并发进程模型CSP与CCS

肖育东　（郑州大学）

摘　要

并发性是当前最活跃的研究领域之一。已经出现了众多并发进程模型。一个好的并发模型应该概念清晰简单，以便于理解；有丰富的表达能力、演绎能力，以反映并发性的各主要方面；有好的构造性质及动态性质，以支持复杂多样的系统构成；有坚实的数学基础，以支持稳固的发展与深刻的洞察。以此来衡量，CSP模型及CCS模型都属于最成功的模型。本文介绍了它们的基本观点，主要结果，理论背景，模型的意义及一些相关的研究工作。

随着并发分布式计算的社会需求日益增长，实际的分布式问题也日益复杂化。人们在开发各种实际分布式并发计算系统的同时，也提出了一大批有待研究解决的问题，这促使并发性的研究已成为目前计算机科学最活跃的领域之一。研究发现，许多问题实质是共同的，与顺序程序设计方法相比，它先天地要复杂、困难得多。说到底，人们还远未真正了解并发性的实质。因此，需要对并发的基本模型进行广泛深入的研究。一种好的并发进程模型，其概念应尽可能简单而清晰，以便于理解，便于应用；它应当有足够的表达能力，以反映并行性的诸方面；它应当有好的构造性质及动态性质以支持各种复杂的并发系统的构成及形式演绎；它应有尽可能有效的实现机制，以支持实际系统的开发。从这几方面的要求来衡量，CCS模型与CSP模型是非常成功的两个模型。

一、CCS模型

CCS (Calculus of Communicating Systems)[1,2,3] 模型是由Milner提出来的。据他自己讲，当他企图把顺序模型的基本概念（存贮状态上的函数）推广到并发系统而失败时，认识到并发系统的语义理论应建立在通讯(communication)之上。CCS 模型是在一种较弱条件下建立起来的普适并发进程模型，它企图俘获并发性及通讯的一般数学性质。CCS最主要的贡献是关于并发系统构成的等价性研究，其中有代表性的是建立在双模拟基础上的观察等价概念。二个动态系统之间成立着一种不变性，通过建立这种不变性可证明二个系统是等价的，正如为证明一个顺序进程的正确性而找出程序的不变性质一样。站在外部观察者的立场来看，关心

200-207

[7] R. Milner, Interpreting one concurrent calculus in another, in TCS, 75 (1990), 3-13

[8] J. Y. Girard, Toward a geometry of interaction, Contemporary Maths. vol. 92, 1989

[9] P. Degano, et al., Axiomatizing Net Computations and Processes, in LICS 1989. 175-185

[10] J. Meseguer, et al., Petri nets are Monoids, A new algebraic foundation for net theory, in LICS, 1988

[11] C. Brown and D. Gurr, A categorical linear framework for Petri nets, in LICS 1990

[12] 黄林鹏　孙永强，线性逻辑导论，计算机科学91，1

的是二个系统的（外部）行为是否相同，而不关注其内部行为的差异。观察等价的概念以严格的数学形式刻画了系统外在行为的等价性。CCS从简单的事实出发，以严谨、优美的数学形式，建立了并发系统行为的形式理论。Milner最近建议把它改称为进程演算（process calculus）[1]，而不称为代数或逻辑，这是因为代数或逻辑都不足以表示这一理论，得使用另外的数学形式。也不要把CCS看成一种程序设计语言，看成一种语言本身就意味着这一理论是一种不可扩展的形式系统，而作为一种理论是没有这种约束的。归根结底，广义来说，我们所要研究的是断言与作用的关系，或称，系统的说明与其性能的关系。

1. CCS的基本计算

设有名字集$\Delta=\{\alpha, \beta, \gamma, \cdots\}$，其补集$\overline{\Delta}=\{\overline{\alpha}\mid\alpha\in\Delta\}$，标号集$\mathbb{L}=\Delta\cup\overline{\Delta}$，还引入一个沉默的作用$\tau$，并定义$Act=\mathbb{L}\cup\{\tau\}$。$K=\{A\mid A$是事件$\}$表示事件常量集，$X=\{x\mid x$是事件变量$\}$表示事件变量集。用$\varepsilon=\{E_i\}$表示表达式集。

一个类（sort）是子集$L\subseteq Act$，使事件P有类L，P::L指P的所有活动都在L中。

表达式的构造规则如下：

·前缀规则 $\alpha\cdot E$，$\alpha\in Act$：$\alpha\cdot E$的行为是作用α之后E的行为。

·求和规则 $\Sigma_{i\in I}E_i$，I是索引集：ΣE_i的行为是任意E_i之一的行为，若$I=0$，它为0；若$I=2$，它是E_1+E_2。

·合成规则 $E_1\mid E_2$：表示E_1，E_2的并发行为，它们通过互补的作用进行彼此通讯。

·限制规则 $E\setminus L$，$L\subseteq\mathbb{L}$：$E\setminus L$的行为是E的行为，只要每一个作用（或其补）不在L中。

·重标号规则 $E[f]$，f是重标号函数：$E[f]$的行为是E的行为，但其作用由f重新标号。

·递归规则 $fix_j(\{x_i=E_i: i\in I\})$，$j\in I$：它指由$x_i=E_i$，$i\in I$所定义的事件族的第j个分量。一般地，记$\{\widetilde{fix}_j(\widetilde{x}=\widetilde{E}): j\in I\}$表示分量族$\widetilde{fix}(\widetilde{x}=\widetilde{E})$。

下面在标号迁移系统上给出上述规则的迁移语义：

$$\mathrm{Act}\ \frac{}{\alpha\cdot E\xrightarrow{\alpha}E},\qquad \mathrm{Sum}_j\ \frac{E_j\xrightarrow{\alpha}E_j'}{\Sigma_{i\in I}E_i\xrightarrow{\alpha}E_j'}\quad(j\in I),$$

$$\mathrm{Com}_1\ \frac{E\xrightarrow{\alpha}E'}{E\mid F\xrightarrow{\alpha}E'\mid F},\qquad \mathrm{Com}_2\ \frac{F\xrightarrow{\alpha}F'}{E\mid F\xrightarrow{\alpha}E\mid F'}$$

$$\mathrm{Com}_3\ \frac{E\xrightarrow{l}E',\ F\xrightarrow{\overline{l}}F'}{E\mid F\xrightarrow{\tau}E'\mid F'},$$

$$\mathrm{Res}\ \frac{E\xrightarrow{\alpha}E'}{E\setminus L\xrightarrow{\alpha}E'\setminus L},\quad(\alpha,\ \overline{\alpha}\notin L),$$

$$\mathrm{Rel}\ \frac{E\xrightarrow{\alpha}E'}{E[f]\xrightarrow{f(\alpha)}E'[f]},$$

$$\mathrm{Con}\ \frac{P\xrightarrow{\alpha}P'}{A\xrightarrow{\alpha}P'}\ (A\overset{def}{=}P),$$

$$\mathrm{Rec}_j\ \frac{E_j\{\widetilde{fix}(\widetilde{x}=\widetilde{E})/\widetilde{x}\}\xrightarrow{\alpha}E'}{fix_j(\widetilde{x}=\widetilde{E})\xrightarrow{\alpha}E'},$$

可以证明这一规则集是完备的。

2. 三组基本定律

由CCS的定义，可以证明组合子的一些基本运算性质，我们自然地把它们分成三组定律：静态律组、动态律组和扩展律组。

①静态律组：它与合成、限制、重新标号组合子有关。其作用定律可以看成流图上的代数。

·合成律：$P\mid Q=Q\mid P$；$P(Q\mid R)=(P\mid Q)\mid R$；$P\mid 0=P$。

·限制律：$P\setminus L=P$, if $\mathbb{L}(P)\cap(L\cup\overline{L})=\emptyset$；$P\setminus L\setminus K=P\setminus(L\cup K)$；$P[f]\setminus L=P\setminus f^{-1}(L)[f]$；$(P\mid Q)\setminus L=P\setminus L\mid Q\setminus L$, if $\mathbb{L}(P)\cap\overline{\mathbb{L}(Q)}\cap(L\cup\overline{L})=\emptyset$。

·重标号律：$P[id]=P$；$P[f]=P[f']$, if $f^{-1}\mathbb{L}(P)=f'^{-1}\mathbb{L}(P)$；$P(f)[f']=P[f'\circ$

$f]$：$(P|Q)[f]=P[f]|Q[f]$，if $f\upharpoonright(L\cup\overline{L})$为一对一，这里$L=\mathbb{L}(P|Q)$。

②动态律组：它与前缀，求和及常事件有关，这些组合子的出现仅在作用发生之前，因此称之为动态的。它们可以看成是迁移图上的代数。我们把常事件视为一元组合子，递归组合子是有选择地使用常事件，也是一种动态组合子。

·**单子律**：$P+Q=Q+P$；$P+(Q+R)=(P+Q)+R$；$P+P=P$；$P+0=P$。

·**τ律**：$\alpha\cdot\tau\cdot P=\alpha\cdot P$；

$P+\tau\cdot P=\tau\cdot P$；

$\alpha\cdot(P+\tau\cdot Q)+\alpha\cdot Q=\alpha\cdot(P+\tau\cdot Q)$。

③扩展律组：它把两种组合子联系起来。

·令$P=(P_1|\cdots|P_n)\setminus L$，$n\geqslant 1$，则

$$P=\Sigma\{\alpha\cdot(P_1|\cdots|P_i'|\cdots|P_n)\setminus L:P_i\xrightarrow{\alpha}P_i',\ \alpha\notin L\cup\overline{L}\}+\Sigma\{\tau\cdot(P_1|\cdots|P_i'|\cdots|P_j'|\cdots|P_n)\setminus L:P_i\xrightarrow{\alpha}P_i',\ P_j\xrightarrow{\overline{\alpha}}P_j',\ i<j\}$$

·对于扩展律的特殊情况，我们得到

$$(\alpha\cdot Q)\setminus L=\begin{cases}0, & \text{if } \alpha\in L\cup\overline{L},\\ \alpha\cdot Q\setminus L, & \text{此外};\end{cases}$$

$(\alpha\cdot Q)[f]=f(\alpha)\cdot Q[f]$；

$(Q+R)\setminus L=Q\setminus L+R/L$；

$(Q+R)[f]=Q[f]+R[f]$。

3．等价性

在CCS理论中，对于模型等价问题的研究占有中心地位，也是CCS最富成果的研究。首先引入几个基本记号：若$t\in Act^*$，则$\hat{t}\in\mathbb{L}^*$，即$\hat{t}$是从t中删去所有τ的出现所得的作用序列。在刻画事件E在作用序列的作用下迁移到E′时，使用：

·E与E′，表示t中τ的作用精确的出现；

·$E\xrightarrow{t}E'$，表示$E\left(\xrightarrow{\tau}\right)^*\xrightarrow{\alpha_1}\left(\xrightarrow{\tau}\right)^*\xrightarrow{\alpha_2}\cdots\cdots\left(\xrightarrow{\tau}\right)^*\xrightarrow{\alpha_n}\left(\xrightarrow{\tau}\right)^*E'$，其中$t=\alpha_1\cdots\alpha_n\in Act^*$；

·$E\xrightarrow{\hat{t}}E'$，表示没有τ作用的出现。

等价的概念是建立在双模拟的基础之上。下面是相关的几组定义及主要性质。

①强双模拟及强等价

定义1 任二元关系$s\in\mathcal{P}\times\mathcal{P}$是强双模拟，若$(P,Q)\in S$，隐含着对于所有$\alpha\in Act$。i）当$P\xrightarrow{\alpha}P'$，则对某$Q'$，$Q\xrightarrow{\alpha}Q'$及$(P',Q')\in S$；ii）当$Q\xrightarrow{\alpha}Q'$，则对某$P'$，$P\xrightarrow{\alpha}P'$，及$(P',Q')\in S$。

把这一定义中的“隐含着”改为“当且仅当”，就是强等价的定义。强等价关系记为“$\sim$”。

强等价的基本性质有：

·$\sim$是最大强双模拟：$\sim=\cup\{s:s$是强双模拟$\}$。

·$\sim$是等价关系。

·除了对于τ律之外，所有的组合子运算都保留强等价性。

②（弱）双模拟及观察等价

定义2 二元关系$s\in\mathcal{P}\times\mathcal{P}$是(弱)双模拟，若$(P,Q)\in S$隐含着对于所有$\alpha\in Act$，i）当$P\xrightarrow{\alpha}P'$，则对某$Q'$，$Q\overset{\hat{\alpha}}{\Rightarrow}Q'$，及$(P',Q')\in S$；ii）当$Q\xrightarrow{\alpha}Q'$，则对某$P'$，$P\overset{\hat{\alpha}}{\Rightarrow}P'$，及$(P',Q')\in S$。

把这一定义中“隐含着”改为“当且仅当”即为观察等价的定义。观察等价记为“$\approx$”。

观察等价的基本性质有：

·$\approx$是最大（弱）双模拟，即$\approx=\cup\{s:s$是(弱)双模拟$\}$。

·$\approx$是等价关系。

·Id，s_1s_2，s_i^{-1}，$\cup s_i$保留双模拟。

·$P\approx\tau\cdot P$，这是双模拟力量的来源。但$\approx$对于+不保留，如$b\cdot 0\approx\tau\cdot b\cdot 0$，但$a\cdot 0+b\cdot 0\approx a\cdot 0+\tau\cdot b\cdot 0$

·静态组合子保留双模拟：若$P\approx Q$，则$P|R\approx Q|R$，$P\setminus L\approx Q\setminus L$，$P[f]\approx Q[f]$。

③观察等同 由于$\approx$不是充分可代换的，我们考虑一种充分可代换关系“$=$”，它应是$\approx$中的最大等同关系。

定义3 P，Q是观察等同的，记为$P=Q$，

若对于所有$\alpha\in Act$，i）当$P\xrightarrow{\alpha}P'$，则对某Q'，$Q\overset{\alpha}{\Rightarrow}Q'$，及$P'\approx Q'$；ii）当$Q\xrightarrow{\alpha}Q'$，则对某$P'$，$P\overset{\alpha}{\Rightarrow}P'$，及$P'\approx Q'$。

这里仅指P及Q的第一次作用α，此后仅要求$P'\approx Q'$，而不是$P'=Q'$。观察等同的基本性质有：

• 它对所有组合子都保留：若$P=Q$，则$\alpha\cdot P=\alpha\cdot Q$，$P+R=Q+R$，$P|R=Q|R$，$P\setminus L=Q\setminus L$，$P[f]=Q[f]$。

• 它是等价关系。

• $P\sim Q$隐含$P=Q$，$P=Q$隐含$P\approx Q$。

• 若P，Q为无τ派生的，$P\approx Q$，则，$P=Q$。

• $P\approx Q$当且仅当$P=Q\vee P=\tau\cdot Q\vee\tau\cdot P=Q$。

二、CSP模型

CSP（Communicating Sequential Processe）[4, 6, 8]模型是C.A.R.Hoare创立的，比CCS较早，但这二个模型的主要方面的工作是平行地发展的。CSP模型的目的是描述一种在计算机应用的广泛领域中适用的最简单的数学理论。从售货机到进程控制，从离散事件模拟到共享资源操作系统；并能在各种计算机结构上有效地执行；它还能在说明、设计、执行及验证等方面给程序员以支援。CSP模型以进程中事件的顺序执行及进程间通讯为出发点，研究计算的一般模式。CSP的主要贡献是它把计算机所涉及的各种计算形式及其性质建立在一套严密的形式系统之上，这套形式系统的数学语义是迹及失效语义。Dijkstra在Hoare的著作[4]的前言中评价这一成就道："墨迹未干就已成为经典"。

1．进程与迹

一个对象从事**事件**x，然后，其行为由P来描述，写为$x\to P$，P称为**进程**，P的字母表为αP，x是P的**前缀**。一个从前缀开始的进程描述符称为**有哨的**。若F(x)是有哨表达式，则方程$X=F(X)$递归定义其解表达式$\mu X:A\cdot F(X)$，其中A是字母表。

一个进程行为的**迹**是记录从其时刻开始所从事事件的有限符号序列。如$\langle x,y\rangle$，$\langle\ \rangle$分别表示二个顺次事件及空事件的迹。迹上的运算主要有连接$u\frown t$，限制$u\restriction A$，迹头$(\langle x\rangle\frown s)_0=x$，迹尾$(\langle x\rangle\frown s)'=s$，迹的序关系$s\leqslant t=(\exists u.s\frown u=t)$及迹的长度$\#u$。P/s的行为是P从事迹s之后的行为。

特殊进程$RUN_A=x:A\to RUN_A$，可以从事A上的任意事件，$traces(RUN_A)=\{u\mid u\in A^*\}$；进程$STOP_A$什么也不做，$traces(STOP_A)=\{\langle\rangle\}$。其中$A^*$是A上的所有有限迹集。

当一个进程由于不同的起始事件而导致不同的后继时，用"|"表记，如$R=(x\to P)|(y\to Q)$，这里并不把"|"看成一种运算，这类选择一般地可表为$x:B\to P(x)$的形式。

2．进程的运算

我们仅讨论几个主要的基本运算。重点区分相似运算之间的差异。

①并行合成 有$P\|Q$及$P\|\!|Q$二种形式。同步式并行合成$P\|Q$，对于P，Q共有的事件集$(\alpha P\cap\alpha Q)$，要求P与Q的行为严格同步锁定地发生。仅出现在P或Q中的事件则可以独立地发生。因此，除了一些基本运算定律外，还有一些严格地依赖于符号表的运算性质。

另一种是不确定并行合成$P\|\!|Q$，$\alpha P=\alpha Q$，P与Q各自完全独立地作用，而不管其字母表是否相同。系统的每一个作用是P，Q之一的一个作用。若P，Q的作用恰巧同时发生，则从这二个作用中不确定地选择一个作为系统的作用。

②选择 CCS中有两种选择算子，当环境不能控制进程P，Q之间的选择时，记为$(P\sqcap Q)$，它是"非确定OR"。产生这类选择的原因可能是环境不能影响，也不能观察到这种选择作用，或者由于忽略了一些决定选择的因素，对于观察者，这种选择是由系统内部不确定地做出的（如：售货机找回零钱的面值组合等）。若环境可以控制这种选择，

则记为（P□Q），这指只要能控制第一个事件的选择，就可以决定选择的是P还是Q。若初始时不可能选择P的第一个作用，则选择Q；若P，Q的第一个作用都不可能选择，则它们之间的选择是不确定的。这二种选择算子的区别可以从下述看出：

$c\to P\,\square\, d\to Q = c\to P\,|\,d\to Q$，if $c\neq d$

而 $c\to P\,\square\, c\to Q = c\to P\sqcap c\to Q$

③**删除** 一般地，一个进程的字母表恰包含它所涉及的事件。但在描述一个系统的内部行为时通常要考虑表示其内部作用的事件。在构造这一系统后，我们希望不再看到其组分结构及内部作用，因为这些内部作用在需要时会自动出现，而不必为外部所察觉。（P\C）是一进程，除了C中每一个事件被删除之外，其行为像P，$\alpha(P\setminus C)=(\alpha P)-C$。如$(a\to c\to P\|c\to b\to Q)\setminus\{c\}$可化为$a\to\mu X.(a\to b\to X|b\to a\to X)$。

3. 通讯

通讯是在通道上传送消息的一类事件。它是一种可观察的行为，是系统部件进行物理连接的一种手段。把输出消息看成一种前缀算符，把输入看成一种特殊的选择，则$R=C!V\to P\|c?x\to Q(x)=c!v\to(P\|Q(v))$。为把通讯作为内部事件隐藏起来，可以使用删除：$R\setminus C=(P\|Q(v))\setminus C$，其中$C=\{c,v\mid v\in\alpha C\}$。若$\alpha P\subseteq\alpha Q$，则$(P\|Q)$中P服务于Q，是从进程，Q是主进程。二者的通讯被删除，可写为$(P/\!/Q)$，且$(P/\!/Q)=(P\|Q)\setminus\alpha P$，于是$\alpha(P/\!/Q)=\alpha Q-\alpha P$。一个从进程可以有多个通道，可以通过不同的通道被多个进程共享使用，形成共享资源的各种模式。使用主从进程的递归构造，可以形成各类动态数据结构及程序结构。

4. 顺序进程

引入一个特殊事件"✓"（正常终止），它是顺序进程的最后一个事件，表示进程正常结束。一个特殊进程SKIP，除了正常终止外，什么也不作。（它不同于STOP，导致STOP的原因可能是死锁或其它设计错误）。顺序进程P，Q的顺序合成（P；Q）的迹是P的迹。若P的迹由"✓"结束，则"✓"被Q的迹置换。由顺序结构可以派生出顺序程序设计中的各种控制概念，包括循环，条件分支，赋值，中断，断点，重新启动等。

5. CSP的数学语义

失效模型上的CSP的语义，是非确定性进程的数学基础。下面先定义几个概念。

定义4 可以作任何事情的进程称为CHAOS。

定义5 进程P的发散（divergence）是该进程的任何迹，在该迹之后，进程行为混乱，即 divergence $(P)=\{s\mid s\in traces(P)\wedge P/s=CHAOS_{\alpha P}\}$

定义6 进程P的拒绝（refusal）是一集合，该集合不包含P初始所从事的事件。

$refusals(P)=X:X\cap P^0=\{\ \}$

定义7 进程的失效（failure）是关系(S，X)，有$failures(P)=\{(S,X)\mid s\in traces(P)\wedge X\in refusals(P/s)\}$，即若$(S,X)\in failures(P)$，指P可以从事由S记录的事件序列，然后拒绝作X中的任何事件。显然，迹与拒绝可以由失效来定义：

$traces(P)=\{S\mid \exists X.(S,X)\in failures(P)\}$。

$refusals(P)=\{X\mid(\langle\rangle,X)\in failures(P)\}$。

一个进程可以由一个三元组〈字母表，失效，发散〉唯一地确定；反之，一个满足适当条件的三元组唯一地确定一个进程。

定义8 失效模型是三元组(A，F，D)，其中A是字母表，F是A^*与$\mathbb{P}(A)$之间的关系集，D是A^*子集，且满足下述条件：

① $(\langle\rangle,\{\ \})\in F$；

② $(s^\frown t,X)\in F\Rightarrow(s,\{\ \})\in F$；

③ $(s,Y)\in F\wedge X\subseteq Y\Rightarrow(s,X)\in F$；

④ $(s,X)\in F\wedge x\in A\Rightarrow(s,X\cup\{x\})\in F\vee(s^\frown\langle x\rangle,\{\ \})\in F$；

⑤ $D\subseteq Domain(F)$；

⑥ $s\in D\wedge t\in A^{*}\Rightarrow s^{\frown}t\in D$;

⑦ $s\in D\wedge X\subseteq A\Rightarrow(s,X)\in F$。

由于可以给出进程在失效模型上的语义。如对于“$\|$”，有

$A[\![P\|Q]\!]_{\varepsilon}=A[\![P]\!]_{\varepsilon}\cup A[\![Q]\!]_{\varepsilon}$

$D[\![P\|Q]\!]_{\varepsilon}=\{st\mid s\in(D[\![P]\!]_{\varepsilon}\cap tracesF[\![Q]\!]_{\varepsilon})\cup(D[\![Q]\!]_{\varepsilon}\cap tracesF[\![P]\!]_{\varepsilon})\wedge t\in(A[\![P]\!]_{\varepsilon}\cup A[\![Q]\!]_{\varepsilon})^{*}\}$

$F[\![P\|Q]\!]_{\varepsilon}=\{(s,X\cup Y)\mid(s,X)\in F[\![P]\!]_{\varepsilon}\wedge(s,Y)\in F[\![Q]\!]_{\varepsilon}\}\cup\{(s,X)\mid s\in D[\![P\|Q]\!]_{\varepsilon}\}$

可以写出两个特殊进程的语义：

$A[\![CHAOS_X]\!]=X$,

$D[\![CHAOS_X]\!]=X^{*}\times\mathrm{P}X$,

$F[\![CHAOS_X]\!]=X^{*}$

它是X上的最大进程，它在任何时候可以作任何事件，可以拒绝任何事件。

$A[\![STOP_X]\!]=X$

$D[\![STOP_X]\!]=\{\langle\rangle\}\times\mathrm{P}X$

$F[\![STOP_X]\!]=\{\}$

它什么也不作，拒绝作任何事件，但不发散。

根据不动点理论研究失效模型的递归情形，先定义偏序：$(A,F_1,D_1)\sqsubseteq(A,F_2,D_2)\equiv(F_2\subseteq F_1\wedge D_2\subseteq D_1)$，$P\sqsubseteq Q$指在较少失效及较少发散的意义下Q等于或好于P，Q较可预测，较可控制。在这一偏序上构成CPO，其中由于$CHAOS\sqsubseteq P$，CHAOS是CPO的底元。

$$\bigsqcup_{n\geqslant 0}(A,F_n,D_n)=(A,\bigcap_{n\geqslant 0}F_n,\bigcap_{n\geqslant 0}D_n),$$

只要$(\forall n\geqslant 0.\ F_{n+1}\subseteq F_n\wedge D_{n+1}\subseteq D_n)$，且可证明，除“/”之外，对于所有算子都是连续的。使用不动点性质，可写：

$$\mu X:A.\ F(X)=\bigsqcup_{i\geqslant 0}F^{i}(CHAOS_A)$$

6. 确定性与非确定性

现在来看一看，在CSP模型下非确定性是如何产生的，它意味着什么？

一进程若决不拒绝它所从事的事件，该进程是确定的。可写

$P\text{是确定的}\equiv\forall s:traces(P).$

$(X\in refusals(P/s)\equiv X\cap(P/s)^{0}=\{\})$

非确定进程不具有这一性质，它有时可以从事一事件，而另一时又拒绝该事件。导致进程不确定的来源有：①不确定选择算子“$\sqcap$”，这种不确定性往往不是来自问题本身，而是来自运算的结果。②非确定并行算子“$|||$”，由于$(P|||Q)$中P与Q迹的任意交互，我们无法预测及控制s之后$(P|||Q)/s$的迹将以什么次序发生。③删除算子“$\backslash$”。这是由确定导致不确定的根本来源。当被删除的事件不可见地发生时，不能确定其中哪一个出现。删除一个进程的前缀导致进程变为无哨的，它把进程的第一事件变成为不确定的选择。无哨的递归进程将不存在唯一解。当删除进程中的一个无限序列时，将导致不可见的无限循环。如$(\mu X:A\cdot(c\to X))\backslash\{c\}=CHAOS_{A-\{c\}}$，这种现象就是发散。发散的另一来源是递归方程的不良定义，如$\mu X\cdot X$，其初始条件不可见（无哨）。发散可归纳为

$divergences(P\backslash C)=\{s\mid(\alpha P-C)^{\frown}t\mid t\in(\alpha P-C)^{*}\wedge(s\in divergences(P)\vee\forall n.\ \exists u\in C^{*}.\#u>n\wedge s^{\frown}u\in traces(P))\}$

7. 再谈失效模型

关于失效模型，有两点值得提及：①模型中使用拒绝集，而不使用其对偶——接受集来定义失效。这是因为拒绝集有较好的数学性质，它对于（除“/”之外）所有算子都是连续的，这给建立CPO提供好的基础。但使用接受集或接受/拒绝集方面也有一些好的工作[8,10]。②在发散的定义中，使用CHAOS来表示发散，这不是一个理想的定义，因为CHAOS对任何外部输入一定做出（接受或拒绝的）响应，但发散是一种不可见的无限循环，它在任何有限时间内都不响应外部输入。因此CHAOS不符合直观。CHAOS还引起信息的丢失，因为实际的发散行为并不象CHAOS那样混乱。另一种处理是把发散与死锁不做区别，使用STOP来表示发散。可惜STOP没有好的代数性质。相对照，对

于删除，CHAOS是连续算子。发散的处理问题之所以重要，是因为当建立失效语义与操作语义的关系以及将失效模型公理化时，是一个绕不过的问题。一些进一步的研究已建立了更为复杂的失效模型，企图克服上述困难。[7,8,10]

参考文献

[1] Milner, R., Communication and Concurrency, Prentice Hall, 1989

[2] Milner, R., A Calculus of Communicating Systems, LNCS, 92, Springer-Verlag, 1980

[3] Milner, R., Lectures on A Calculus for Communicating Systems, in Control Flow and Data Flow, pp205—228, Springer-Verlag, 1985

[4] Hoare, C, A, R., Communicating Sequential Processes, Prentice Hall, 1985

[5] Hoare, C, A, R., Communicating Sequential Processes, CACM, Vol.21, pp666—677, 1978

[6] Brooks, S, D., Hoare, C, A, R., and Roscoe, A, W., A Theory of Communicating Sequential Processes, JACM, Vol.30, pp560—599, 1984

[7] Brooks, S, D., and Roscoe, A W., An Improved Failures Model for Communicating Processes, LNCS, 197, Springer-Verlag, 1984, pp281—305

[8] Golson, W. G., Denotational Models Based on Synchronously Communicating Processes; Refusal, Acceptance, Safety, LNCS, 197, Springer—Verlag, 1984, pp373—388

[9] Nicala, R., Two Complete Sets of Axioms for a Theory of Communicating Sequential Processes, Proc. International Conf. on Foundations of Computation Theory, LNCS, 1983, Springer-Verlag

[10] Kennaway, J., A Theory of Nondeterminism, LNCS 85 pp338—350, 1980, Springer-Verlag

（接第43页）

对象管理器应包括一个描述数据库中全部对象类的完整数据词典。该"词典"为浏览器和查询语言所用，也为需要访问对象类定义信息的其它工具所用。

版本管理、配置控制管理和变更管理 用了对象管理器，一个给定对象的多种版本就可以同时存在。配置是对象版本的集合。对象管理器负责对系统的各个用户进行追踪，用户以不同的配置进行工作。变更管理软件负责通知用户何时因其它用户的更新而使自己当前配置可能过时。

多用户环境 可在分布式工作站的环境中支持多个用户。例如，一个小组可能有一个中心数据库。它包含目标系统模型的"获准"状态。任何个人均可从这个中心数据库中提取子集，并将其移入某工作站的局部数据库。于是，用户便可在局部数据库上工作，在修改完成之后，要检查更新并送返中心数据库。这些登记和销帐特征实际上是前面所述的变更管理系统的一部分。

标准DBMS的特征 这些特征应包括备份和恢复工具、事务管理功能、数据和字符文件间来回映射的工具（以便数据出入外部系统）、数据库管理员定义数据的物理组织的工具，以及产生报告的工具等等。

本文的大量篇幅出自笔者和"国际软件系统公司"的同事们正在进行的研究。特别值得一提的是，Terry Welch博士已指导开发了一个非常高级的原型化与设计的语言，Don Hartman博士正负责开发一个对象管理器和图形化的用户界面管理器以作为集成工具的支柱。这些开发都是在UNIX工作站上(SUN和APOLLO)做的。笔者尤衷感谢Terry Welch和Don Hartman所提供的本文内容。（参考文献略）

〔**龚中凡 赵磊**译自《Modern Software Engineering, Foundations and Current Perspectives》, Peter A. Ng 和 Raymond T. Yeh 主编, Von Nostrand Reinhold出版社 1990年版〕