稳定有界Petri网的进程表达式

汪明新 刘关俊 闫春钢
(同济大学计算机科学与工程系 上海 201804)
(同济大学嵌入式系统与服务计算教育部重点实验室 上海 201804)

摘 要 文献[1]证明一个有界Petri网的进程表达式是以该网的基本子进程集为字母表的正规表达式,然而没有给出基本子进程的求解方法。定义了一类有界Petri网——稳定有界Petri网,并给出其基本子进程的求解算法,进而利用有限自动机的语言表达式的求解方法来求解稳定有界Petri网的进程表达式。另外,还给出了由符合一定条件的S-网的进程表达式来构造其同步合成网的进程表达式的算法。

关键词 有界Petri网,进程表达式,基本子进程,S-网
中图法分类号 TP393 **文献标识码** A

Process Expression of Stable Bounded Petri Nets

WANG Ming-xin LIU Guan-jun YAN Chun-gang
(Department of Computer Science & Engineering, Tongji University, Shanghai 201804, China)
(Key Laboratory of Embedded System and Service Computing of Ministry of Education, Tongji University, Shanghai 201804, China)

Abstract The paper[1] proved that the process expression of a bounded Petri net is a regular expression with the basic subprocess set of the net system as alphabet. However, methods for the solution of the basic subprocess set have not been proposed so far. This paper defined a class of bounded Petri nets that are named by stable bounded Petri nets(SBPNs). A method was presented to compute the set of basic subprocesses, and then the process expression of an SBPN could be produced by a finite state machine with the basic subprocess set of the net system as alphabet. Next, an algorithm was proposed to construct the process expression of a Petri net synchronized by a set of S-nets.

Keywords Bounded petri net, Process expression, Basic subprocess, S-net

1 引言

作为一种系统模型,Petri网便于描述系统中的顺序、并发、冲突以及同步等关系。为了通过Petri网研究被模拟系统的性质,网论中已提出了若干种Petri网分析方法,其中Petri网进程是Petri网行为描述和分析的方法之一。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(可能是无限多个)个进程,不可能一一列举。为此,文献[1]提出了有界Petri网进程表达式的概念,并给出了求有界Petri网进程表达式的一个算法。一个有界Petri网的进程表达式是以该网的基本子进程集为字母表的正规表达式。文献[2]对可重复子进程的概念进行了拓展,给出了无界公平Petri网进程表达式的求取方法。对于任意无界Petri网,可以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[3]。

文献[1]尽管证明一个有界Petri网的进程表达式是以该网的基本子进程集为字母表的正规表达式,但并没有给出基本子进程和每个互达类所对应的表达式的求取方法。文献[3]中通过特征可达树求取基本进程段集,但是求得的基本进程段集并不能很好地反映原网系统的行为,而且文中也没给出怎样连接所求得的基本子进程来构造其进程网系统的方法。

本文提出了稳定有界Petri网的概念,给出了稳定有界Petri网的基本子进程的求取方法,并由所求的基本子进程集来构造进程表达式,即能够表达出此Petir网的所有进程。另外,本文给出了由符合一定条件的S-网的进程表达式来构造其同步合成网进程表达式的一个算法。

2 基本概念

关于Petri网及其进程的详细内容参看文献[1-3]。这里只给出有关的基本概念、术语和记号。

定义1 若$N=(B,E;G)$满足条件

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

$$\forall x,y\in B\cup E:(x,y)\in G+\rightarrow(y,x)\notin G^{+} \quad (2)$$

则称N为一个出现网,其中G^{+}表示流关系G的传递闭包。

到稿日期:2010-03-02 返修日期:2010-06-09 本文受国家重点研究发展计划课题(2010CB328101),国家"八六三"高技术研究发展计划基金项目(2009AA01Z401),国家自然科学基金项目(90718012)和上海市基础研究重点项目(08JC1419300)资助。

汪明新(1986-),男,硕士,主要研究方向为Petri网理论与应用、工作流建模,E-mail:wmxint@gmail.com;刘关俊(1978-),男,博士;闫春钢博士,教授。

定义 2 设 $N=(B,E;G)$为一个出现网，$u\subseteq B\cup E$，如果 $\forall x,y\in u:(x,y)\notin G^+\wedge(y,x)\notin G^+$，则称 u 为 N 的一个切集，而且任意 $u'\supset u$ 都不是 N 的切集，则称 u 为 N 的一个切。若 u 是 N 的一个切，且 $u\subseteq B$，则称 u 为 N 的一个 s-切。

设 u_1,u_2 为 N 的两个切，如果 $\forall x\in u_1,\exists y\in u_2:(x,y)\in G^*$，则记为 $u_1\leqslant u_2$。

定义 3 设 $\Sigma=(S,T;F,M_0)$为一个 Petri 网，$N=(B,E;G)$为一个出现网，若映射 $\varphi:B\cup E\to S\cup T$ 满足条件

$$\varphi(B)\subseteq S,\varphi(E)\subseteq T \tag{3}$$

$$\forall x,y\in B\cup E:(x,y)\in G\to(\varphi(x),\varphi(y))\in F \tag{4}$$

$$\forall e\in E:\varphi(\cdot e)=\cdot\varphi(e)\wedge\varphi(e^\cdot)=\varphi(e)^\cdot \tag{5}$$

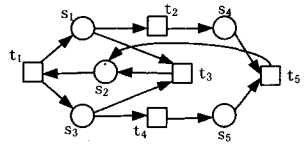
则称 φ 定义了 N 到 Σ 的一个映射，记为 $\varphi:N\to\Sigma$。如果这个映射满足条件

$$\forall b_1,b_2\in B(b_1\neq b_2):\varphi(b_1)=\varphi(b_2)\to\cdot b_1\neq\cdot b_2\wedge b_1{}^\cdot\neq b_2{}^\cdot \tag{6}$$

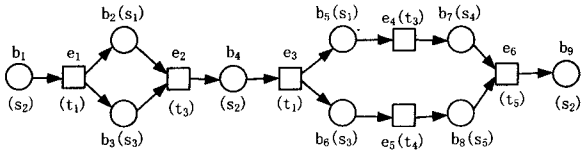
$$\forall s\in S:|\{b|\varphi(b)=s\wedge\cdot b=\emptyset\}|\leqslant M_0(s) \tag{7}$$

则称(N,φ)为 Petri 网 Σ 的一个进程。

例 1　图 1(a)是一个 Petri 网 $\Sigma=(S,T;F,M_0)$，图 1(b)是一个出现网 $N=(B,E;G)$，出现网中的每个基本元素旁有两个符号。括号外的符号是出现网中的元素标号，括号内的符号是映射 $\varphi:B\cup E\to S\cup T$ 中该元素的映像。可以看出映射 $\varphi:B\cup E\to S\cup T$ 满足式(3)－式(5)，从而它定义了从出现网 N 到原网的一个映射。再结合 Petri 网的初始标识进行考察，可以发现式(6)和式(7)也成立，因此图 1(b)的出现网是 Petri 网 Σ 的一个进程。



(a)Petri 网$\Sigma=(S,T;F,M_0)$



(b)出现网 $N=(B,E;G)$及映射 $\varphi:B\cup E\to S\cup T$

图 1　Petri 网Σ和Σ的一个进程(N,φ)

定义 4 设 φ 为出现网 $N=(B,E;G)$到有界 Petri 网 $\Sigma=(S,T;F,M_0)$的一个映射，如果

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

$$\varphi(b_1)=\varphi(b_2)\to(\cdot b_1\neq\cdot b_2\vee\cdot b_1=\cdot b_2=\emptyset)\wedge(b_1{}^\cdot\neq b_2{}^\cdot\vee b_1{}^\cdot=b_2{}^\cdot=\emptyset) \tag{8}$$

(2) $\forall s\in S:|\{b|\varphi(b)=s\wedge\cdot b=\emptyset\}|=M_0(s)$ (9)

则称 $P=(N,\varphi)$为 Σ 的一个满进程。

定义 5 设 $P=(N,\varphi)$为 Σ 的一个满进程，u_1 和 u_2 是 N 的两个 s 切，$u_1\leqslant u_2$，$\varphi(u_1)=\varphi(u_2)$，则称$(N[u_1,u_2],\varphi)$为 P 的一个可重复子进程。

定义 6 设 $P=(N,\varphi)$为 Σ 的一个满进程，$P_1=(N[u_1,u_2],\varphi)$是 P 的一个非空可重复子进程，如果 P_1 的任意非空真子进程都不是可重复子进程，则称 P_1 为 Σ 的一个闭基本子进程。

通过验证可以发现图 1(b)的进程(N,φ)满足式(8)和式(9)，所以(N,φ)是 Petri 网 Σ 的一个满进程。$\{b_1\},\{b_4\}$和$\{b_9\}$都是 N 的 s 切，$\{b_1\}\leqslant\{b_4\}\leqslant\{b_9\}$，$\varphi(b_1)=\varphi(b_4)=\varphi(b_9)$，所以$(N[b_1,b_4],\varphi)$，$(N[b_1,b_9],\varphi)$和$(N[b_4,b_9],\varphi)$均为$(N,\varphi)$的可重复子进程。再由闭基本子进程的定义可以验证$(N[b_1,b_4],\varphi)$和$(N[b_4,b_9],\varphi)$均为$(N,\varphi)$的闭基本子进程。

定义 7 设 $P=(N,\varphi)$为 Σ 的一个满进程，$P_1=(N[u_1,u_2],\varphi)$是 P 的一个子进程，如果 $N[u_1,u_2]$中的任意两个 s 切 u_i 和 u_j 都有 $u_i\neq u_j\to\varphi(u_i)\neq\varphi(u_j)$，则称 P_1 为 Σ 的一个开基本子进程。

定义 8 设 Σ 为一个有界 Petri 网，Σ 的闭基本子进程集和开基本子进程集的并集称为 Σ 的基本子进程集。

注意：如果 $N[u_1,u_2]$与 $N[u_3,u_4]$在映射 φ 下对应的 Petri 网相同，则$(N[u_1,u_2],\varphi)$与$(N[u_3,u_4],\varphi)$被看作是同一子进程。

定义 9 设 Σ 为一个有界 Petri 网，$Exp(P(\Sigma))$是以 Σ 的基本子进程集为字母表的一个正规表达式，它所表示的正规集记为 $RS(P(\Sigma))$。如果 Σ 的每个满进程都是集合

$$Pref\{Exp(P(\Sigma))\}=\bigcup_{P\in RS(P(\Sigma))}Pref(P) \tag{10}$$

的一个元素，则称 $Exp(P(\Sigma))$为 Σ 的进程表达式。

3　稳定有界 Petri 网的进程表达式求解

定义 10 设 $\Sigma=(S,T;F,M_0)$为一个 Petri 网，$M\in R(M_0)$，记

$$\cdot M=\{M'\in R(M_0)\wedge\text{存在 }t\in T\text{ 使得 }M'[t>M\}。$$

称 $\cdot M$ 为 M 的前状态集。

定义 11 设 $\Sigma=(S,T;F,M_0)$为一个 Petri 网，$M\in R(M_0)$。若不存在 $t\in T$ 使得 $M[t>$，则称 M 为 Σ 的死状态。

定义 12[4]　设 $\Sigma=(S,T;F,M_0)$为一个 Petri 网，$M\in R(M_0)$。如果 $\forall M'\in R(M)$，都有 $M\in R(M')$，则称 M 为 Σ 的一个家态。对于 Σ 的任意一个家态 M，如果它的前状态集中存在某个状态不是家态，则称 M 为 Σ 的准家态。

定义 13 若 $t\in T$ 满足 $|\cdot t|>1$，则 t 称为同步变迁。

定义 14 给定一个有界 Petri 网 $\Sigma=(S,T;F,M_0)$，对于 Σ 的同步变迁 $t\in T$，若存在 $M_i\in R(M_0)$满足：1) $\forall s\in\cdot t:M_i(s)=1$；2) $\forall s\notin\cdot t:M_i(s)=0$，则称 M_i 为 Σ 的关于变迁 t 的稳态。如果对于 Petri 网 Σ 的所有同步变迁 t 均存在稳态，则 Σ 称为稳定有界 Petri 网。所有同步变迁 t 的稳态的集合称为 Σ 的稳态集。

显然，一个网可以有一个或多个稳态，也可能没有稳态。如图 2 中的 Petri 网 Σ 就没有稳态。算法 1 是针对稳定有界 Petri 网的进程表达式的求解。

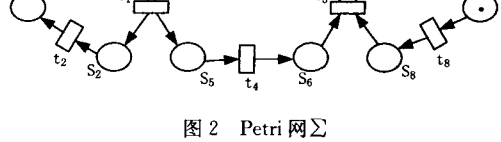


图 2　Petri 网 Σ

算法 1　稳定有界 Petri 网的进程表达式求解

输入：稳定有界 Petri 网 $\Sigma=(S,T;F,M_0)$

输出：$Exp(\Sigma)$

Step1　依据 Σ 的可达图求出 Σ 的所有稳态(记为 $M_l-M_l,l\geqslant 0$)、准

家态(记为 $M_{l+1}-M_m, m\geqslant l$)和死状态(记为 $M_{m+1}-M_n, n\geqslant m$)。令 $A=\{M_0\}\cup\{M_1,\cdots,M_n\}$。

Step2 求出Σ的基本子进程集 P。

$P:=\emptyset$;

对每个 $M_i(i=0,1,\cdots,n)$:

For $i=0$ to n

For $j=0$ to n

If(如果在Σ的可达图中存在从 M_i 到 M_j 的一条简单路径使得这条简单路径上的其他结点均不是 A 中的元素) Then

将对应 M_i 到 M_j 的基本子进程加入到 P 中。

最后可以得到 Σ 的基本子进程集 P。

Step 3 将Σ转换成相应的确定有限自动机 $FM=(A,P,\delta,M_0,q_f)$。该有限自动机以 M_0 作为初态,以 q_f 作为终态集(记Σ的死状态集为 Q_1,只能到达自身且不能到达其他状态的准家态集为 Q_2。若 $Q_1\neq\emptyset$,则 $q_f=Q_1$;若 $Q_1=\emptyset$,则 $q_f=Q_2$;若 $Q_1=\emptyset\wedge Q_2=\emptyset$,则 $q_f=M_0$)。

Step 4 求该网的进程表达式。求解 Step 3 中确定有限自动机的语言表达式即为该网的进程表达式。具体求解过程如下:

取 R_{im}^k 表示所有从状态 M_i 到状态 M_m 且不经过下标超过 k 的状态的字符串,R_{im}^k 的递归定义如下:

$$R_{im}^k=R_{ik}^{k-1}(R_{kk}^{k-1})^* R_{km}^{k-1}\cup R_{im}^{k-1}$$

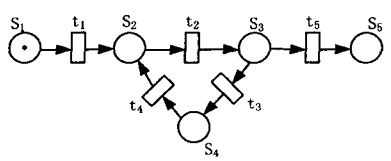
$$R_{ij}^k=\begin{cases}\{a\mid\delta(M_i,a)=M_m\}, & \text{if } i\neq m\\ \{a\mid\delta(M_i,a)=M_m\}\cup\{\varepsilon\}, & \text{if } i=m\end{cases}$$

R_{im}^n 表示所有从状态 M_i 到状态 M_m 的串,这样我们就可以得到

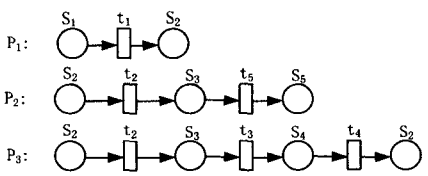
$\mathrm{Exp}(\Sigma)=\mathrm{L(FM)}=R_{1j_1}^n+R_{1j_2}^n+\cdots+R_{1j_p}^n$ ($q_f=\{q_{j_1},q_{j_2},\cdots,q_{j_p}\}$)。

算法 1 的正确性证明思路:经过步骤 1—3 得到的确定有限自动机 FM 可以看成是Σ对应的确定有限自动机经过一定的化简而得到的。化简的时候,将Σ对应的确定有限自动机的特定部分用 Step2 中的基本子进程来代替,这样得到 Step3 中的 FM 与Σ所对应的确定有限自动机的语言是等价的。再通过 Step4 得到的 FM 的语言表达式即为Σ的进程表达式,Step4 中的求解过程参考文献[5]中的定理 2.4 的证明过程。

例 2 Petri 网Σ_1 如图 3(a)所示,网中不含同步变迁,结构比较简单。根据算法 1,初始状态为 $M_0=[10000]$,准家态 $M_1=[01000]$,死状态 $M_2=[00001]$。我们可以求得Σ_1 的基本子进程集,如图 3(b)所示。然后由基本子进程集构造Σ_1 的进程表达式:$Exp(\Sigma_1)=P_1(P_3)^* P_2$。



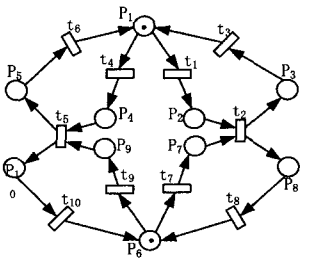
(a) Petri 网Σ_1



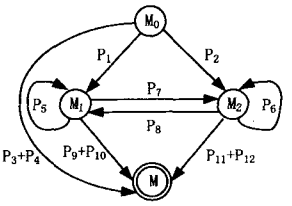
(b) Petri 网Σ_1 的基本子进程集

图 3 Petri 网Σ_1 及其基本子进程集

例 3 Petri 网Σ_2 如图 4(a)所示。



(a) Petri 网Σ_2



(b) 对应的 DFA FM

图 4 Petri 网Σ_2 及其对应的 DFA FM

对于图 4(a)中的 Petri 网Σ_2,根据文献[1]中的求进程表达式的算法,Petri 网 Σ_2 有 3 个互达类[1] $[M_0]$,$[M_1]$和 $[M_2]$。其中$|[M_1]|=|[M_2]|=1$,然而$[M_0]$所对应的可重复子进程比较复杂,算法中也并没有给出求基本子进程的方法,所以文献[1]中的算法并不能有效地求解这类有界 Petri 网的进程表达式。

下面根据算法 1 来求 Σ_2 的进程表达式,它的求解过程如下。

Step1 该网的初始状态为 $M_0=[1000010000]$,同步变迁有 t_2 和 t_5,Σ_2 的稳态有 $M_1=[0100001000]$和 $M_2=[0001000010]$。死状态有 $Q_1=\{M_{01}=[0100000010]$和 $M_{02}=[0001001000]\}$。

Step2 求出 Σ 的基本子进程集。

由 M_0 到 M_1 对应的基本子进程为 $P_1=t_1||t_7$,由 M_0 到 M_2 对应的基本子进程为 $P_2=t_4||t_9$,由 M_0 到终态 M_{01} 和 M_{02}对应的基本子进程分别为 $P_3=t_2||t_9$ 和 $P_4=t_4||t_7$,由 M_1 到 M_1 对应的基本子进程为 $P_5=t_2(t_3t_1||t_8t_7)$,由 M_2 到 M_2 对应的基本子进程为 $P_6=t_5(t_{10}t_9||t_6t_4)$,由 M_1 到 M_2 对应的基本子进程为 $P_7=t_2(t_3t_4||t_8t_9)$,由 M_2 到 M_1 对应的基本子进程为 $P_8=t_5(t_{10}t_7||t_6t_1)$,由 M_1 到 Q_1 对应的基本子进程为 $P_9=t_2(t_3t_1||t_8t_9)$或 $P_{10}=t_2(t_3t_4||t_8t_7)$,由 M_2 到 Q_1 对应的基本子进程为 $P_{11}=t_5(t_{10}t_9||t_6t_1)$或 $P_{12}=t_5(t_{10}t_7||t_6t_4)$。所以求得该网的基本进程子进程集为 $P=\{P_1,P_2,P_3,P_4,P_5,P_6,P_7,P_8,P_9,P_{10},P_{11},P_{12}\}$。

Step3 将改网转换成相应的有限自动机,这里 $Q_1\neq\emptyset$,所以 $q_f=Q_1$。

$FM=(\{M_0,M_1,M_2,Q_1\},\{P_1,P_2,P_3,P_4,P_5,P_6,P_7,P_8,P_9,P_{10},P_{11},P_{12}\},\delta,M_0,q_f)$,其中 δ 为

$\delta(M_0,P_1)=M_1$ $\delta(M_0,P_2)=M_2$

$\delta(M_0,P_3)=Q_1$ $\delta(M_0,P_4)=Q_1$

$\delta(M_1,P_5)=M_1$ $\delta(M_2,P_6)=M_2$

$\delta(M_1,P_7)=M_2$ $\delta(M_2,P_8)=M_1$

$\delta(M_1,P_9)=Q_1$ $\delta(M_1,P_{10})=Q_1$

$\delta(M_2,P_{11})=Q_1$ $\delta(M_2,P_{12})=M_1$

图 4(b)为对应的 DFA。

Step4 求得该有限自动机的语言表达式,亦即该网的进程表达式。根据 Step4 中的方法求得 Petri 网Σ_2 的进程表达式如下:

$$Exp(PN)=P_3+P_4+(P_2+P_1P_5^*P_7)(P_6+P_8P_5^*P_7)^*[(P_{11}+P_{12})+P_8P_5^*(P_9+P_{10})]+P_1P_5^*(P_9+P_{10})$$

4 一类 S-网的同步合成网的进程表达式的求解

定义 15[7] 如果可以寻找一组 S-网 $\Sigma_1,\Sigma_2,\cdots,\Sigma_k$ 满足同步合成的结果恰好为 $\Sigma=(P,T;F,M_0)$，每个子网 $\Sigma_i=(P_i,T_i;F_i,M_{0i})(i=1,2,\cdots,k)$ 满足：

1) $P=P_1\cup P_2\cup\cdots\cup P_k, P_1\cap P_2\cap\cdots\cap P_k=\emptyset$;

2) $T=T_1\cup T_2\cup\cdots\cup T_k, T_1\cap T_2\cap\cdots\cap T_k=\emptyset$;

3) $F_i=(S_i\times T_i)\cup(T_i\times S_i)\cap F$;

4) $M_0(p)=M_{0i}(p)$，若 $p\in P_i$;

则称 Σ 是 $\Sigma_i(i=1,2,\cdots,k)$ 的同步合成网，记作 $\Sigma=\Sigma_1\ O_T\ \Sigma_2\ O_T\cdots O_T\ \Sigma_k$。

下面给出的定义 16－定义 18 可参考文献[6]。

定义 16 取 X 是一个字母表。对任意一个字符串 $\omega\in X^*$，我们用 $|\omega|$ 代表字符串 ω 的长度；对任意一个字符 $a\in X$，我们用 $|\omega|_a$ 代表 a 在 ω 中出现的次数；对任意语言 $L\subseteq X^*$，我们用 $\alpha(L)$ 表示 L 的字母表，即 $\alpha(L)=\{x\in X\mid\exists u\in L:|u|_x>0\}$。

定义 17 设 X 是一个有限字母表，$Y\subseteq X$。令 $\Pi_{X,Y}:X^*\to Y^*$，使得 $\forall\sigma\in X^*$，且 $\Pi_{X,Y}(\sigma)$ 是从 σ 中删除 $X-Y$ 中所有字符后剩余的子串，则称 $\Pi_{X,Y}$ 为从 X 到 Y 的一个投影映射。在不至于混淆的情况下，我们用 Π_Y 代替 $\Pi_{X,Y}$。

定义 18 设 X 是一个字母表，L_1,L_2 是包含在 X^* 中的两个语言，则 L_1 和 L_2 的同步合成运算记作 $L_1\Theta L_2=\{\omega\in(\alpha(L_1)\cup\alpha(L_2))^*\mid\Pi_{\alpha(L_i)}(\omega)\in L_i,i=1,2\}$。

由定义 15－定义 18，可以得到以下运算规则：

(1) $A_1a_1A_2a_2\cdots A_ma_mA_{m+1}\Theta B_1a_1B_2a_2\cdots B_ma_mB_{m+1}=(A_1||B_1)a_1(A_2||B_2)a_2\cdots(A_m||B_m)a_m(A_{m+1}||B_{m+1})$;

(2) $(a_1A_1a_2A_2\cdots a_mA_m)^*\Theta(a_1B_1a_2B_2\cdots a_mB_m)^*=(a_1(A_1||B_1)a_2(A_2||B_2)\cdots a_m(A_m||B_m))^*$;

(3) $(A_1a_1A_2a_2\cdots A_ma_mA_{m+1})^*\Theta(B_1a_1B_2a_2\cdots B_ma_mB_{m+1})^*=(A_1||B_1)(a_1(A_2||B_2)a_2\cdots(A_m||B_m)a_m(A_{m+1}A_1||B_{m+1}B_1))^*a_1(A_2||B_2)a_2\cdots(A_m||B_m)a_m(A_{m+1}||B_{m+1})$;

(4) $aA\Theta((aB)^*C)^*=a(A||BC)$;

(5) $(aA)^*\Theta((aB)^*C)^*=((a(A||BC))^*C+a(A||BC))^*$;

(6) $(a_1A_1a_2A_2\cdots a_iA_ia_{i+1}A_{i+1}\cdots a_jA_j\cdots)^*\Theta(a_1B_1a_2B_2\cdots a_iB_ia_jB_{i+1}\cdots a_{i+1}B_t\cdots)^*=a_1(A_1||B_1)a_2(A_2||B_2)\cdots a_i(A_i||B_t)(j,t>i+1)$。

注：以上的 A_i 和 B_j 可以是有限变迁序列，也可以是无限变迁序列，并且它们没有共同的元素。

Petri 网的同步合成操作是复杂系统行为分析的一种十分有效的方法，下面给出的算法是通过两个 S-网的进程表达式来求解同步合成网的进程表达式。算法针对可重复基本子进程的同步合成来求解合成网的进程表达式。如果基本子进程中既含有闭基本子进程，又含有开基本子进程，则对于开基本子进程那部分的同步合成比较简单，可直接根据规则(1)—(5)直接求解；对于可重复子进程那部分的同步合成，如果满足定理 1 中的条件，则可以根据算法 2 来求解。

设 $\Sigma_i=(P_i,T_i;F_i,M_{0i})(i=1,2)$ 是两个 Petri 网，Σ 为 Σ_i 与 Σ_2 的同步合成网，即 $\Sigma=\Sigma_1O_T\Sigma_2$。$Exp(\Sigma_1)$，$Exp(\Sigma_2)$ 和 $Exp(\Sigma_2)$ 分别为 Σ_1，Σ_2 和 Σ 的进程表达式，这里假设 Σ_1 和 Σ_2 中只含有可重复子进程。这样可以令 $Exp(\Sigma_1)=pref(A_1+A_2+\cdots+A_m)^*$，$Exp(\Sigma_2)=pref(B_1+B_2+\cdots+B_m)^*$。$A_i(1\leqslant i\leqslant m)$ 为 Σ_1 的闭基本子进程，$B_j(1\leqslant j\leqslant n)$ 为 Σ_2 的闭基本子进程。

定理 1 若 $Exp(\Sigma_1)$ 与 $Exp(\Sigma_2)$ 满足 $A_i\cap B_j\neq\emptyset(1\leqslant i\leqslant m,1\leqslant j\leqslant n)\to A_i$ 与 B_j 的第一个字符相同，$BP(*)$ 表示 $*$ 的基本进程段集合，其中 $*\in(PN_1,PN_2,PN)$，$BP(\Sigma)=\{C\mid C=A_i\Theta B_j\wedge A_i\cap B_j\neq\emptyset\}\cup\{A_i\mid\alpha(A_i)\cap\alpha(Exp(\Sigma_2)=\emptyset)\}\cup\{B_j\mid\alpha(B_j)\cap\alpha(Exp(\Sigma_1)=\emptyset)\}$。

定理 1 的正确性由算法 2 是显而易见的。下面给出算法 2，根据该算法可以构造满足定理 1 的两个 Petri 网的同步合成网的进程表达式。

算法 2 一类特殊 S-网的同步合成网的进程表达式的求解

输入：$Exp(\Sigma_1)$ 和 $Exp(\Sigma_2)$

输出：$Exp(\Sigma)$

Step1 初始化，令

$a[1]=a[2]=\cdots a[m]=1, b[1]=b[2]=\cdots b[n]=1, t=0, s=0, h=0, k=0$。

Step2 For $i=1$ to m

For $j=1$ to n

如果 $A_i\cap B_j\neq\emptyset$，判断 $\Pi_{A_i,\Delta}(A_i)=\Pi_{B_j,\Delta}(B_j)(\Delta=A_i\cap B_j)$ 是否成立，如果成立，则按规则(2)计算 $C_t=A_i\otimes B_j$。如果不成立，则按运算规则(6)计算 $D_s=A_i\otimes B_j$。令 $a[i]=0, b[j]=0$。$t{+}{+}, s{+}{+}$。

Step 3 For $i=1$ to m

If$(a[i]=1)$ $E_h=A_i$

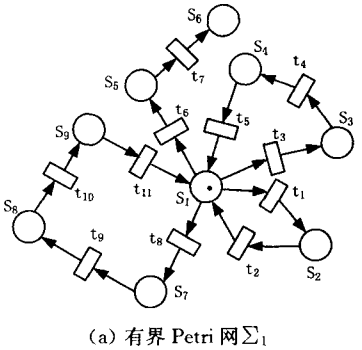
For $j=1$ to n

If$(b[j]=1)$ $F_k=B_j$

最后得到 $Exp(\Sigma)=pref(C_1+C_2+\cdots+C_t+(E_1+E_2+\cdots+E_h)^*\mid|(F_1+F_2+\cdots+F_k)^*)^*+D_1+D_2+\cdots+D_s$，算法结束。

算法 2 的正确性可以由定义 18 和同步合成的运算规则保证。

例 4 有界 Petri 网 $\Sigma_i=(P_i,T_i;F_i,M_{0i})(i=1,2)$ 如图 5 和图 6 所示，它们的同步合成网为 $\Sigma=(P,T;F,M)$。Σ_1 对应的基本子进程集为 $BP(\Sigma_1)=\{P_{11},P_{12},P_{13},P_{14}\}$，$\Sigma_2$ 对应的基本子进程集为 $BP(\Sigma_2)=\{P_{21},P_{22},P_{23}\}$；$\Sigma_1$ 的进程表达式为 $Exp(\Sigma_1)=(P_{12}+P_{13}+P_{14})^*+P_{11}$，$\Sigma_2$ 的进程表达式为 $Exp(\Sigma_2)=(P_{21}+P_{22}+P_{23})^*$。其中 P_{11} 是非可重复子进程，它与 P_{21} 有同步变迁 t_6，根据规则(4)得到同步合成网的一个基本子进程为 $P_1=t_6(t_7||t_{12})$。对于可重复子进程的同步合成，我们根据算法 2 来求解，P_{13} 与 P_{22} 同步合成得到 $C_1=t_3(t_4t_5||t_{13})$，P_{14} 与 P_{23} 同步合成得到 $D_1=t_8(t_9||t_{14})$，P_{12} 与 Σ_2 没有同步变迁，所以 $E_1=P_{12}$，从而得到有界 Petri 网 Σ_1 和 Σ_2 的同步合成网 Σ 的基本子进程集 $BP(\Sigma_1)=\{P_1,P_2=C_1,P_3=D_1,P_4=E_1\}$，根据算法 2 得到 Σ 的进程表达式为：$Exp(\Sigma)=(P_2+P_4)^*+P_3+P_1$。



(a) 有界 Petri 网 Σ_1

(下转第 67 页)

比，如果同时应用两个约束条件则可以平均提升 16.3%的缩减比。

在接下来的工作中，我们将在以下方面做进一步研究：(1)扩大存储环境的感知范围，例如通过感知网络特性来分析网络存储环境下优化 CDC 算法的约束条件；(2)分析约束条件 2 中降低缩减比的原因，并对边界特征值选择算法进行调整。

参 考 文 献

[1] Zhu B, Kai L, Patterson H. Avoiding the disk bottleneck in the data domain deduplication file system[C] // 6th USENIX Conference on File and Storage Technologies(FAST'08). San Jose, CA, USA, 2008

[2] Cox L P, Murray C D, Noble B D. Pastiche: making backup cheap and easy [C] // Proceedings of the Fifth Symposium on Operating Systems Design and Implementation(OSDI'02). Boston, MA, USA, 2002

[3] Quinlan S, Dorward S. Venti: a new approach to archival storage [C]// Proceedings of the Conference on File and Storage Technologies(FAST'02). Monterey, CA, USA, 2002

[4] Douglis F, Iyengar A. Application-specific delta-encoding via resemblance detection [C] // General Track 2003 USENIX Annual Technical Conference. San Antonio, TX, USA, 2003

[5] Jain N, Dahlia M, Tewari R. TAPER: tiered approach for eliminating redundancy in replica synchronization [C] // 4th USENIX Conference on File and Storage Technologies(FAST'05). San Francisco, CA, USA, 2005

[6] Bobbarjung D R, Jagannathan P, Dubnicki P. Improving duplicate elimination in storage systems [J]. ACM Transactions on Storage(TOS), 2006, 2(4): 25

[7] Chuanyi L, Dapeng J, Yu G, et al. Semantic data de-duplication for archival storage systems [C]// 13th Asia-Pacific Computer Systems Architecture Conference(ACSAC'08). Hsinchu, China, 2008

[8] Lillibridge M, Eshghi K, Bhagwat D, et al. Sparse Indexing: Large Scale, Inline Deduplication Using Sampling and Locality [C] // FAST 09. 2009

[9] You L L, Karamanolis C. Evaluation of Efficient Archival Storage Techniques [C]// Proceedings of the 21st IEEE Symposium on Mass Storage Systems and Technologies(MSST'04). 2004

[10] Muthitacharoen A, Chen B, Mazieres D. A low-bandwidth network file system[J]. Operating Systems Review(ACM), 2002, 35(5): 174-187

[11] Hong B, Plantenberg D, Long D D E, et al. Duplicate Data Elimination in a SAN File System [C] // Proceedings of the 21st IEEE/12th NASA Goddard Conference on Mass Storage Systems and Technologies(MSST'04). 2004

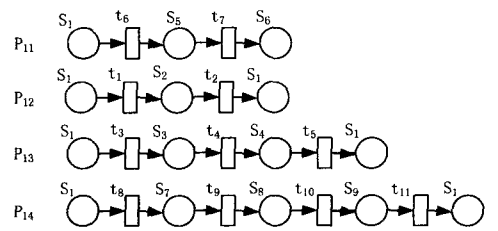
[12] Rabin. Fingerprinting by random polynomials [R]. 1981

[13] Manber U. Finding similar files in a large file system [C]// Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. San Francisco, CA, USA, 1994

[14] Moodalbail G, Mandagere N, Raghuveer A, et al. Backup aware object based storage[R]. 2007-25. DTC Intelligent Storage Consortium, June 2007

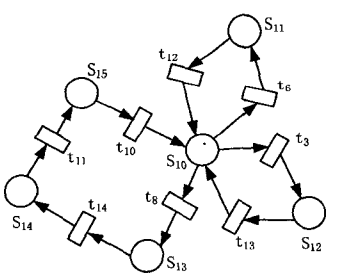
[15] Mesnier M, Ganger G R, Riedel E. Object-based storage [J]. IEEE Communications Magazine, 2003, 41(8): 84-90

(上接第 58 页)

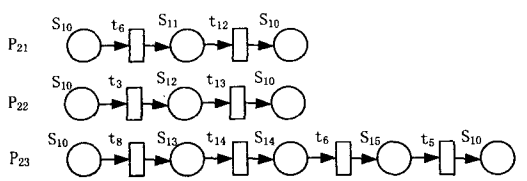


(b) Σ_1 的基本子进程集

图 5 有界 Petri 网 Σ_1 及其基本子进程集



(a) 有界 Petri 网 Σ_2



(b) Σ_2 的基本子进程集

图 6 有界 Petri 网 Σ_2 及其基本子进程集

结束语 文献[1]中提出了有界 Petri 网进程表达式的概念，即一个有界 Petri 网的进程表达式是以该网的基本子进程集为字母表的正规表达式。但在现有的文献中并没有给出有效求解基本子进程的方法。本文的贡献之一是提出了稳态网的概念并给出了一个算法，根据该算法可以求解有界稳态网的基本子进程集，然后根据所求得的基本子进程集来构造该网的进程表达式。本文的另外一个贡献是给出了由符合一定条件的 S-网的进程表达式来构造其同步合成网进程表达式的算法。进程表达式全面反映了网系统运行的各种可能，为 Petri 网分析提供了很好的途径。

参 考 文 献

[1] Wu Zhe-hui. Process expression of bounded Petri net[J]. Science in China(Series E), 1996, 39(1): 37-49

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

[3] 曾庆田，吴哲辉. Petri 网的进程网系统[J]. 计算机学报，2002，25(12)：1308-1315

[4] 吴哲辉. Petri 网导论[M]. 北京：机械工业出版社，2006

[5] Hopcroft J E, Ullman J D. Introduction to Automata Theory, Languages, and Computation [M]. Addison-Wesley Publishing Company, 1979

[6] Latteux M, Roos Y. Synchronized shuffle and regular languages [M]. Jewels are Forever-Contributions on Theoretical Computer Science in Honor of Arto Salomaa, Berlin: Springer-Verlag, 1999: 35-44

[7] 蒋昌俊. 离散事件动态系统的 PN 机理论[M]. 北京：科学出版社，2000