释：公式作为对象，证明是射，对象 A 到 B 的射是公式 A 与 B 之间的逻辑蕴含关系，恒等射是自反公理的实例，复合反映逻辑蕴含关系的传递性，因而满足结合律．

习惯上实体、集合元素等理解为对象，但给出范畴论解释时未必总是如此，有时需理解为射，射在范畴论中的作用远比对象重要．例如独异点 $(M, \circ, e)$ 理解为范畴时，并不是将 M 中元素作为对象、二元运算作为射，正确解释是范畴中只有一个对象，M 中元素作为从该对象到自身的射，单位元 e 对应恒等射，二元运算对应射的复合．

2.2 不断引入约束以形成更精确的范畴模型

第一步建立了最抽象的范畴模型，提炼了目标系统最基本的性质．为刻划不同目标系统各自的特性，需进一步为范畴增加约束条件，形成笛卡尔闭范畴（CCC）、Allegory、Fibration 等特殊范畴概念．这些范畴中有新的性质要挖掘与证明，并用于解释目标系统相应特性．范畴模型的这一求精过程可能需重复多次，最终才能使模型足够精确．

CCC 与 λ 理论的联系[1]很好地体现了这一思路．CCC 为范畴增加 3 个约束条件：要求有终结对象以解释基本类型，要求有二元积以解释二元组类型，要求有幂以解释函数类型．任给 λ 理论 L，可构造相应CCC(L)：对象是 L 中的类型；A 到 B 的射是具有一个类型 A 的自由变元且返回类型为 B 的项；多变元的项对应那些域为积对象的射；将 L 中的置换转换为 CCC (L)中的复合运算，λ 演算中的 β 转换和 η 转换规则正好反映了笛卡尔封闭性．这种范畴模型为类型化 λ 演算提供了方便的代数处理方法，成为研究多态类型系统语义的基础．

2.3 根据范畴模型指导实际工作

范畴模型不仅提供统一、通用的概念框架，还可用于指导实际工作．例如 C++ 这类支持多态运算符与隐式类型转换的语言中，如果变量 i，j 为 *int* 类型，x 为 *float* 类型，涉及多态运算符 *op* 的运算 $x = i\ op\ j$ 应理解为将 i，j 转换为 *float* 后执行 op_{float}，还是执行 op_{int} 后将结果转换为 *float*？这两种做法是否等价从而可由编译器根据优化需要自由选择？

类型的隐式转换是一种偏序，以此为射可形成范畴 **Ω**．类型 σ 可隐式转换为 τ（记为 $\sigma \leqslant \tau$）意味着存在 σ 值集到 τ 值集的转换函数，这种映射关系可表达为函子 B：**Set**←**Ω**．如果证明了图 1 可交换，则编译器可按任意次序进行隐式转换与多态运算，其中 γ_{op}（T，T）是类型 T 上的运算．γ_{op} 还可看作自然变换，从而利用伴随给出更简洁的描述[1]．

$$\begin{array}{ccc} B(int) \times B(int) & \xrightarrow{\gamma_{op}(int,\,int)} & B(int) \\ \downarrow & & \downarrow \\ B(int \leqslant float) \times B(int \leqslant float) & & B(int \leqslant float)\ B(float) \\ \downarrow & & \\ B(float) \times B(float) & \xrightarrow{\gamma_{op}(float,\,float)} & B(float) \end{array}$$

图 1　类型隐式转换的交换图

2.4 通过分析范畴模型寻求新的机制

这是范畴论应用在理论计算机科学的最高境界．*Bjorner* 指出：形式化方法除为现有机制的陈述、证明提供框架外，还是一种发现新机制的途径．例如通过对面向对象类型系统的研究，寻找新的类型构造方法，形成新的程序设计机制；通过对各种并发模型间关系的研究，整理并发模型的抽象层次，为验证并发系统的实现相对于规格说明的正确性或规格说明的求精提供形式化方法．

3 范畴论在并发模型研究中的应用

范畴论提供了一种思维方式和通用指南，让我们从更抽象的层次观察对象，集中考察对象间关系而不是对象内部结构．利用范畴论可将一些直观或数学的概念统一用范畴论术语重新阐述，从而寻找足够通用与抽象的特征，使这些特征适用于不同的并发模型．类似 *Petri* 网研究分特殊网论和通用网论，目前利用范畴论对并发模型的研究主要有 2 类：

第 1 类继续以单个并发模型为研究对象，为该并发模型构造合适的射从而形成一个范畴．进程演算的各种算子可解释为范畴中的泛构造，定义射时通常要求保持行为，并注意对动作粒度的选择．一种常用的射是将行为映射到其子成分．

第 2 类以所有并发模型为研究对象，考察不同模型间的关系，为模型转换提供理论依据．例如可用反射与共反射表达两个数学表示形式完全不同的模型是否具有嵌入关系（即模型 $M1$ 是模型 $M2$ 的抽象），或给出不同模型按各自方式定义的并行复合的统一表示．类似研究有代数语义学利用范畴论对初始语义、终结语义、格语义等进行模型分类[2]．

4 范畴论应用在单个并发模型研究

本节以带标号变迁系统（*LTS*）为例介绍范畴论在单个并发模型研究的应用方法[3]．*LTS* 在变迁系统的基础上引入标号，使之适合描述并发程序的结构化操作语义，成为理论计算机科学最常用、最基础的模型．*CSP*、*CCS*、π 演算等并发模型不仅利用 *LTS* 定义操作语义，而且还用于定义进程的行为等价性．本文中 *LTS* 也简称为变迁系统．

4.1 带标号变迁系统

变迁系统是四元组 $(Q, q_0, L, \mapsto)$，其中 Q 是可数状态集，$q_0 \in Q$ 是初态，L 是可数标号集，$\mapsto \subseteq Q \times L \times Q$ 是变迁集. 变迁 (q, l, q') 通常记为 $q \xrightarrow{l} q'$，表示系统在状态 q 执行原子动作 l 后进入状态 q'. 这种记号很容易扩充到标号组成的串，如果存在串 s 使得 $q_0 \xrightarrow{s} q$，则称状态 q 是可达的.

4.2 带标号变迁系统范畴 LTS

建立 *LTS* 的首要问题是定义合适的射，射应保持初态和变迁：$T = (Q, q_0, L, \mapsto)$ 到 $T' = (Q', q_0', L', \mapsto')$ 的射是二元组 (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 $(q, l, q') \in \mapsto$，如果 $\sigma(l)$ 有定义则 $(\delta(q), \sigma(l), \delta(q')) \in \mapsto'$，否则 $\delta(q) = \delta(q')$. 将 σ 定义为偏函数使处理进程并行复合时不仅可改变动作名字，还允许将不关心的动作隐藏起来.

引入不属于任何标号集的空变迁 τ，满足 $(\forall q \in Q, (q, \tau, q) \in \mapsto) \wedge ((q, \tau, q') \in \mapsto \Rightarrow q = q')$. 定义 $T = (Q, q_0, L, \mapsto)$ 的闭包 $T^* = (Q, q_0, L^*, \mapsto^*)$，其中 $L^* = L \cup \{\tau\}$，$\mapsto^* = \mapsto \cup \{(q, \tau, q) \mid q \in Q\}$. 将 τ 和所有无定义 l 映射为 τ，则可将 σ 扩展为全函数 $\sigma^*: L'^* \leftarrow L^*$，满足 $(q, l, q') \in \mapsto \Rightarrow (\delta(q), \sigma^*(l), \delta(q')) \in \mapsto'^*$. T^* 到 T'^* 的射定义为 (δ, σ^*). 这样不必考虑 σ 为偏函数带来的问题，在上下文清楚时不严格区别 T 与 T^*.

以所有变迁系统闭包为对象，上述二元组映射为射形成的范畴记为 LTS.

4.3 范畴 LTS 上的泛构造

建立范畴 *LTS* 后，变迁系统上的并行复合、不确定选择、限制、重标号等各种进程算子可看作范畴中的泛构造. 这些泛性质很容易推广到其他并发模型.

4.3.1 *并行复合* 任给变迁系统 $T_1 = (Q_1, q_{01}, L_1, \mapsto_1)$ 和 $T_2 = (Q_2, q_{02}, L_2, \mapsto_2)$，$T_1$ 与 T_2 的积 $T_1 \times T_2$ 定义为变迁系统 $T = (Q, q_0, L, \mapsto)$，其中，$q_0 = (q_{01}, q_{02})$；$Q = Q_1 \times Q_2$，投影函数分别记为 θ_1 和 θ_2；$L = L_1^* \times L_2^* - \{(\tau, \tau)\}$，投影函数分别为 π_1 和 π_2；$(q, l, q') \in \mapsto^* \Leftrightarrow (\theta_1(q), \pi_1(l), \theta_1(q')) \in \mapsto_1^* \wedge \theta_2(q), \pi_2(l), \theta_2(q')) \in \mapsto_2^*$.

并行复合对应的泛构造是范畴 LTS 中的积. 二元积可推广到多元积情况，定义零元积为 $Nil = (\{q_0\}, q_0, \emptyset, \emptyset)$，*Nil* 既是范畴 LTS 中的终结对象也是初始对象. 以上泛构造描述了所有可能的同步，结合限制与重标号则可在并行复合时屏蔽某些同步动作.

4.3.2 *不确定选择* T_1 与 T_2 的和 $T_1 + T_2$ 定义为变迁系统 $T = (Q, q_0, L, \mapsto)$，其中，$q_0 = (q_{01}, q_{02})$；$Q = (Q_1 \times \{q_{02}\}) \cup (\{q_{01}\} \times Q_2)$，射影函数分别记 i_1 和 i_2；L 是 L_1 与 L_2 的不相交并，射影函数分别记 j_1 和 j_2；$((q, l, q') \in \mapsto_1 \Rightarrow (i_1(q), j_1(l), i_1(q')) \in \mapsto) \vee ((q, l, q') \in \mapsto_2 \Rightarrow (i_2(q), j_2(l), i_2(q')) \in \mapsto)$.

不确定选择对应的泛构造是范畴 LTS 中的共积，二元共积可扩展为任意元. 利用重标号可满足不确定选择算子不改变进程字母表的要求，即先用重标号将 T_1 和 T_2 的标号集都扩大为 $L_1 \cup L_2$，从而不确定选择 $T_1[L_1 \cup L_2] + T_2[L_1 \cup L_2]$ 无需改变标号集.

4.3.3 *限制* 任给变迁系统 $T = (Q, q_0, L, \mapsto)$ 和标号集 $L' \subseteq L$（范畴术语表达为射影 $\sigma: L \leftarrow L'$），限制定义为 $T \setminus \sigma = (Q, q_0, L', \mapsto')$，其中 $\mapsto' = \{(q, l, q') \mid l \in L' \wedge (q, l, q') \in \mapsto\}$. 限制同时涉及范畴 LTS 和标号集对应的范畴 Set，所以利用函子建立 2 个范畴的联系. 定义忘却函子 *labl*：Set←LTS 将变迁系统映射为标号集，变迁系统之间的射

映射为标号集上的函数. 限制算子对应对象 $T \setminus \sigma$ 和射 $(id_Q, \sigma): T \leftarrow T \setminus \sigma$，具有图 2 所示泛性质：$\forall f: T \leftarrow X, labl(f) = \sigma \Rightarrow \exists ! g: T \setminus \sigma \leftarrow X, labl(g) = id_{L'} \wedge f = (id_Q, \sigma) \cdot g$.

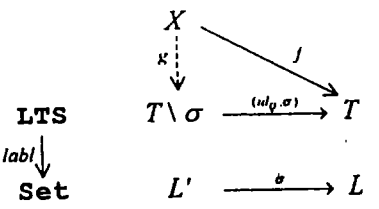


图 2　限制算子对应笛卡尔提升

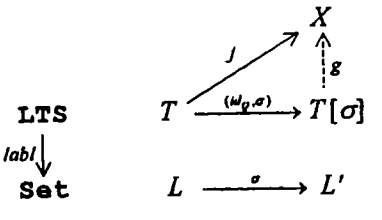


图 3　重标号算子对应共笛卡尔提升

由图 2 可知射 (id_Q, σ) 是 σ 关于 T 的笛卡尔提升[4]. 对任意标号集为 L 的变迁系统 T 和任意 $\sigma: L \leftarrow L'$ 都存在笛卡尔提升，故 $labl$ 是纤维函子，其基范畴和全范畴分别为 **Set** 和 **LTS**. 标号集 L 上的纤维范畴(fiber)是 **LTS** 的子范畴 $labl^{-1}(L)$，其对象满足 $labl(T) = L$，射满足 $labl(f) = id_L$. 由于任意 σ 都具有笛卡尔提升，限制算子实际上定义了一个从 $labl^{-1}(L')$ 到 $labl^{-1}(L)$ 的函子，它作用在射上的效果正是笛卡尔提升的泛性质.

4.3.4　**重标号**　任给变迁系统 $T = (Q, q_0, L, \mapsto)$ 和全函数 $\sigma: L' \leftarrow L$，重标号定义为 $T[\sigma] = (Q, q_0, L', \mapsto')$，其中 $\mapsto' = \{(q, \sigma(l), q') \mid (q, l, q') \in \mapsto\}$. 对象 $T[\sigma]$ 和射 $(id_Q, \sigma): T[\sigma] \leftarrow T$ 具有图 3 所示泛性质：$\forall f: X \leftarrow T, labl(f) = \sigma \Rightarrow \exists !\ g: X \leftarrow T[\sigma], labl(g) = id_{L'} \wedge f = g \cdot (id_Q, \sigma)$.

重标号对应的泛构造是共笛卡尔提升，也可定义从 $labl^{-1}(L)$ 到 $labl^{-1}(L')$ 的函子. 对任意标号集为 L 的变迁系统 T 和任意 $\sigma: L' \leftarrow L$ 都存在共笛卡尔提升，所以 $labl$ 是共纤维函子.

5　范畴论应用在并发模型关系研究

并发系统间的行为关系一直是人们关心的重要课题. CCS、π 演算等模型利用双模拟研究进程的直接等价、强等价、观察等价以及相应的等效关系，并利用消隐机制（如 CSP 的隐藏、CCS 的限制）建立不同抽象层次. 规格说明与实现所处的抽象层次很难由单一的广谱并发模型描述，所以验证正确性、系统优化或规格说明求精等都需要研究不同模型间的关系. 范畴论是研究模型间关系的强有力工具，本节将应用在几种典型的交错模型[5,6].

5.1　同步树范畴 ST 和 Hoare 踪迹语言范畴 HTL

同步树的特点是直观、易理解，同时也给出线性表示法. 同步树可作为变迁系统的特例，定义为满足如下性质的变迁系统 $(Q, q_0, L, \mapsto)$：① 连通性，$\forall q \in Q, \exists s \in L^*, q_0 \xrightarrow{s} q$；② 无环性，$\forall q \in Q, \forall s \in L^*, q \xrightarrow{s} q \Rightarrow s = \epsilon$；③ 单亲性，$\forall q, q', q'' \in Q, \forall a, b \in L, q' \xrightarrow{a} q \wedge q'' \xrightarrow{b} q \Rightarrow a = b \wedge q' = q''$. 利用变迁系统间的映射可建立以所有同步树为对象的范畴 **ST**.

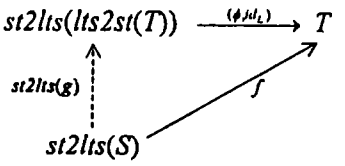
踪迹语言源于形式语言与自动机，通过顺序的观察记录描述并发系统的非顺序行为. Hoare 踪迹语言既可描述 CSP 的规格说明，也用于定义 CSP 进程行为的语义. Hoare 踪迹语言定义为二元组 (Σ, L)，其中 Σ 是标号集，$L \subseteq \Sigma^*$ 满足：① 非空性，$L \neq \emptyset$；② 前缀封闭性，

$\forall s \in \Sigma^*, \forall a \in \Sigma, sa \in L \Rightarrow s \in L$. 任给字母表上偏函数 $\sigma: \Sigma' \leftarrow \Sigma$, 扩充 σ 为串的函数并作为 (Σ, L) 到 (Σ', L') 的映射, 满足 $\forall s \in L \Rightarrow \sigma(s) \in L'$. 以所有 Hoare 踪迹语言为对象、映射 σ 为射形成的范畴记为 **HTL**.

5.2 范畴 LTS 与 ST 之间关系

同步树是变迁系统的特例，变迁系统具有的循环结构在同步树中被抽象掉. 范畴术语表达为 **ST** 是 **LTS** 的完全子范畴，**ST** 到 **LTS** 的包含函子 $st2lts$: LTS←ST 则是一个完全函子. 变迁系统展开后即得同步树，这种展开可形式化定义为函子 $lts2st$: $ST \leftarrow LTS$. 任给变迁系统 $T = (Q, q_0, L, \mapsto)$, 定义 $lts2st(T) = (Q', q_0', L, \mapsto')$, 其中, $q_0' = \varepsilon$, 即空变迁序列作为初态; Q' 的元素是有限变迁序列 $t_1 t_2 \cdots t_n$, 满足 $\forall 1 \leqslant i \leqslant n, t_i = (q_{i-1}, l_i, q_i)$; $\mapsto' = \{(t_1 t_2 \cdots t_n, l, t_1 t_2 \cdots t_{n+1}) \mid n \geqslant 0 \wedge t_1 t_2 \cdots t_{n+1} \in Q' \wedge t_{n+1} = (q_n, l, q_{n+1})\}$.

定义映射 $\phi: T \leftarrow st2lts(lts2st(T))$ 为 $\phi(\varepsilon) = q_0 \wedge (t_n = (q_{n-1}, l, q_n) \Rightarrow \phi(t_1 t_2 \cdots t_n) = q_n)$, 则射 (ϕ, id_L) 在 **LTS** 中有图 4 所示泛性质: $\forall f: T \leftarrow st2lts(S), \exists ! g: lts2st(T) \leftarrow S, f = (\phi, id_L) \cdot st2lts(g)$. 故函子 $st2lts$ 和 $lts2st$ 形成从 **ST** 到 **LTS** 的伴随, 射 (ϕ, id_L) 是该伴随的共单元, $st2lts$ 是 $lts2st$ 的左伴, $lts2st$ 是 $st2lts$ 的右伴. 由于 $st2lts$ 是一个包含函子, **ST** 也称 **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, 同步树与其展开是同构的[7].



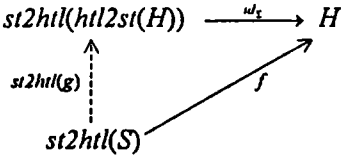


图 4 ST 是 LTS 的共反射子范畴　　图 5 从 ST 到 HTL 的伴随

5.3 范畴 ST 与 HTL 之间关系

踪迹语言摒弃了同步树的不确定选择结构，因而变迁系统与同步树是分支时间模型，踪迹语言是线性时间模型. 将 Hoare 踪迹语言的串作为状态、串的增长作为变迁，可转换为同步树: 任给 $H = (\Sigma, L)$, 定义 $htl2st(H)$ 为同步树 $(L, \varepsilon, \Sigma, \mapsto)$, 其中 $(s, a, s') \forall \mapsto \Leftrightarrow s' = sa$; 任给 Hoare 踪迹语言的射 $\sigma: H' \leftarrow H$, 定义 $htl2st$ 将 σ 映射为同步树的射 $htl2st(\sigma) = (id_Q, \sigma)$, 从而形成函子 $htl2st$: **ST** ← **HTL**.

同步树所有可能出现的变迁序列对应的标号串可作为踪迹语言: 任给 $S = (Q, q_0, \Sigma, \rightarrow)$, 定义 $st2htl(S)$ 为 Hoare 踪迹语言 (Σ, L), 其中 $s \in L \Leftrightarrow \exists q \in Q, q_0 \xrightarrow{s} q$; 任给同步树之间的射 $(\delta, \sigma): S' \leftarrow S$, 定义 $st2htl$ 将 (δ, σ) 映射为 Hoare 踪迹语言的射 $st2htl(\delta, \sigma) = \sigma$, 从而形成函子 $st2htl$: **HTL** ← **ST**.

函子 $st2htl$ 和 $htl2st$ 形成了从 **ST** 到 **HTL** 的伴随, id_Σ 是该伴随的共单元, $st2htl$ 和 $htl2st$ 分别是该伴随的左伴和右伴, 它们的泛性质如图 5 所示: $\forall f: H \leftarrow st2htl(S), \exists ! g: htl2st(H) \leftarrow S, f = id_\Sigma \cdot st2htl(g)$.

利用范畴论不仅给出并发模型之间嵌入关系的形式化解释，而且范畴之间函子的定义实际上给出了不同模型进行转换的算法，具有很好的可实现性.

6 结束语

本文探讨了范畴论应用在形式语义模型研究的一般方法，并以实例说明范畴论应用在并发模型研究的两种途径．正如本文第二节指出，利用范畴论研究并发模型的最终目标是为寻求新机制或指导实际应用．对并发模型间关系的研究从不同的正交坐标出发，形成一个完整的分类体系．这些结论至少为我们带来两点启示：其一，可通过挖掘新的并发模型或扩充新的分类坐标进一步完善并发模型谱系；其二，在此基础上形成一种基于模型变换的逐步求精并发系统开发与验证方法，解决阻碍并发系统实际应用的瓶颈问题．

参考文献：

[1] PIECE B. Basic category theory for computer scientists. London: MIT Press, 1991. 1 ~ 80.

[2] 陆汝钤．计算机语言的形式语义．北京：科学出版社，1992. 527 ~ 699.

[3] WINSKEL G, NIELSEN M. Models for concurrency, in handbook of logic in computer science. ABRAMSKY S, et al eds. London: Oxford University Press, 1995. 1 ~ 148.

[4] BARR M, WELLS C. Category theory for computing science. London: Prentice Hall International, 1995. 1 ~ 270.

[5] SASSONE V, NIELSEN M, WINSKEL G. Models for concurrency: towards a classification. Theoretical Computer Science, 1996, 170: 297 ~ 348.

[6] DEGANO P, GORRIERI R, VIGNA S. On relating some models for concurrency. In: Proceedings of TAPSOFT'93, Lecture Notes in Computer Science. New York: Springer, 1993. 15 ~ 30.

[7] MACLANE S. Categories for the working mathematics. New York: Springer, 1971. 31 ~ 101.

The Categorical Approach to Models for Concurrency

LI Wen-jun *, *ZHOU Xiao-cong*, *LI Shi-xian*

Abstract: Category theory is a general conceptual framework that can be used as the theoretical foundations of software engineering. The way of thinking of category theory is introduced, then the general approach to applying category theory to formal semantic modeling is proposed. The application of category theory to the study of some interleaving models for concurrency and the relationships between these models is also suggested.

Keywords: models for concurrency; category theory

* Department of Computer Science, Zhongshan University, Guangzhou 510275, China