S-网的进程表达式及求取方法研究*)

段 华[2,3] 曾庆田[1,3]

(中国科学院计算技术研究所 北京 100080)[1] (上海交通大学数学系 上海 200240)[2]

(山东科技大学信息科学与工程学院 青岛 266510)[3]

摘 要 Petri网的进程是用于系统行为和状态描述的有效工具,但是通常很难给出结构复杂Petri网进程的全部描述。本文考察结构简单的S-网的进程行为,给出各种类型的S-网的进程表达式的描述方法,为结构复杂Petri网系统的进程描述提供借鉴和帮助。

关键词 Petri网,S-网,进程,进程表达式

Methods for Obtaining the Process Expressions of S-Nets

DUAN Hua[2,3] ZENG Qing-Tian[1,3]

(Institute of Computing Technology, Chinese Academy of Sciences, Beijing100080)[1]

(Department of Mathmetics, Shanghai Jiaotong Univrsity, Shanghai 200240)[2]

(Department of Computer Science and Technology, Shandong University of Science and Technology, Qingdao 266510)[3]

Abstract Process is most useful for property analysis among all the analysis tools of Petri net, however it is usually difficult to present all the processes of a structure-complex Petri net. The process characteristics of S-Nets, a class of structure-simple Petri net, are analyzed with details in this paper. The process expressions of all kinds of S-Nets are presented, which can be used for references of process analysis for structure-complex Petri net.

Keywords Petri net, S-Net, Process, Process expression

1 引言

进程是Petri网众多的分析方法中对系统行为描述和分析的最有力工具,因为进程将状态和变迁并重,把系统中发生的变化和引起的状态改变如实记录下来,它可以很清楚地反映出网系统运行中的变迁之间的顺序、并发、同步等现象[1~4]。然而,一个进程只能反映Petri网的一种可能运行情况。一个Petri网往往有许多(可能无限多个)进程,可能无法一一列举。这就为利用进程分析Petri网的行为带来了许多困难。为此,国内外学者先后提出了P/R网[5]、进程网系统[6]、进程表达式[7~9]等理论和方法用于描述Petri网的进程行为。对于结构复杂的Petri网,需要首先建立其特征可达树求取基本进程段集,然后构造其进程网系统,将其进程描述问题转换成进程网系统的语言描述问题[6,9]。文[9]中给出了通过分解的方法描述进程网系统的语言行为的方法,从而给出原先Petri网的进程表达式的求取方法。可见,结构复杂的Petri网进程描述是非常复杂的,目前所给出的方法和结果也仅限于进程行为的约束语义描述,所以仍然无法直接利用进程表达式对Petri网的状态和行为进行分析和判定。

S-网是一类结构简单具有良好性质的Petri网[10,11]。本文考察S-网的进程行为,给出各种类型的S-网的进程行为描述方法。首先分析S-网的进程行为,然后给出了各类S-网的进程表达式的求取方法。通过对结构简单S-网的进程行为分析,可以为结构复杂Petri网的进程行为分析提供方法和依据。

2 Petri网进程的基本概念

本文假设读者对Petri网及其进程的概念有所了解,这里只对与本文讨论有关的基本概念、术语和记号做简述或约定。为使定义尽量简洁,我们只讨论P/T网的进程,假定$K=\omega$和$W=1$。

定义1[1] 设$N=(B,E;G)$为一个网,如果:

(1) $\forall b\in B:|{}^{\cdot}b|\leqslant 1\wedge|b^{\cdot}|\leqslant 1$;

(2) $\forall x,y\in B\cup E:(x,y)\in G^{+}\rightarrow(y,x)\notin G^{+}$,则称$N$为一个出现网,其中$G^{+}$表示流关系$G$的传递闭包。

定义2[1] 设$N_1=(S,T;F)$为一个网,$N_2=(B,E;G)$为一个出现网,若映射$\varphi:B\cup E\rightarrow S\cup T$满足:

(1) $\varphi(B)\subseteq S;\varphi(E)\subseteq T$;

(2) $\forall x,y\in B\cup E:(x,y)\in G\rightarrow(\varphi(x),\varphi(y))\in F$;

(3) $\forall e\in E:\varphi({}^{\cdot}e)={}^{\cdot}\varphi(e),\varphi(e^{\cdot})=\varphi(e)^{\cdot}$;

则称φ定义了N_2到N_1的一个映射,记为$\varphi:N_2\rightarrow N_1$。

定义3[1] 设$\Sigma=(N_1,M_0)=(S,T;F,M_0)$为一个Petri网,$N=(B,E;G)$为一个出现网,如果$\varphi:N\rightarrow N_1$满足条件

(1) $\forall b_1,b_2\in B:(b_1\neq b_2)$ 若$\varphi(b_1)=\varphi(b_2)$则:${}^{\cdot}b_1\neq{}^{\cdot}b_2$,$b_1^{\cdot}\neq b_2^{\cdot}$;

(2) $\forall s\in S:|\{b|\varphi(b)=s\wedge{}^{\cdot}b=\varphi\}|\leqslant M_0(s)$;

则称(N,φ)为Σ的一个进程。

为了讨论问题的方便,我们给出Petri网的满进程的概念。所谓满进程是指其每个S-切都对应着Petri网的一个可达标识的那一类进程。

定义4[7] 设φ为出现网$N=(B,E;G)$到Petri网$\Sigma=(S,T;F,M_0)$的一个网映射,如果:

(1) $\forall b_1,b_2\in B\quad(b_1\neq b_2)$:

$\varphi(b_1)=\varphi(b_2)\rightarrow({}^{\cdot}b_1\neq{}^{\cdot}b_2\vee{}^{\cdot}b_1={}^{\cdot}b_2=\phi)$

$\wedge(b_1^{\cdot}\neq b_2^{\cdot}\vee b_1^{\cdot}=b_2^{\cdot}=\phi)$;

*)本课题得到国家自然科学基金(60173053和60274063),山东省"泰山学者"专项基金和山东省自然科学基金(Y2002G09)的资助。

(2)$|\{b|\varphi(b)=s\wedge {}^{\cdot}b=\phi\}|=M_0(s)$;

则称 $P=(N,\varphi)$ 为 Σ 的一个满进程。

定义 5[6] 设 $P=(N,\varphi)$ 为 Σ 的一个满进程，其中 $N=(B,E;G)$，u_1,u_2 为 N 的两个 S-切，$u_1\leqslant u_2$。记：

(1)$B_1=\{x\in B\}|\exists b_1\in u_1,b_2\in u_2:$

$(b_1,x)\in G^*\wedge(x,b_2)\in G^*\}$;

(2)$E_1=\{y\in E|\exists b_1\in u_1,b_2\in u_2;$

$(b_1,y)\in G^+\wedge(y,b_2)\in G^+\}$;

(3)$G_1=G\cap\{(B_1\times E_1)\cap(E_1\times B)\}$;

令 $N_1=(B_1,E_1;G_1)$，$\varphi_1:N_1\to\Sigma$ 满足：$\forall x\in B_1\cup E_1:\varphi_1(x)=\varphi(x)$，则称$(N,\varphi_1)$为进程 P 的(界于 u_1 和 u_2 之间的)一段，也称作 Σ 的一个进程段，记为$(N[u_1,u_2],\varphi)$。

定义 6[6] 设 $P=(N,\varphi)$ 为 Σ 的一个满进程，$P_1=(N[u_1,u_2],\varphi)$是 Σ 的一个进程段，如果 $N[u_1,u_2]$中的任意两个 s 切 u_i 和 $u_j(i,j\neq1,2)$有：

$$u_i\neq u_j\to(\varphi(u_i)\neq\varphi(u_j))\wedge(\varphi(u_i)\not<\varphi(u_j))\wedge(\varphi(u_i)\not>\varphi(u_j))$$

则称 $P_1=(N[u_1,u_2],\varphi)$是 Σ 的一个基本进程段。

定义 7[9] 设 $\Sigma=(S,T;F,M_0)$为一个 Petri 网，$BP(\Sigma)$为 Σ 的基本进程段集，$Exp(P(\Sigma))$是以 $BP(\Sigma)$中的元素为字母表的一个表达式，该表达式所描述的集合为 $CRE(P\Sigma))$。如果 Σ 的每个满进程都是集合：$Pref\{Exp(P(\Sigma))\}=\bigcup_{P\in CRE(P(\Sigma))}Pref(P)$的一个元素，则称 $Exp(P(\Sigma))$为 Σ 的进程表达式。

3 S-网及其进程特性分析

定义 8[1] $\Sigma=(S,T;F,M_0)$为一个 Petri 网，Σ 称为S-网当且仅当 $\forall t\in T:|{}^{\cdot}t|\leqslant1$ 并且$|t^{\cdot}|\leqslant1$。

定义 9[10] $\Sigma=(S,T;F,M_0)$为一个 S-网，$\forall t\in T$:若$|{}^{\cdot}t|=0$，则称变迁 t 为 Σ 的源变迁；若$|t^{\cdot}|=0$，则称变迁 t 为 Σ 的汇变迁。

容易证明，S-网的进程满足下面的一些命题。

命题 1 设 $P=(N,\varphi)$为S-网 Σ 的一个满进程，其中 $N=(B,E;G)$，则任意 $e\in E$，$|{}^{\cdot}e|\leqslant1$ 且$|e^{\cdot}|\leqslant1$。

命题 2 设 S-网 $\Sigma=(S,T;F,M_0)$的基本进程段集为 $BP(\Sigma)$，则任意 $P_i\in BP(\Sigma)$，P_i 不是减进程段。

文[6]给出了 Petri 网的进程网系统的概念，关于进程网系统的定义以及相关的讨论可参见文[6]。

命题 3 设 $\Sigma_P=(S_P,T_P;F_P,M_{0P})$为 S-网 $\Sigma=(S,T;F,M_0)$的进程网系统，则 Σ_P 也是一个 S-网。

易知：任意给定一个含有汇变迁的 S-网 Σ 在保持变迁引发序列不变的条件下，均可转变成一个不含汇变迁的 S-网。具体方法：给每个汇变迁增加一个输出库所。新增加的输出库所虽然改变了 S-网进程中的状态，但是得到所有的进程后去掉所有的增加的库所即可得到原 S-网的进程。因此，在本文中，我们假定所讨论的 S-网均不含有汇变迁，即 S-网 Σ 中任意变迁 t 满足：$|{}^{\cdot}t|\leqslant1$ 且$|t^{\cdot}|=1$。

4 S-网的进程表达式

根据 S-网中是否含有源变迁以及是否含有初始标识，可以将 S-网分为四类：

(1)既不含源变迁也不含初始标识；

(2)不含源变迁但含初始标识；

(3)含源变迁但不含初始标识；

(4)既含源变迁又含初始标识。

其中，第(1)类 S-网中不可能包含可引发的变迁，所以这类网不会有进程发生，故在本文中没有讨论的意义。易知，不含源变迁但含初始标识的 S-网就是标识 S-图。下面我们分别讨论后三类 S-网的进程特性，在不会引起混淆的情况下，我们直接称第(2)类 S-网为标识 S-图，第(3)类 S-网为不含初始标识的 S-网，第(4)类 S-网为含初始标识的 S-网。

4.1 标识 S-图的进程表达式

引理 1[1] 标识 S-图是有界的。

定理 1 标识 S-图 $\Sigma=(S,T;F,M_0)$的进程表达式 $Exp(\Sigma)$是一个正规表达式。

证明：根据文[7]知，有界 Petri 网的进程表达式是一个正规表达式。

标识 S-图是有界的，根据可达图可以求得其基本进程段集，具体方法可参见文[7]。

例 1 图 1 给出了一个标识 S-图 Σ_1，可以求得 Σ_1 的四个基本进程段，如图 2 所示，其中 P_1 是一个起始程段，P_2 是一个不变进程段，P_3 和 P_4 是两个终结进程段。可以给出 Σ_1 的进程表达式为：

$Exp(\Sigma_1)=P_1(P_2)^*(P_3+P_4)$。

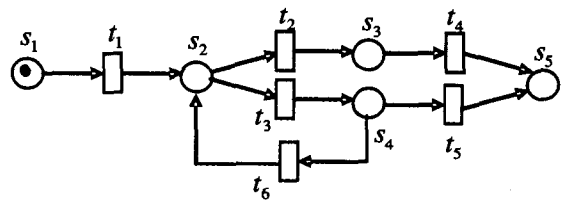


图 1 标识 S-图 Σ_1

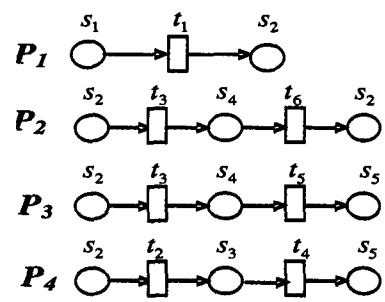


图 2 Σ_1 的四个基本进程段

4.2 不含初始标识的 S-网的进程表达式

引理 2 设 S-网 $\Sigma=(S,T;F,M_0)$中，$\forall s\in S:M_0(s)=0$，有且只有一个 $t'\in T$ 为 Σ 的源变迁，则 Σ 只有一个初始基本进程段。

证明：显然。

引理 3 设 S-网 $\Sigma=(S,T;F,M_0)$中，$\forall s\in S:M_0(s)=0$，有且只有一个 $t'\in T$ 为 Σ 的源变迁，则存在正规表达式使得 Σ 的进程表达式 $Exp(\Sigma)=(RE)^\alpha$。

证明：设 Σ 的基本进程段集为 $BP(\Sigma)$，记 $BP(\Sigma)$中的任意基本进程段为 $P_i=(N[u_{1i},u_{2i}],\varphi)$，这里 $1\leqslant i\leqslant|BP(\Sigma)|$，$u_{1i}$和 u_{2i}分别为 P_i 两端的 S-切。首先构造 Σ 的进程网系统 $\Sigma_p=(S_P,T_P;F_P,M_{0P})$，其中

(1)$S_P=\{u_{1i}|(N[u_{1i},u_{2i}],\varphi)\in BP(\Sigma)\}$;

$\cup\{u_{2i}|(N[u_{1i},u_{2i}],\varphi)\in BP(\Sigma)\}$

(2) $T_P=BP(\Sigma)$;

(3) $F_P=\{(u_{1i},P_i)\mid P_i\in BP(\Sigma)\}\cup\{(P_i,u_{2i})\mid P_i\in BP(\Sigma)\}$,其中 $P_1=(N[u_{11},u_{21}],\varphi)$为 Σ 的初始基进,因 $u_{11}=\phi$,故$(u_{11},P_1)\notin F_p$;

(4) $\forall S\in S_P: M_{0P}(s)=0$。

根据进程网系统的意义知[6],Σ的进程表达式与进程网系统Σ_P 的语言表达式之间存在一一对应关系。可以证明 E_p 是一个只含有一个源变迁不含初始标识的 S-网,由文[10]知 Σ_p 的语言表达式可表示为一个正规表达式的α-闭包形式,即存在正规表达式 RE 使得 Σ的进程表达式 $Exp(\Sigma)=(RE)^{\alpha}$。

定理 2 设 S-网 $\Sigma=(S,T;F,M_0)$中,$\forall s\in S: M_0(s)=0$且存在 $t_1,\cdots,t_k(k\geqslant 2)$为 Σ 的 k 个源变迁,则存在正规表达式 $RE_1,\cdots,RE_k$ 使得Σ 的进程表达式 $Exp(\Sigma)=(RE_1)^{\alpha}\parallel\cdots\parallel(RE_k)^{\alpha}$。

证明:采用如下的构造法证明定理的成立。令 $T_0=\{t_1,\cdots,t_k\}$,分别构造如下的 k 个 S-网 $\Sigma_i=(S,T_i;F_i,M_0)$,其中每个 S-网的库所集 S 和初始标识 M_0 均保持不变,并且满足:

(1) $T_i=(T-T_0)\cup\{t_i\}$;

(2) $F_i=F-(T_0-\{t_i\})\times S$。

易知,$Exp(\Sigma)=Exp(\Sigma_1)\parallel\cdots\parallel Exp(\Sigma_k)$。

对于每个 $\Sigma_i=(S,T_i;F_i,M_0)$,由引理 3 知,存在正规表达式 RE_i 使得 $Exp(\Sigma_i)=(RE_i)^{\alpha}$。

故,存在正规表达式 $RE_1,\cdots,RE_k$ 使得Σ 的进程表达式 $Exp(\Sigma)=(RE_1)^{\alpha}\parallel\cdots\parallel(RE_k)^{\alpha}$。

例 2 图 3 给出了一个不含初始标识的 S-网图 Σ_2,Σ_2 含有两个源变迁 t_{01}和 t_{02}。根据定理 2 的证明过程构造两个新的 S-网Σ_{21}和Σ_{22},如图 4 所示。可以求得Σ_{21}的五个基本进程段,如图 5 所示,其中 P_1 是一个起始程段,P_2 是一个传递进程段,P_3 是一个不变进程段,P_4 和 P_5 是两个终结进程段。Σ_{22}的五个基本进程段$(P'_1,P_i,i=2,3,4,5)$与Σ_{21}的相似,P'_1是将图 5 中 P_1 的 t_{01}改为 t_{02},其他 P_i 均为图 5 中的 P_i。可以求得 Σ_{21}和 Σ_{22}的进程表达式分别为:

$Exp(\Sigma_{21})=(P_1P_2(P_3)^*(P_4+P_5))^{\alpha}$,

$Exp(\Sigma_{22})=(P'_1P_2(P_3)^*(P_4+P_5))^{\alpha}$。

则 Σ_2 的进程表达式为:

$Exp(\Sigma_2)=Exp(\Sigma_{21})\parallel Exp(\Sigma_{22})$。

$=(P_1P_2(P_3)^*(P_4+P_5))^{\alpha}\parallel(P'_1P_2(P_3)^*(P_4+P_5))^{\alpha}$

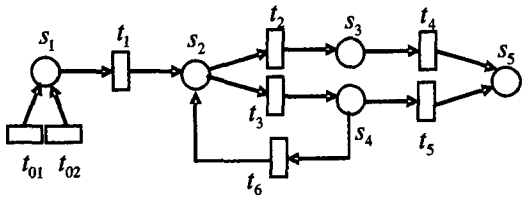
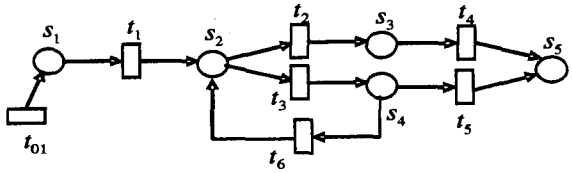


图 3 不含初始标识的 S-网Σ_2



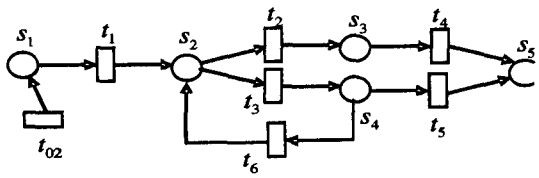


图 4 S-网Σ_{21}和Σ_{22}

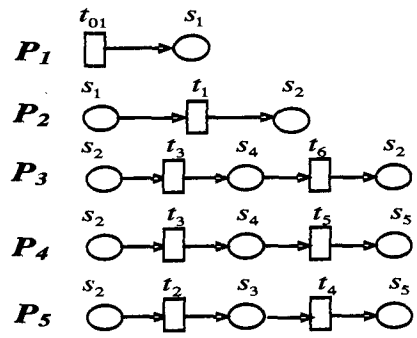


图 5 Σ_{21}的五个基本进程段

4.3 含初始标识的 S-网的进程表达式

定理 3 设 S-网 $\Sigma=(S,T;F,M_0)$中,存在 $t_1,\cdots,t_k(k\geqslant 1)$为 Σ的 k 个源变迁,且存在 $s\in S: M_0(s)\neq 0$,则存在正规表达式 $RE_1,\cdots,RE_k$ 和RE_{k+1}使得 Σ 的进程表达式 $Exp(\Sigma)=(RE_1)^{\alpha}\parallel\cdots\parallel(RE_k)^{\alpha}\parallel RE_{k+1}$。

证明:采用如下的方法证明该定理成立。令 $T_0=\{t_1,\cdots,t_k\}$,构造两个 S-网 $\Sigma_i=(S_i,T_i;F,M_{0i})(i=1,2)$,其中:

(1) $S_1=S;T_1=T;F_1=F;\forall s\in S: M_{01}(s)=0$。

(2) $S_2=S\cup\{s_0\};T_2=T;F_2=F\cup(s_0\times T_0);M_{02}=M_0$。

易证,$Exp(\Sigma)=Exp(\Sigma_1)\parallel Exp(\Sigma_2)$。同时可证,$\Sigma_1=(S_1,T_1;F_1,M_{01})$为含有 k 个源变迁但不含初始标识的 S-网,$\Sigma_2=(S_2,T_2;F_2,M_{02})$为标识 S-图。根据引理 3 和定理 2 知,存在正规表达式 $RE_1,\cdots,RE_k$ 使得 E_1 的进程表达式 $Exp(\Sigma_1)=(RE_1)^{\alpha}\parallel\cdots\parallel(RE_k)^{\alpha}$。根据定理 1 知,$Exp(\Sigma_2)$是一个正规表达式。所以,存在正规表达式 $RE_1,\cdots,RE_k$ 和RE_{k+1}使得 Σ 的进程表达式 $Exp(\Sigma)=(RE_1)^{\alpha}\parallel\cdots\parallel(RE_k)^{\alpha}\parallel RE_{k+1}$。

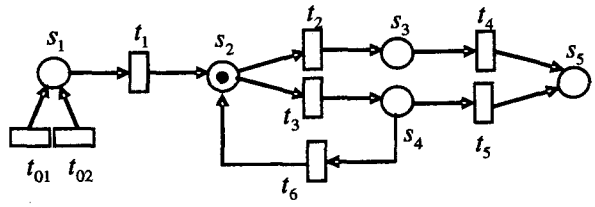
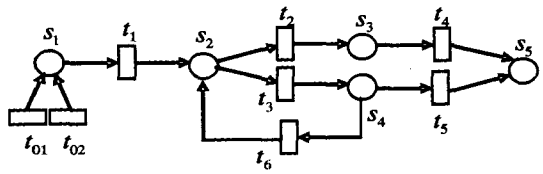


图 6 含初始标识的 S-网 Σ_3



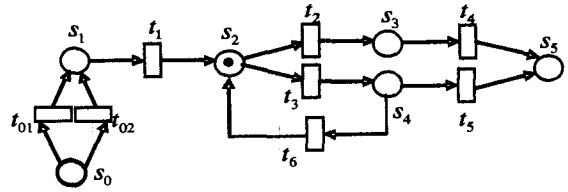


图 7 S-网 Σ_{31}和 Σ_{32}

例 3 图 6 给出一个含初始标识的 S-网 Σ_3,Σ_3 含有两个源变迁 t_{01}和 t_{02}。根据定理 3 的证明过程,首先构造两个新的 S-网Σ_{31}和 Σ_{32},如图 7 所示,容易证明 $Exp(\Sigma_3)=Exp(\Sigma_{31})\parallel Exp(\Sigma_{32})$。$\Sigma_{31}$即为例 2 中的 Σ_2,故:

(下转第 237 页)

……

}

5 运行时实时构件的反射流程

反射实时构件在运行时，可能由于构件运行时环境的变化，或用户某些特定的要求，需要更换不同的构件实现，来组装新的实时应用软件系统。例如，实时应用系统运行时，可能要求替换效率更高的实时调度构件，或选取具有完成时间约束特征不同的普通功能构件，这时需要依照反射实时构件模型，对构件库中的实时构件进行相应的按需调控，获取新的满足变化性要求的实时应用系统。其过程依次顺序对应反射实时构件模型中两种不同的反射流——UP 流和 DOWN 流。

5.1 UP 流

UP 流即从反射实时构件模型的基层，经元层 1 上升到元层 2，完成一个上升流过程。这个过程首先从实时应用系统中剥离出需要替换的实时构件，在构件库中查找其对应的配置属性文件，获得其相关的构件规约，进行语义的抽象和语法的提取，最后得到该实时构件的规约描述元信息，并进行分析。设 UP 流对应进程 $Flow_{up}$，其字母表为 $\alpha Flow_{up}$。

5.2 DOWN 流

DOWN 流即从反射实时构件模型的元层 2，经元层 1 下降到基层，完成一个下降流过程。这个过程首先对 UP 流得到的最终元信息分析结果，进行按需的“反省”或“调解”操作，获取新的实时构件规约描述元信息，并根据生成算法形成新的构件规约，然后借用用户平台相关的编译器生成新的满足要求的构件实现，最后装配到原有的实时应用系统中去。同时组织该构件实现和构件配置文件入库，以便今后对类似构件的使用。设 DOWN 流对应进程 $Flow_{down}$，其字母表为 $\alpha Flow_{down}$。

这两种方向截然不同的反射流在基于反射实时构件的实时应用系统中，正好形成了一个完整的反射流，实现了实时构件的按需变动和演化，增强了实时应用软件系统的适应性和健壮性。那么，运行时实时构件的反射流程可表示为 $ReflectiveFlow \overset{\Lambda}{=\!=} \mu X \cdot (Flow_{up} \rightarrow Flow_{down} \rightarrow X)$。

结束语 本文以现有的构件模型为研究基点，应用反射技术，针对实时应用系统的开发，提出了一类新的构件模型——反射式实时构件模型。它由反射式实时构件语义模型和反射式实时构件语法模型组成。该模型在规约了构件应有的功能需求特征的基础上，有效地标识了构件的时间约束特征，使得它与传统的功能性构件区开来，能被系统开发者更好地选用。同时，该模型结合反射技术，还能根据用户需求的变化，对实时构件进行动态的设计时修改，以便更准确地保障实时应用系统的构建与开发。这样，增强了构件设计的灵活性，达到了构件更好实现的效果。

反射式实时构件模型既是一个构件理论模型，也是一个工程模型。本文着重给出了它的规约描述机理。在接下来的工作中，我们将进一步完善该实时构件模型的反射基础设施，尤其是语法模型的构件实现所需的反射实时构件运行时环境。

参 考 文 献

1 Smith B C. Reflection and semantics in a procedural language: [LCS Technical ReportTR-272]. Cambridge, MA: M I T, 1982

2 Smith B C. Reflection and semantics in Lisp. In: Proceeding of the 1984 ACM Principles of Programming Language Conference, ACM, December 1984. 23～25

3 王渊峰，张涌，任洪敏. 基于刻面描述的构件检索. 软件学报，2002，13(8)：1546～1552

4 Damiani E, Fugini M G, Bellettini C. A hierarchy-aware approach to faceted classification of objected-oriented components. ACM Transactions on Software Engineering and Methodology, 1999, 8(3): 215～262

5 Henninger S. Supporting the process of satisfying information needs with reusable software libraries: an empirical study. In: Samadzadeh M H, Mansour K Z, eds. Proceedings of the 17th International Conference on Software Engineering on Symposium on Software Reusability. Seattle, WA: ACM Press, 1995. 267～270

6 Cazzola W. Evaluation of Object-Oriented Reflective Model. In: Proceeding of ECOOP Workshop on Reflective Object-Oriented programming and Systems(EWROOPS'98), Brussels, Belgium, July 1998

7 Zaremski A M, Wing J M. Specification Matching of Software Components. ACM Trans Softw Eng Method, 1997, 6(4): 333～369

8 Hoare C A R. Communicating Sequential Processing. Prentice Hall, 1985

9 Schneider S. An Operational Semantics for Timed CSP. Information and Computation Programming Research Group, Oxford University Computing Laboratory

10 Object Management Group. CORBA Components, version 3.0. OMG, Inc. June 2002

(上接第 227 页)

$Exp(\Sigma_{31}) = (P_1P_2(P_3)^*(P_4+P_5))^\alpha \parallel (P'_1P_2(P_3)^*(P_4+P_5))^\alpha$，其中 $P'_1, P_1 (i=2,3,4,5)$ 分别如图 5 所示。

容易求得 Σ_{32} 三个基本进程段就是图 2 中的 P_3，P_4 和 P_5 并且可以求得 $Exp(\Sigma_{32}) = P_3^*(P_4+P_5)$。故：

$$\begin{aligned} Exp(\Sigma_3) &= Exp(\Sigma_{31}) \parallel Exp(\Sigma_{32}) \\ &= (P_1P_2(P_3)^*(P_4P_5))^\alpha \parallel (P'_1P_2(P_3)^*(P_4P_5))^\alpha \parallel P_3^*(P_4+P_5) \end{aligned}$$

结束语 由于 S-网具有很好的结构特征，其进程行为易于分析和描述。本文分析了结构简单的 S-网的进程行为，给出了各类 S-网的进程表达式的求取方法。我们分析 S-网的进程行为是为结构复杂 Petri 网的进程行为描述服务的。通过适当的分解方法可以将一个结构复杂的 Petri 网分解成结构简单的子网(如，S-网)，分析分解过程中满足的进程语义，通过分解后的子网来分析原系统的进程行为。或者将一些结构简单的子网合成一个结构复杂的 Petri 网，分析合成过程中的进程语义，通过结构简单的子网分析合成后的复杂网系统的进程行为。通过分解(合成)分析结构复杂 Petri 网的进程行为，我们已经得到了相关的方法和结果，将会在其他的文章中给出。

参 考 文 献

1 袁崇义. Petri 网原理. 北京：电子工业出版社，1998

2 Peterson J. Petri net theory and the modeling of systems. 吴哲辉译. 徐州：中国矿业大学出版社，1989

3 Murata T. Petri Nets: Properties, Analysis and Applications. Proceedings of The IEEE, 1989, 77(4)

4 Gltz U, Reisig W. Processes of place/transition net, LNCS, New York: Springer-Verlag, 1983, 154: 264～277

5 Lu Ruqian. P/R nets and P/R processes. Science in China (Series E), 1992, 35(1): 21～31

6 曾庆田，吴哲辉. Petri 网的进程网系统. 计算机学报，2002，25(12)：1308～1315

7 Wu Zhehui. Process expression of bounded Petri net, Science in China (Series E), 1996, 39(1): 37～49

8 吴哲辉，王培良，赵茂先. 无界公平 Petri 网的进程表达式. 计算机学报，2000，23(4)：337～344

9 曾庆田，吴哲辉. 无界 Petri 网的进程表达式，计算机学报，2003，26(12)：1629～1636

10 曾庆田，吴哲辉. 类 S－图的语言性质分析. 计算机科学，2002，29(5)：120～123

11 段华，曾庆田. S-网的活性分析. 小型微型计算机系统，2004，25(11)：1975～1978