

一种基于同步合成构造 Petri 网进程表达式的方法

曾庆田

(山东科技大学信息科学与工程学院 山东 青岛 266510)

摘 要 Petri 网的进程是用于系统行为和状态描述的有效工具, Petri 网的进程表达式可以给出系统全部进程的描述, 但是对于任意无界 Petri 网而言求取其进程表达式十分困难. 文中首先考察结构简单的 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. 然后拓展了 Petri 网同步合成的概念, 分析了同步合成过程中基本进程段集之间的关系, 并利用同步混排给出了进程表达式之间的关系. 随后证明了一个 Petri 网可以通过一组 S-网同步合成得到, 利用 S-网的进程表达式给出了构造 Petri 网的进程表达式的方法.

关键词 Petri 网; S-网; 同步合成; 同步混排; 进程; 进程表达式

中图法分类号 TP301

A Construction Method for the Process Expression of a Petri Net Based on Synchronization Composition

ZENG Qing-Tian

(College of Information Science and Technology, Shandong University of Science and Technology, Qingdao, Shandong 266510)

Abstract Process is one of the most useful tools for property analysis of a Petri net, however it is usually difficult to present all the processes of a structure-complex Petri net. Process expression can be used to define the set of all processes for a Petri net. A construction method for the process expression of a Petri net is proposed based on synchronization composition in this paper. The process characteristics of S-Net, a kind of structure-simple Petri net, are analyzed firstly with details, and the approaches to obtain the process expressions of all kinds of S-Nets are obtained. Then, the relationship between the sets of the basic process section during the synchronization composition is analyzed, and the relationship between process expressions is expressed by synchronization shuffle operation of processes. It is proved that a Petri net can be constructed by the synchronization composition of a set of S-Nets, and the process expression of the original Petri net can be obtained based on the process expressions of these S-Nets.

Keywords Petri net; S-Net; synchronization composition; synchronization shuffle; 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. 文献[7]定义了有界 Petri 网的进程表达式并给出了具体求解方法. 文献[8]给出了无界公平 Petri 网的进程表达式的求取方法. 对于任意无界 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]. 文献[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 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^[6-9].

Petri 网的同步合成是研究复杂系统的一种有效方法^[10-13]. 文献[11]研究了同步合成网的进程特征, 考察了 Petri 网同步合成过程中, 基本进程语言的合成关系以及与之有关的线集、切集等性质. 本文在文献[11]已有的部分结果的基础上, 对同步合成 Petri 网的进程特性进行更深入的研究. 本文首先考察一类结构简单的 Petri 网——S-网的进程行为, 给出各种类型的 S-网的进程表达式的求取方法. 为了适应大规模系统分析的需要, 将 Petri 网同步合成的概念拓展到多个子网的情形, 并分析同步合成过程中基本进程段集之间的关系, 进而利用同步混排运算给出了进程表达式之间的关系. 文中证明了一个 Petri 网可以通过一组 S-网同步合成得到, 将求取 Petri 网进程表达式的问题转化成求取 S-网的进程表达式问题. 利用同步混排运算, 可以由 S-网的进程表达式给出 Petri 网的进程表达式, 由此给出利用同步合成构造 Petri 网的进程表达式的方法.

本文第 2 节介绍与本文讨论有关的 Petri 网进程的基本概念; 第 3 节首先分析 S-网的进程特性并给出 S-网的进程表达式的求取方法; 第 4 节拓展 Petri 网同步合成的概念, 并着重分析同步合成过程中基本进程段之间以及进程表达式之间满足的关系; 第 5 节证明一个 Petri 网可以通过一组 S-网同步合成得到, 给出 Petri 网进程表达式的求取方法; 第 6 节给出一个例子, 并将本文的方法与已有的工作作比较.

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^[1-4], 这里只对与本文讨论有关的基本概念、术语和记号做一下简述或约定. 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:

$$(1) \forall b \in B: |\cdot b| \leq 1 \wedge |b \cdot| \leq 1;$$

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

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:

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

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

$$(3) \forall e \in E: \varphi(e \cdot) = \varphi(e), \varphi(e \cdot) = \varphi(e) \cdot,$$

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:

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

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

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:

$$(1) \forall b_1, b_2 \in B (b_1 \neq b_2): \varphi(b_1) = \varphi(b_2) \rightarrow (\cdot b_1 \neq \cdot b_2 \vee \cdot b_1 = \cdot b_2 = \emptyset) \wedge (b_1 \cdot \neq b_2 \cdot \vee b_1 \cdot = b_2 \cdot = \emptyset);$$

$$(2) |\{b \mid \varphi(b) = s \wedge \cdot b = \emptyset\}| = M_0(s),$$

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, $u_1 \leq u_2$. 记:

$$(1) B_1 = \{x \in B \mid \exists b_1 \in u_1, b_2 \in u_2: (b_1, x) \in G^* \wedge (x, b_2) \in G^*\};$$

$$(2) E_1 = \{y \in E \mid \exists b_1 \in u_1, b_2 \in u_2: (b_1, y) \in G^+ \wedge (y, b_2) \in G^+\};$$

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

令 $N_1=(B_1, E_1; G_1)$, $\varphi_1: N_1 \rightarrow \Sigma$ 满足: $\forall x \in B_1 \cup E_1: \varphi_1(x) = \varphi(x)$, 则称 (N, φ_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 ($i, j \neq 1, 2$) 有: $u_i \neq u_j \rightarrow (\varphi(u_i) \neq \varphi(u_j)) \wedge (\varphi(u_i) \not\prec \varphi(u_j)) \wedge (\varphi(u_i) \not\succ \varphi(u_j))$, 则称 $P_1 = (N[u_1, u_2], \varphi)$ 是 Σ 的一个基本进程段.

Petri 网 Σ 的全体基本进程段的集合记为 $BP(\Sigma)$. 基本进程段之间的运算包括连接运算“ \circ ”、选择运算“ $+$ ”、“ $*$ ”-闭包、并发运算“ \parallel ”和“ α ”-闭包等运算, 具体参见文献[9], 在此不做具体介绍.

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-网的进程表达式

为了描述结构复杂 Petri 网的进程行为, 本节首先讨论一类结构简单的 Petri 网——S-网的进程特性, 并给出各类 S-网的进程表达式的求取方法. 通过后面的讨论, 我们可以看到 S-网的进程表达式是基于同步合成描述结构复杂 Petri 网进程行为的前提和基础.

定义 8^[1]. 一个 Petri 网 $\Sigma = (S, T; F, M_0)$ 称为 S-网当且仅当 $\forall t \in T: |t| \leq 1$ 并且 $|t'| \leq 1$.

定义 9. $\Sigma = (S, T; F, M_0)$ 为一个 S-网, $\forall t \in T$: 若 $|t| = 0$, 则称变迁 t 为 Σ 的源变迁; 若 $|t'| = 0$, 则称变迁 t 为 Σ 的汇变迁.

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 $e \in E, |e| \leq 1$ 且 $|e'| \leq 1$.

文献[6]给出了 Petri 网的进程网系统的概念. 简言之, 进程网系统实际上就是将原 Petri 网的每个基本进程段浓缩为一个变迁, 每个基本进程段的输入、输出库所集作为对应的变迁的输入、输出库所集, 建立相应的流关系, 将原网系统的初始标识向进程网的库所集上做投影. 关于进程网系统的定义以及相关的讨论可参见文献[6].

命题 2. 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, 但是在得到所有的进程后去掉所有的新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: $|t| \leq 1$ 且 $|t'| = 1$. 根据 S-网中是否含有源变迁以及初始标识是否为空, 可以将 S-网分为 4 类:

- (1) 不含源变迁且初始标识为空;
- (2) 不含源变迁且初始标识不为空;
- (3) 含源变迁且初始标识为空;
- (4) 含源变迁且初始标识不为空.

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, 故没有讨论的意义. 第(2)类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 3 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-网为初始标识为空的 S-网, 第(4)类 S-网为初始标识不为空的 S-网.

3.1 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 $Exp(P(\Sigma))$ 是一个正规表达式.

证明. 根据文献[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, 具体方法可参见文献[7].

例 1. 图 1(a) 给出了一个标识 S-图 Σ_1 , 可以求得 Σ_1 的 4 个基本进程段, 如图 1(b) 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, 关于进程段的具体分类可参见文献[6, 9]. 利用文献[7]的方法可以给出 Σ_1 的进程表达式为 $Exp(P(\Sigma_1)) = P_1(P_2)^*(P_3 + P_4)$.

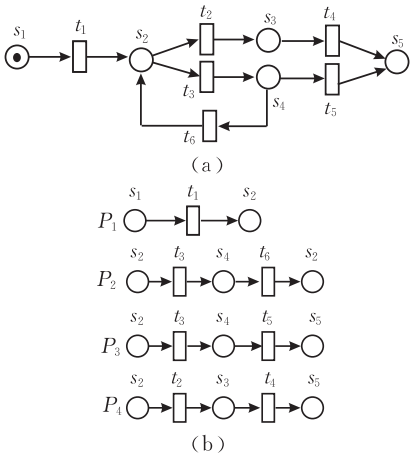


图 1 标识 S-图 Σ_1 及其 4 个基本进程段

3.2 初始标识为空的 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 RE 使得 Σ 的进程表达式 $Exp(P(\Sigma)) = (RE)^\alpha$ ①。

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

(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 的初始基本进程段, 因 $u_{11} = \emptyset$, 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 Σ_P 是一个只含有一个源变迁且初始标识为空的 S-网, 由文献[9]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 $Exp(P(\Sigma)) = (RE)^\alpha$ 。

证毕。

定理 2. 设 S-网 $\Sigma = (S, T; F, M_0)$ 中, $\forall s \in S: M_0(s) = 0$ 且存在 $t_1, t_2, \dots, t_k (k \geq 2)$ 为 Σ 的 k 个源变迁, 则存在正规表达式 RE_1, RE_2, \dots, RE_k 使得 Σ 的进程表达式 $Exp(P(\Sigma)) = (RE_1)^\alpha \parallel (RE_2)^\alpha \parallel \dots \parallel (RE_k)^\alpha$ 。

证明. 采用如下的构造法证明该定理的成立。令 $T_0 = \{t_1, t_2, \dots, t_k\}$ 为 Σ 的 k 个源变迁 ($k \geq 2$), 分别构造如下的 k 个 S-网 $\Sigma_i = (S_i, T_i; F_i, M_{0i}) (1 \leq i \leq k)$ 满足下面的条件:

(1) $S_i = S$, 其中 S 为 Σ 的库所集;

(2) $T_i = (T - T_0) \cup \{t_i\}$, 其中 T 为 Σ 的变迁集, $T_0 = \{t_1, t_2, \dots, t_k\}$ 为 Σ 的 k 个源变迁;

(3) $F_i = F - (T_0 - \{t_i\}) \times S$, 其中 F 为 Σ 的流关系集, $T_0 = \{t_1, t_2, \dots, t_k\}$ 为 Σ 的 k 个源变迁;

(4) $M_{0i} = M_0$, 其中 M_0 为 Σ 的初始标识。

由 $\Sigma_i (1 \leq i \leq k)$ 的构造过程易知, $Exp(P(\Sigma)) =$

$Exp(P(\Sigma_1)) \parallel Exp(P(\Sigma_2)) \parallel \dots \parallel Exp(P(\Sigma_k))$, 其中 \parallel 为进程表达式之间的并发运算。

由 Σ_i 的构造过程的(2)知, 每个 $\Sigma_i = (S, T_i; F_i, M_0) (1 \leq i \leq k)$ 含有且只含有一个源变迁。由 Σ_i 的构造过程的(4)知, 对于每个 $\Sigma_i = (S, T_i; F_i, M_0) (1 \leq i \leq k)$, $\forall s \in S: M_{0i}(s) = 0$ 。即每个 $\Sigma_i = (S, T_i; F_i, M_0) (1 \leq i \leq k)$ 均满足引理 3 的条件。由引理 3 知, 存在正规表达式 RE_i 使得 $Exp(P(\Sigma_i)) = (RE_i)^\alpha (1 \leq i \leq k)$ 。

故, 存在正规表达式 RE_1, RE_2, \dots, RE_k 使得 Σ 的进程表达式 $Exp(P(\Sigma)) = (RE_1)^\alpha \parallel (RE_2)^\alpha \parallel \dots \parallel (RE_k)^\alpha$ 。

证毕。

例 2. 图 2(a) 给出了一个初始标识为空的 S-网 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 2(b) 所示。可以求得 Σ_{21} 的 5 个基本进程段如图 2(c) 所示。

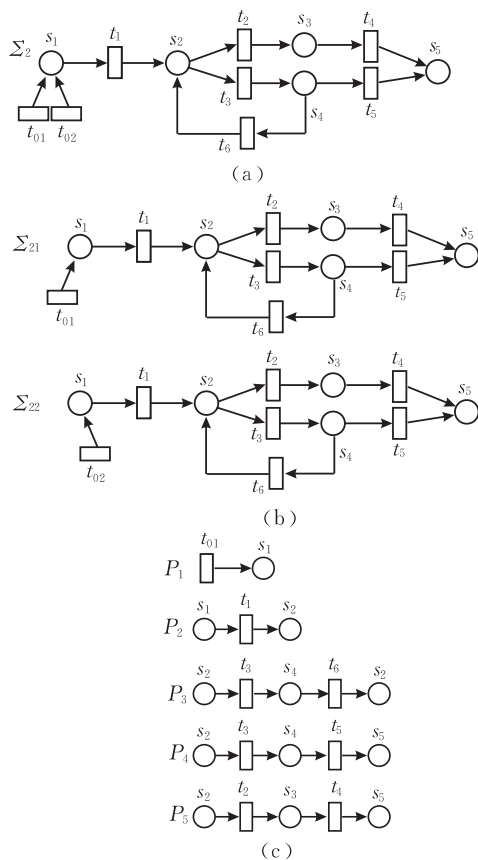


图 2 初始标识为空的 S-网 Σ_2 及其变形网 Σ_{21} 和 Σ_{22} 、基本进程段

① 参见文献[14]. 语言 L 的 α -闭包定义为 $L^\alpha = \bigcup_{i=0,1,\dots} L^{(i)}$, 其中 $L^{(2)} = L \parallel L$; $L^{(n)} = L \parallel L^{(n-1)} (n \geq 2)$. 如: $L = \{abc\}$, 则: $L^\alpha = \{w \mid \forall s \in Pre(w), \#(a, s) \geq \#(b, s) \geq \#(c, s) \wedge \#(a, w) = \#(b, w) = \#(c, w)\}$ (其中: $Pre(w)$ 表示 w 的前缀, $\#(a, w)$ 表示 a 在 w 中出现的次数)。

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 5 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 2(c) 中 P_1 的 t_{01} 改为 t_{02} , 其它 P_i 均为图 2(c) 中的 P_i . 可以求得 Σ_{21} 和 Σ_{22} 的进程表达式分别为: $Exp(P(\Sigma_{21})) = (P_1 P_2 (P_3)^* (P_4 + P_5))^\alpha$ 和 $Exp(P(\Sigma_{22})) = (P'_1 P_2 (P_3)^* (P_4 + P_5))^\alpha$. 则 Σ_2 的进程表达式为

$$Exp(P(\Sigma_2)) = Exp(P(\Sigma_{21})) \parallel Exp(P(\Sigma_{22})) = (P_1 P_2 (P_3)^* (P_4 + P_5))^\alpha \parallel (P'_1 P_2 (P_3)^* (P_4 + P_5))^\alpha.$$

3.3 初始标识不为空的 S-网的进程表达式

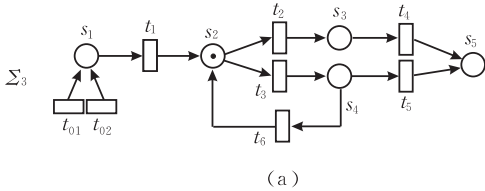
定理 3. 设 S-网 $\Sigma = (S, T; F, M_0)$ 中, 存在 $t_1, t_2, \dots, t_k (k \geq 1)$ 为 Σ 的 k 个源变迁, 且存在 $s \in S: M_0(s) \neq 0$, 则存在正规表达式 RE_1, RE_2, \dots, RE_k 和 RE_{k+1} 使得 Σ 的进程表达式 $Exp(P(\Sigma)) = (RE_1)^\alpha \parallel (RE_2)^\alpha \parallel \dots \parallel (RE_k)^\alpha \parallel RE_{k+1}$.

证明. 采用如下的方法证明该定理成立. 令 $T_0 = \{t_1, t_2, \dots, t_k\}$, 构造两个 S-网 $\Sigma_i = (S_i, T_i; F_i, M_{0i}) (i=1,2)$, 其中:

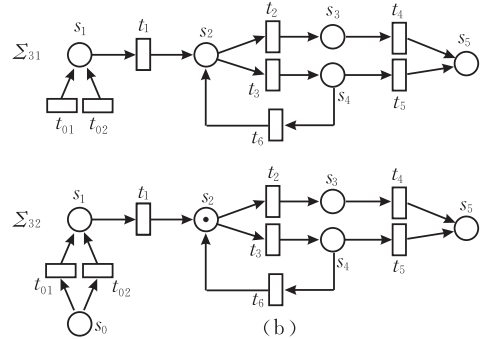
$$(1) S_1 = S; T_1 = T; F_1 = F; \forall s \in S: M_{01}(s) = 0.$$

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

$$M_{02} = M_0.$$



(a)



(b)

图 3 含初始标识的 S-网 Σ_3 及其变形网 Σ_{31} 和 Σ_{32}

4 同步合成 Petri 网的进程特性分析

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^[10-13], 文献[10-12]仅给出了两个网系统的同步合成运算, 在分析大规模网系统时, 可能需要多个 (≥ 2 个) 网系统之间的同步合成运算. 为此, 我们首先将网系统的同步合成运算推广到 $k (k \geq 2)$ 的情况^[13].

定义 10^[13]. 设 Petri 网 $\Sigma_i = (S_i, T_i; F_i, M_{0i}) (i=1,2,\dots,k)$, 令 $\Sigma = (S, T; F, M_0)$ 使得:

$$(1) S = \bigcup_{i=1}^k S_i \text{ 且 } \forall i, j, i \neq j: S_i \cap S_j = \emptyset;$$

易证, $Exp(P(\Sigma)) = Exp(P(\Sigma_1)) \parallel Exp(P(\Sigma_2))$. 同时可证, $\Sigma_1 = (S_1, T_1; F_1, M_{01})$ 为含有 k 个源变迁但初始标识为空的 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, RE_2, \dots, RE_k 使得 Σ_1 的进程表达式 $Exp(P(\Sigma_1)) = (RE_1)^\alpha \parallel (RE_2)^\alpha \parallel \dots \parallel (RE_k)^\alpha$. 根据定理 1 知, $Exp(P(\Sigma_2))$ 是一个正规表达式. 所以, 存在正规表达式 RE_1, RE_2, \dots, RE_k 和 RE_{k+1} 使得 Σ 的进程表达式 $Exp(P(\Sigma)) = (RE_1)^\alpha \parallel (RE_2)^\alpha \parallel \dots \parallel (RE_k)^\alpha \parallel RE_{k+1}$. 证毕.

例 3. 图 3(a) 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 3(b) 所示, 容易证明 $Exp(P(\Sigma_3)) = Exp(P(\Sigma_{31})) \parallel Exp(P(\Sigma_{32}))$. Σ_{31} 即为例 2 中的 Σ_2 , 故 $Exp(P(\Sigma_{31})) = (P_1 P_2 (P_3)^* (P_4 + P_5))^\alpha \parallel (P'_1 P_2 (P_3)^* (P_4 + P_5))^\alpha$, 其中 $P'_1, P_i (i=2,3,4,5)$ 分别如图 2(c) 所示. 容易求得 Σ_{32} 3 个基本进程段就是图 2(c) 中的 P_3, P_4 和 P_5 , 并且可以求得 $Exp(P(\Sigma_{32})) = P_3^* (P_4 + P_5)$. 故 $Exp(P(\Sigma_3)) = Exp(P(\Sigma_{31})) \parallel Exp(P(\Sigma_{32})) = (P_1 P_2 (P_3)^* (P_4 + P_5))^\alpha \parallel (P'_1 P_2 (P_3)^* (P_4 + P_5))^\alpha \parallel P_3^* (P_4 + P_5)$.

(2) $T = \bigcup_{i=1}^k T_i$ 且 $\forall i$, 至少存在一个 $j \in \{1, 2, \dots, k\}$ 使得 $T_i \cap T_j \neq \emptyset$;

$$(3) F = \bigcup_{i=1}^k F_i;$$

(4) 当 $s \in S_i (i=1,2,\dots,k): M_0(s) = M_{0i}(s)$, 则称 Σ 为 k 个 Petri 网 $\Sigma_1, \Sigma_2, \dots, \Sigma_k$ 的同步合成网, 记作 $\Sigma = \Xi_{1 \leq i \leq k} \Sigma_i$.

可以证明, 定义 10 是对文献[10-12]所给出的同步合成定义的扩展. 当 $k=2$ 时, 定义 10 即文献[10-12]所给出的同步合成的定义.

文献[11]分析了同步合成网的进程特征, 着力考察了 Petri 网同步合成过程中, 基本进程语言的

合成关系以及与之有关的线集、切集等性质. 我们在文献[11]已有的部分结果的基础上, 对同步合成 Petri 网的进程特性进行更深入的研究. 首先分析 Petri 网同步合成过程中基本进程段集之间的关系, 并在此基础上着重讨论进程表达式之间的关系, 由此给出利用同步合成构造 Petri 网的进程表达式的方法.

4.1 基本进程段特性分析

引理 4^[11]. Petri 网 Σ 为 Σ_1 与 Σ_2 的同步合成网, $BP(*)$ 表示 $*$ 的基本进程段集合, 其中 $*$ $\in \{\Sigma, \Sigma_1, \Sigma_2\}$, 则: $BP(\Sigma) = \{\varphi_1(N_1) \Xi \varphi_2(N_2) \mid \varphi_i(N_i) \in BP(\Sigma_i) \wedge i=1, 2\}$.

证明. 见文献[11]的定理 1.

定理 4. 若 $\Sigma = \Xi_{1 \leq i \leq k} \Sigma_i (i=1, 2, \dots, k)$, 则 $BP(\Sigma) = \{\Xi_{1 \leq i \leq k} \varphi_i(N_i) \mid \varphi_i(N_i) \in BP(\Sigma_i) \wedge (1 \leq i \leq k)\}$.

证明. 在引理 4 的基础上, 利用归纳法对 k 进行归纳, 可以证明定理成立.

定义 11. 设 $\Sigma = (S, T; F, M_0)$ 为 $\Sigma_i = (S_i, T_i; F_i, M_{0i}) (i=1, 2, \dots, k)$ 的同步合成网, $SN(*)$ 是 $*$ 的出现网集合, 其中 $*$ $\in \{\Sigma, \Sigma_1, \dots, \Sigma_k\}$. 若 $\Pi: \varphi(SN(\Sigma)) \rightarrow \varphi_i(SN(\Sigma_i))$, 使得对 $\forall N \in SN(\Sigma)$ 满足:

(1) $\forall b \in B$, 若 $\varphi(b) \in S_j (j \neq i)$, 则删去 b 及其输入和输出弧;

(2) $\forall e \in E$, 若 $\varphi(e) \in E_j - E_i (j \neq i)$, 则删去 e 及其输入和输出弧,

则称 Π 为 $\varphi(SN(\Sigma))$ 到 $\varphi_i(SN(\Sigma_i))$ 的投影映射. 在不会引起混淆的情况下, 直接简记为 $\Pi_{\Sigma \rightarrow \Sigma_i}$.

Petri 网的基本进程段只是出现网的一部分. 如果将定义 11 中 $SN(*)$ 扩展为出现网集合和基本进程段集合, 其它条件不变, 根据投影运算可以得到下面的结论.

引理 5. Petri 网 Σ 为 Σ_1 与 Σ_2 的同步合成网, $BP(*)$ 表示 $*$ 的基本进程段集合, 其中 $*$ $\in \{\Sigma, \Sigma_1, \Sigma_2\}$, $\forall BP \in BP(\Sigma)$, 则 $\Pi_{\Sigma \rightarrow \Sigma_i} BP \in BP(\Sigma_i)$.

证明. 根据引理 4 和定义 11 容易证明.

定理 5. 若 $\Sigma = \Xi_{1 \leq i \leq k} \Sigma_i (i=1, 2, \dots, k)$, $BP(*)$ 表示 $*$ 的基本进程段集合, 其中 $*$ $\in \{\Sigma, \Sigma_1, \dots, \Sigma_k\}$, $\forall BP \in BP(\Sigma)$, 则 $\Pi_{\Sigma \rightarrow \Sigma_i} BP \in BP(\Sigma_i)$.

证明. 根据引理 5 容易证明.

4.2 进程表达式关系分析

为了描述 Petri 网同步合成过程中进程表达式之间满足的关系, 本节首先给出同步混排的概念. 为

了表述方便, 首先引入几个记号. 若 X 为任意字母表, X 上的任意字符 $a \in X$, 字符串 $\omega \in X^*$, 记 $\#(a, \omega)$ 为字符 a 在 ω 中出现的次数. X 上的任意语言 $L \subseteq X^*$, 记 $\delta(L) = \{x \in X \mid \exists \omega \in L: \#(x, \omega) > 0\}$. 例如 $\delta(a+ab) = \delta(aa+abbb) = \{a, b\}$. 易知 $\delta(L) \subseteq X$.

定义 12. $\Sigma_i = (S_i, T_i; F_i, M_{0i}) (i \in \{1, 2\})$ 为两个 Petri 网, $Exp(P(\Sigma_i))$ 为 Σ_i 的进程表达式, $Exp(P(\Sigma_1)) \Theta Exp(P(\Sigma_2))$ 为 $Exp(P(\Sigma_1))$ 和 $Exp(P(\Sigma_2))$ 的同步混排, 当且仅当:

$$\begin{aligned} Exp(P(\Sigma_1)) \Theta Exp(P(\Sigma_2)) = \\ \{\omega \in \delta(Exp(P(\Sigma_\Delta)))^* \mid \delta(Exp(P(\Sigma_\Delta))) = \\ \delta(Exp(P(\Sigma_1))) \cup \delta(Exp(P(\Sigma_2)))\}, \\ \Pi_{\Sigma_\Delta \rightarrow \Sigma_i}(\omega) \in Exp(P(\Sigma_i))\}. \end{aligned}$$

将该定义可以扩展为 $k (k \geq 2)$ 个表达式的同步混排, 记为 $\Theta_{1 \leq i \leq k} Exp(P(\Sigma_i))$:

$$\begin{aligned} \Theta_{1 \leq i \leq k} Exp(P(\Sigma_i)) = \\ Exp(P(\Sigma_1)) \Theta Exp(P(\Sigma_2)) \Theta \dots \Theta Exp(P(\Sigma_k)) = \\ ((Exp(P(\Sigma_1)) \Theta Exp(P(\Sigma_2))) \Theta \dots \Theta Exp(P(\Sigma_k))). \end{aligned}$$

引理 6. 若 $\Sigma = (S, T; F, M_0)$ 为 $\Sigma_i = (S_i, T_i; F_i, M_{0i}) (i \in \{1, 2\})$ 的同步合成网, 则 $Exp(P(\Sigma)) = Exp(P(\Sigma_1)) \Theta Exp(P(\Sigma_2))$.

证明. 首先证明 $Exp(P(\Sigma)) \subseteq Exp(P(\Sigma_1)) \Theta Exp(P(\Sigma_2))$, 然后证明 $Exp(P(\Sigma_1)) \Theta Exp(P(\Sigma_2)) \subseteq Exp(P(\Sigma))$.

先证 $Exp(P(\Sigma)) \subseteq Exp(P(\Sigma_1)) \Theta Exp(P(\Sigma_2))$.

$\forall \varphi(N) \in Exp(P(\Sigma))$, 令 $N = (B, E; G)$. 由于 $\varphi(N)$ 是 Σ 的进程且根据 $\Pi_{\Sigma \rightarrow \Sigma_i}$ 的定义知:

(1) $\Pi_{\Sigma \rightarrow \Sigma_i}(\varphi(B)) \subseteq S_i \wedge \Pi_{\Sigma \rightarrow \Sigma_i}(\varphi(E)) \subseteq T_i \wedge \Pi_{\Sigma \rightarrow \Sigma_i}(\varphi_i(G)) \subseteq F_i$;

(2) $\forall e \in \Pi_{\Sigma \rightarrow \Sigma_i}(E) \subseteq E$: 由 $\varphi(e) = \varphi(e) \wedge \varphi(e) = \varphi(e)$ 得 $\varphi(e) \cap S_i = \varphi(e) \cap S_i \wedge \varphi(e) \cap S_i = \varphi(e) \cap S_i$, $\Pi_{\Sigma \rightarrow \Sigma_i} \varphi(\cdot \Pi_{\Sigma \rightarrow \Sigma_i}(e)) = (\Pi_{\Sigma \rightarrow \Sigma_i} \varphi(\Pi_{\Sigma \rightarrow \Sigma_i}(e))) \wedge \Pi_{\Sigma \rightarrow \Sigma_i} \varphi(\Pi_{\Sigma \rightarrow \Sigma_i}(e)) = (\Pi_{\Sigma \rightarrow \Sigma_i} \varphi(\Pi_{\Sigma \rightarrow \Sigma_i}(e)))$;

(3) $\forall b_1, b_2 \in \Pi_{\Sigma \rightarrow \Sigma_i}(B) \subseteq B$: 由 $\varphi(b_1) = \varphi(b_2)$ 得 $b_1 \neq b_2 \wedge b_1 \neq b_2$, 则 $\varphi(b_1) \cap S_i = \varphi(b_2) \cap S_i$ 且 $b_1 \cap T_i \neq b_2 \cap T_i \wedge b_1 \cap T_i \neq b_2 \cap T_i$, 则由 $\Pi_{\Sigma \rightarrow \Sigma_i} \varphi(\Pi_{\Sigma \rightarrow \Sigma_i}(b_1)) = \Pi_{\Sigma \rightarrow \Sigma_i} \varphi(\Pi_{\Sigma \rightarrow \Sigma_i}(b_2))$ 得 $\Pi_{\Sigma \rightarrow \Sigma_i}(b_1) \neq \Pi_{\Sigma \rightarrow \Sigma_i}(b_2) \wedge \Pi_{\Sigma \rightarrow \Sigma_i}(b_1) \neq \Pi_{\Sigma \rightarrow \Sigma_i}(b_2)$;

(4) $\forall s \in S_i \subseteq S$: $|\{b \mid b \neq \emptyset \wedge \varphi(b) = s\}| = M_0(s)$, 则 $|\{\Pi_{\Sigma \rightarrow \Sigma_i}(b) \mid \Pi_{\Sigma \rightarrow \Sigma_i}(b) \neq \emptyset \wedge \Pi_{\Sigma \rightarrow \Sigma_i} \varphi(b) = s\}| = M_{0i}(s)$. 其中 $M_{0i} = \Gamma_{S \rightarrow S_i}(M_0)$, 即 M_0 在 S_i 上的投影向量,

这样 $\Pi_{\Sigma \rightarrow \Sigma_i} \varphi(N) \in Exp(P(\Sigma_i)) (i \in \{1, 2\})$, 根据 Θ

的定义, $\varphi(N) \in Exp(P(\Sigma_1)) \Theta Exp(P(\Sigma_2))$ 即 $Exp(P(\Sigma)) \subseteq Exp(P(\Sigma_1)) \Theta Exp(P(\Sigma_2))$.

再证明 $Exp(P(\Sigma_1)) \Theta Exp(P(\Sigma_2)) \subseteq Exp(P(\Sigma))$.

$\forall \varphi(N) \in Exp(P(\Sigma_1)) \Theta Exp(P(\Sigma_2))$, 记 $\varphi(N)$ 所包含的基本进程个数为 $|\varphi(N)|$, 下面对 $|\varphi(N)|$ 做归纳证明.

(1) 当 $|\varphi(N)| = 1$ 时, 此时 $\varphi(N)$ 是一个基本进程段且 $\Pi_{\Sigma \rightarrow \Sigma_i}(\varphi(N)) \in BP(\Sigma_i)$, 据引理 4 知 $\varphi(N) \in BP(\Sigma) \subseteq Exp(P(\Sigma))$, 因此 $Exp(P(\Sigma_1)) \Theta Exp(P(\Sigma_2)) \subseteq Exp(P(\Sigma))$.

(2) 假设 $|\varphi(N)| \leq k$ 时, 结论成立, 即 $Exp(P(\Sigma_1)) \Theta Exp(P(\Sigma_2)) \subseteq Exp(P(\Sigma))$. 下面我们考虑 $\varphi(N) \in Exp(P(\Sigma_1)) \Theta Exp(P(\Sigma_2))$ 且 $|\varphi(N)| = k+1$ 的情况. 根据进程表达式的定义知, 存在 $\varphi(N) \in Exp(P(\Sigma_1)) \Theta Exp(P(\Sigma_2))$ 和 $\overline{\varphi(N)}$ 使得 $\overline{\varphi(N)} = \varphi(N) \circ \overline{\varphi(N)}$, 其中 \circ 为进程段之间的连接运算, $|\varphi(N)| = k$ 且 $|\overline{\varphi(N)}| = 1$. 下面对 $\overline{\varphi(N)}$ 分情况讨论.

第 1 种情况. 若 $\overline{\varphi(N)} \in BP(\Sigma_i) - BP(\Sigma_j)$, $i, j \in \{1, 2\}$ 且 $i \neq j$. 则

(1.1) $\overline{\varphi(B)} = \varphi(B) \circ \overline{\varphi(B)}$, 因为 $\varphi(B) \circ \overline{\varphi(B)} \subseteq S_1 \cup S_2 \cup S_i$, 故 $\overline{\varphi(B)} \subseteq S_1 \cup S_2 \cup S_i$, 即 $\overline{\varphi(B)} \subseteq S_1 \cup S_2$;

$\overline{\varphi(E)} = \varphi(E) \circ \overline{\varphi(E)}$, 因为 $\varphi(E) \circ \overline{\varphi(E)} \subseteq T_1 \cup T_2 \cup T_i$, 故 $\overline{\varphi(E)} \subseteq T_1 \cup T_2 \cup T_i$, 即 $\overline{\varphi(E)} \subseteq T_1 \cup T_2$;

$\overline{\varphi(G)} = \varphi(G) \circ \overline{\varphi(G)}$, 因为 $\varphi(G) \circ \overline{\varphi(G)} \subseteq F_1 \cup F_2 \cup F_i$, 故 $\overline{\varphi(G)} \subseteq F_1 \cup F_2 \cup F_i$, 即 $\overline{\varphi(G)} \subseteq F_1 \cup F_2$.

(1.2) $\forall \hat{e} \in \hat{E}$, 其中 $\hat{E} = E \cup \overline{E}$, 当 $\hat{e} \in E$ 时, 根据归纳假设知 $\varphi(\cdot \hat{e}) = \cdot \varphi(\hat{e})$ 且 $\varphi(\hat{e} \cdot) = \varphi(\hat{e}) \cdot$. 当 $\hat{e} \in \overline{E}$ 时, 因为 $\overline{\varphi(N)} \in BP(\Sigma_i) - BP(\Sigma_j)$, 所以 $\overline{\varphi}(\cdot \hat{e}) = \cdot \overline{\varphi}(\hat{e})$ 且 $\overline{\varphi}(\hat{e} \cdot) = \overline{\varphi}(\hat{e}) \cdot$. 故 $\varphi(\cdot \hat{e}) = \cdot \varphi(\hat{e})$ 且 $\varphi(\hat{e} \cdot) = \varphi(\hat{e}) \cdot$.

(1.3) $\forall \hat{b}_1, \hat{b}_2 \in \hat{B}$, 其中 $\hat{B} = B \cup \overline{B}$, 类似与 (1.2) 可以证明 $\overline{\varphi(b_1)} = \overline{\varphi(b_2)} \Rightarrow \cdot \hat{b}_1 \neq \cdot \hat{b}_2 \wedge \hat{b}_1 \neq \hat{b}_2 \cdot$.

(1.4) $\forall s \in S$, 其中 $S = S_1 \cup S_2$, $|\{b \mid b = \emptyset \wedge \varphi(b) = s\}| = M_0(s)$, 也即 $|\{b \mid \cdot \hat{b} = \emptyset \wedge \overline{\varphi}(\hat{b}) = s\}| = M_0(s)$.

综合 (1.1) ~ (1.4) 知, $\overline{\varphi(N)} \in Exp(P(\Sigma))$. 从而, $Exp(P(\Sigma_1)) \Theta Exp(P(\Sigma_2)) \subseteq Exp(P(\Sigma))$.

第 2 种情况. 若 $\overline{\varphi(N)} \in BP(\Sigma_1) \cap BP(\Sigma_2)$, 则 $\overline{\varphi(B)} = \varphi(B) \circ \overline{\varphi(B)}$, 因为 $\varphi(B) \circ \overline{\varphi(B)} \subseteq S_1 \cup S_2 \cup S_1 \cup S_2$, 故 $\overline{\varphi(B)} \subseteq S_1 \cup S_2 \cup S_1 \cup S_2$, 即 $\overline{\varphi(B)} \subseteq$

$S_1 \cup S_2$;

$\overline{\varphi(E)} = \varphi(E) \circ \overline{\varphi(E)}$, 因为 $\varphi(E) \circ \overline{\varphi(E)} \subseteq T_1 \cup T_2 \cup T_1 \cup T_2$, 故 $\overline{\varphi(E)} \subseteq T_1 \cup T_2 \cup T_1 \cup T_2$, 即 $\overline{\varphi(E)} \subseteq T_1 \cup T_2$;

$\overline{\varphi(G)} = \varphi(G) \circ \overline{\varphi(G)}$, 因为 $\varphi(G) \circ \overline{\varphi(G)} \subseteq F_1 \cup F_2 \cup F_1 \cup F_2$, 故 $\overline{\varphi(G)} \subseteq F_1 \cup F_2 \cup F_1 \cup F_2$, 即 $\overline{\varphi(G)} \subseteq F_1 \cup F_2$.

其它元素的证明类似于第 1 种情况中的证明. 因此, $\overline{\varphi(N)} \in Exp(P(\Sigma))$, 也即 $Exp(P(\Sigma_1)) \Theta Exp(P(\Sigma_2)) \subseteq Exp(P(\Sigma))$.

综合两种情况, 由归纳假设知 $Exp(P(\Sigma_1)) \Theta Exp(P(\Sigma_2)) \subseteq Exp(P(\Sigma))$.

由 $Exp(P(\Sigma)) \subseteq Exp(P(\Sigma_1)) \Theta Exp(P(\Sigma_2))$ 且 $Exp(P(\Sigma_1)) \Theta Exp(P(\Sigma_2)) \subseteq Exp(P(\Sigma))$, 故:

$$Exp(P(\Sigma)) = Exp(P(\Sigma_1)) \Theta Exp(P(\Sigma_2)).$$

证毕.

定理 6. 若 $\Sigma = \bigwedge_{1 \leq i \leq k} \Sigma_i$, 则 $Exp(P(\Sigma)) = \bigwedge_{1 \leq i \leq k} Exp(P(\Sigma_i))$.

证明. 在引理 6 的基础上, 利用归纳法对 k 进行归纳, 可以证明定理成立.

定理 6 表明, 如果一个 Petri 网 Σ 是另外几个 Petri 网 $\Sigma_1, \Sigma_2, \dots, \Sigma_k$ 的同步合成网, 则 Σ 的进程表达式 $Exp(P(\Sigma))$ 可以通过 $\Sigma_1, \Sigma_2, \dots, \Sigma_k$ 的进程表达式的同步混排得到. 如果能够找到满足条件的这样一组 Petri 网 $\Sigma_1, \Sigma_2, \dots, \Sigma_k$, 并且 $\Sigma_1, \Sigma_2, \dots, \Sigma_k$ 的进程表达式是可求的, 那么我们就可以给出 Petri 网 Σ 的进程表达式.

5 求取 Petri 网的进程表达式的构造算法

第 3 节分析了 S-网的进程特性, 并给出了各类 S-网的进程表达式的求取方法. 也就是说, S-网的进程表达式是可求的. 下面证明任给 Petri 网 Σ 都可由一组 S-网 $\Sigma_1, \Sigma_2, \dots, \Sigma_k$ 通过同步合成得到, 这里假定 Σ 本身不是 S-网, 若 Σ 是 S-网, 则 Σ 的语言本身就是可求的. 同时假定 Σ 是有限的、连通的.

定理 7. $\Sigma = (S, T; F, M_0)$ 是一个 Petri 网, 存在一组 S-网 $\Sigma_1, \Sigma_2, \dots, \Sigma_k (k \geq 2)$ 满足 $\Sigma = \bigwedge_{1 \leq i \leq k} \Sigma_i$.

证明. 通过下面的构造过程来证明该定理成立.

1. 定义函数 $f: S \rightarrow \{1, 2, \dots, k\}$ 满足 $\forall s_1, s_2 \in S: (s_1' \cap s_2' \neq \emptyset) \vee (\cdot s_1 \cap \cdot s_2 \neq \emptyset) \rightarrow f(s_1) \neq f(s_2)$;

2. 根据函数 f , 将 Σ 分解成 $\Sigma_1, \Sigma_2, \dots, \Sigma_k$, 其中 $\Sigma_i = (S_i, T_i; F_i, M_{0i})$ 满足:

- 2.1. $S_i = \{s \in S \mid f(s) = i\}$;
- 2.2. $T_i = \{t \in T \mid \exists s \in S_i, t \in \cdot s \cup s \cdot\}$;
- 2.3. $F_i = \{(S_j \times T_j) \cup (T_j \times S_j)\} \cap F$;
- 2.4. $M_{0i} = \Gamma_{S \rightarrow S_i}(M_0)$;
3. 容易证明 $\Sigma_i = (S_i, T_i; F_i, M_{0i}) (i \in \{1, 2, \dots, k\})$ 是一组 S-网;

4. 可以证明 $\Sigma_i = (S_i, T_i; F_i, M_{0i}) (i \in \{1, 2, \dots, k\})$ 满足下面的条件:

- 4.1. 任意 $i, j \in \{1, 2, \dots, k\}, i \neq j, S_i \cap S_j = \emptyset, \bigcup_{i=1}^k S_i = S$.

4.2. 若 $k > 1$, 则 $\forall i \in \{1, 2, \dots, k\}, \exists j \in \{1, 2, \dots, k\}, i \neq j$ 满足: $T_i \cap T_j \neq \emptyset$, 且 $\bigcup_{i=1}^k T_i = T$.

由定义 10, $\Sigma = \bigoplus_{1 \leq i \leq k} \Sigma_i$. 证毕.

定理 7 说明任给 Petri 网 Σ , 都可找到一组 S-网 $\Sigma_1, \Sigma_2, \dots, \Sigma_k$, 使得这组 S-网同步合成的结果是 Σ . 定理 7 的证明过程实际上给出了寻找 $\Sigma_1, \Sigma_2, \dots, \Sigma_k$ 的方法, 执行完步 1 和步 2 后, 输出 $\Sigma_1, \Sigma_2, \dots, \Sigma_k$ 即可, 这里就不再重复给出这个具体算法, 关于这个算法的具体构造过程和相关的讨论可参见文献[13], 算法的时间复杂性主要取决于 $|S|, |s|$ 以及 $|s'|$ 的大小. 定理 7 具有重要意义, 为利用“自底向上”的同步合成方法分析复杂系统提供了寻找结构简单子网的方法.

定理 8. $\Sigma = (S, T; F, M_0)$ 是一个 Petri 网, 存在一组 S-网 $\Sigma_1, \Sigma_2, \dots, \Sigma_k (k \geq 2)$ 满足 $Exp(P(\Sigma)) = \bigoplus_{1 \leq i \leq k} Exp(P(\Sigma_i))$.

证明. 由定理 7, 存在一组 S-网 $\Sigma_1, \Sigma_2, \dots, \Sigma_k (k \geq 2)$ 满足 $\Sigma = \bigoplus_{1 \leq i \leq k} \Sigma_i$. 由定理 6 知, $Exp(P(\Sigma)) = \bigoplus_{1 \leq i \leq k} Exp(P(\Sigma_i))$.

给定一个 Petri 网 Σ , 根据定理 7, 我们可以找到一组 S-网 $\Sigma_1, \Sigma_2, \dots, \Sigma_k$ 满足 $\Sigma = \bigoplus_{1 \leq i \leq k} \Sigma_i$. 根据第 3 节的方法可以求得 $\Sigma_1, \Sigma_2, \dots, \Sigma_k$ 的进程表达式. 根据定理 8, $\bigoplus_{1 \leq i \leq k} Exp(P(\Sigma_i))$ 即为 Σ 的进程表达式.

至此, 我们得到了一种基于同步合成构造 Petri 网的进程表达式的方法, 具体过程由算法 1 给出.

算法 1. Petri 网的进程表达式的求取算法.

输入: Petri 网 $\Sigma = (S, T; F, M_0)$

输出: Σ 的进程表达式 $Exp(P(\Sigma))$

1. 寻找一组 S-网 $\Sigma_1, \Sigma_2, \dots, \Sigma_k$, 满足同步合成的结果恰好为 Σ , 具体方法为

1.1. 定义函数 $f: S \rightarrow \{1, 2, \dots, k\}$ 满足 $\forall s_1, s_2 \in S: (s_1' \cap s_2' \neq \emptyset) \vee (\cdot s_1 \cap \cdot s_2 \neq \emptyset) \rightarrow f(s_1) \neq f(s_2)$;

1.2. 根据函数 f , 将 Σ 分解成 $\Sigma_1, \Sigma_2, \dots, \Sigma_k$, 每个 $\Sigma_i = (S_i, T_i; F_i, M_{0i})$ 满足: ① $S_i = \{s \in S \mid f(s) = i\}$; ② $T_i = \{t \in T \mid \exists s \in S_i, t \in \cdot s \cup s \cdot\}$; ③ $F_i = \{(S_j \times T_j) \cup (T_j \times S_j)\} \cap F$; ④ $M_{0i} = \Gamma_{S \rightarrow S_i} M_0$;

1.3. $\Sigma_i = (S_i, T_i; F_i, M_{0i}) (i \in \{1, 2, \dots, k\})$ 即为所求的一组 S-网;

2. 根据第 3 节的方法, 分别求得各个 S-网的进程表达式 $Exp(P(\Sigma_i)) (i \in \{1, 2, \dots, k\})$;

3. $Exp(P(\Sigma)) \leftarrow \bigoplus_{1 \leq i \leq k} Exp(P(\Sigma_i))$ 输出, 算法结束.

算法的正确性和终止性可由前面的分析保证. 算法 1 给出 Petri 网的进程表达式的求解方法, 对于 Petri 网尤其是结构复杂的网系统的进程行为描述具有重要意义. 该算法采用了“自顶向下”和“自底向上”相结合的方法, 首先依据“自顶向下”的原则首先找到一组结构简单的子网, 然后依据“自底向上”的原则, 利用这组子网的进程表达式描述原网系统的进程行为.

6 例子与讨论

本节给出一个实例说明本文方法的具体过程及方法可行性, 并给出本文的方法与已有的 Petri 网进程表达式求取方法的比较.

例 4. Petri 网已经广泛地应用于 FMS(Flexible Manufacture System)的建模与分析, 这里不再讨论具体的建模方法. 图 4(a) 给出了一个两条生产线 FM_A 和 FM_B 组成的 FMS 的 Petri 网模型 Σ_{FMS} ,

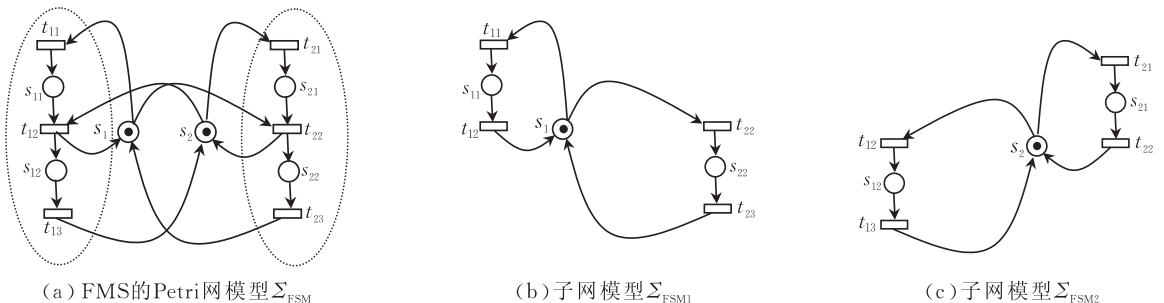


图 4 一个 FMS 的 Petri 网模型及其子网

FM_A 和 FM_B 共用 2 台机床 M_1 和 M_2 , FM_A 利用 M_1 进行第 1 道加工工序, 完成后释放 M_1 , 同时启动 M_2 进行第 2 道加工工序, 完成后释放 M_2 . FM_B 则利用 M_2 进行第 1 道加工工序, 完成后释放 M_2 , 同时启动 M_1 进行第 2 道加工工序, 完成后释放 M_1 . 在图 1 中, 两个虚椭圆分别标记了 2 条生产线 FM_A 和 FM_B 对应的 Petri 网模型, 库所 s_1 和 s_2 表示 2 台机床 M_1 和 M_2 , 含有托肯时, 表示机床可用.

下面利用本文所给的方法求取 FMS 对应的 Petri 网模型的进程表达式. 因为 $s_{12}, s_1 \in t_{12}, s_{21}, s_1 \in t_{22}, s_{22}, s_2 \in t_{22}$ 和 $s_{11}, s_2 \in t_{12}$, 所以可以定义函数 $f: f(s_{11}) = f(s_1) = f(s_{22}) = 1, f(s_{12}) = f(s_2) = f(s_{21}) = 2$. 根据 f 将 Σ_{FMS} 分解成 Σ_{FMS1} 和 Σ_{FMS2} , 分别如图 4 的 (b) 和 (c) 所示. 可以验证 Σ_{FMS1} 和 Σ_{FMS2} 均为 S-网且 $\Sigma_{FMS} = \Sigma_{FMS1} \Xi \Sigma_{FMS2}$. 可以分别求得 Σ_{FMS1} 和 Σ_{FMS2} 的基本进程段 P_{11}, P_{12}, P_{21} 和 P_{22} , 如图 5 的 (a) 和 (b) 所示.

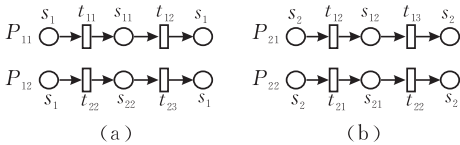


图 5 Σ_{FMS1} 和 Σ_{FMS2} 的基本进程段集

根据第 3 节的方法, 可以分别求得 Σ_{FMS1} 和 Σ_{FMS2} 的进程表达式分别为 $Exp(P(\Sigma_{FMS1})) = (P_{11} + P_{12})^*$ 和 $Exp(P(\Sigma_{FMS2})) = (P_{21} + P_{22})^*$. 根据定理 6, 求得 Σ_{FMS} 的进程表达式为 $Exp(P(\Sigma_{FMS})) = Exp(P(\Sigma_{FMS1})) \Theta Exp(P(\Sigma_{FMS2})) = (P_{11} + P_{12})^* \Theta (P_{21} + P_{22})^*$.

可以验证 Σ_{FMS} 的任意进程都是 $Exp(P(\Sigma_{FMS}))$ 所表示语言的前缀语言.

讨论. 根据定理 4, 对 Σ_{FMS1} 和 Σ_{FMS2} 的基本进程段 P_{11}, P_{12}, P_{21} 和 P_{22} 进行同步合成即可得到 Σ_{FMS} 的基本进程段, P_{11}, P_{12}, P_{21} 和 P_{22} 同步合成的结果为如图 6 所示的 P_1 和 P_2 . 可以验证 P_1 和 P_2 分别为 Σ_{FMS} 的基本进程段. 由于 Σ_{FMS} 是有界的, 根据文献[7]给出的有界 Petri 网的进程表达式求取方法, 可以求得 $Exp(P(\Sigma_{FMS})) = (P_1 + P_2)^*$. 可以证明 $(P_1 + P_2)^*$ 与 $(P_{11} + P_{12})^* \Theta (P_{21} + P_{22})^*$ 等价, 具体证明过程不在本文中给出.

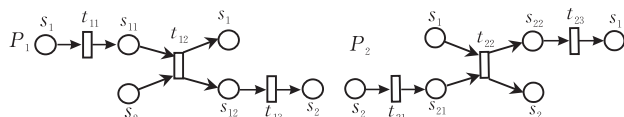


图 6 Σ_{FMS} 的两个基本进程段

显然, 利用文献[7]求得的进程表达式 $(P_1 + P_2)^*$ 要比本文给出的表达式 $(P_{11} + P_{12})^* \Theta (P_{21} + P_{22})^*$ 简洁明了. 但是, 文献[7]的方法只适应于结构有界 Petri 网的情况, 本文的方法对有界 Petri 网和无界 Petri 网均是适应的. 文献[9]通过构造无界 Petri 网的进程网系统, 然后通过分解进程网系统, 利用语言的同步交运算给出了无界 Petri 网的进程表达式的求取方法. 同文献[9]的工作相比, 本文的方法不需要构造进程网系统, 直接通过结构简单的 S-网的进程表达式的同步混排得到, 求解过程变得简单.

基于本文的工作, 需要继续研究: (1) 含有 Θ 运算的进程表达式的化简和优化方法; (2) 利用 Petri 网的进程表达式分析系统的活性、公平性、无死锁性等问题.

参 考 文 献

- [1] Yuan Chong-Yi. Principle and Application of Petri Net. Beijing: Publishing House of Electronics Industry, 2005(in Chinese)
(袁崇义. Petri 网原理与应用. 北京: 电子工业出版社, 2005)
- [2] Peterson J. Petri Net Theory and the Modeling of Systems. Englewood Cliffs: Prentice-Hall Inc., 1981
- [3] Murata T. Petri nets, properties, analysis and applications. Proceedings of the IEEE, 1989, 77(4): 541-577
- [4] Gltz U, Reisig W. Processes of place/transition net. Lecture Notes of Computer Science 154. New York: Springer-Verlag, 1983: 264-277
- [5] Lu Ru-Qian. P/R nets and P/R processes. Science in China (Series E), 1992, 35(1): 21-31
- [6] Zeng Qing-Tian, Wu Zhe-Hui. Process net system of Petri net. Chinese Journal of Computers, 2002, 25(12): 1308-1315(in Chinese)
(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 Zhe-Hui. Process expression of bounded Petri net. Science in China (Series E), 1996, 39(1): 37-49
- [8] Wu Zhe-Hui, Wang Pei-Liang, Zhao Mao-Xian. Process expression of unbounded fair Petri net. Chinese Journal of Computers, 2000, 23(4): 337-344(in Chinese)
(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] Zeng Qing-Tian, Wu Zhe-Hui. Process expression of unbounded Petri net. Chinese Journal of Computers, 2003, 26(12): 1629-1636(in Chinese)
(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] Jiang Chang-Jun. Petri net dynamic invariance. Science in China (Series E), 1997, 27(4): 605-611(in Chinese)
(蒋昌俊. Petri 网的动态不变性. 中国科学(E 辑), 1997, 27(4): 605-611)
- [11] Jiang Chang-Jun. Research of process characters of synchronous composition nets. Journal of Electronics, 1996, 25(2): 57-60(in Chinese)
(蒋昌俊. 同步合成网的进程特性研究. 电子学报, 1996, 25(2): 57-60)
- [12] Jiang Chang-Jun. Complete sequence behavior invariance of synchronous composition nets. Journal of Applied Science, 2000, 18(3): 271-275(in Chinese)
(蒋昌俊. 同步合成网的完全顺序行为不变性. 应用科学学报, 2000, 18(3): 271-275)
- [13] Zeng Qing-Tian. Behavior descriptions of structure-complex Petri nets based on synchronous composition. Journal of Software, 2004, 15(3): 327-337
- [14] Garg VK, Ragunath MT. Concurrent regular expressions and their relationship to Petri nets. Theoretical Computer Science, 1992, 96(2): 285-304



ZENG Qing-Tian, born in 1976, Ph. D., associated processor. His main research interests include theory and application of Petri net, process mining, and domain Ontology.

Background

This work is supported by National Science Foundation of China (60603090, 90718011) and the Excellent Young Scientist Foundation of Shandong Province of China (2006BS01019), supported partially by the Taishan Scholar Program of Shandong Province, and is finished by Petri net group, department of computer science and technology, Shandong University of Science and Technology.

To use Petri net for analyzing the properties of the physical systems, several methods have been presented in the net theory. Petri net processes are very convenient for analyzing concurrent phenomena and system properties relating to concurrency. This is one of the advantages of process method compared to other analysis methods of Petri net. However, a process of a Petri net only gives one possible running case for the net system. There are usually many (sometime maybe infinite) cases during a Petri net system running. It is difficult to obtain all the running cases, which brings difficulties for the analysis of Petri nets using their processes. We introduced the concept of process net system of a Petri net, a reconstruction Petri net based on the set of the basic process

sections, which can describe the process behaviors of the original system very well. And, the concept of process expression for Petri net is also introduced by our research group. Roughly, any process of a Petri net is really a composition of several basic process sections according to the process expression. We have proved the one-to-one corresponding relation between the transition firing sequences of the process net system and the processes of the original Petri net, and presented a method to obtain the process expression of a Petri net based on its process net system. The method to obtain the process expression of a Petri net especially an unbounded Petri net is usually complex. This paper proposes a new approach to construct the process expression of a Petri net based on synchronization composition. It is proved that a Petri net can be constructed by the synchronization composition of a set of S-Nets which are a kind of structure-simple Petri nets and their processes are easy to express. The process expression of the original Petri net can be obtained based on the synchronization shuffle operation of processes of these S-Nets.