Pi^+演算及其对 Petri 网的表达

郝克刚[1)] 郭小群[1)] 李向宁[2)]

[1)](西北大学信息科学与技术学院 西安 710069)

[2)](西安电子科技大学电子装备结构实验室 西安 710071)

摘 要 为了研究 Pi 演算模型的表达能力,作者用它来表达 Petri 网系统,证明了 Petri 网的某些子类,如自由选择网等,可以直接用 Pi 演算表达.然而对于一般的 Petri 网,表达却遇到了困难.文中提出了一种对 Pi 演算的扩展,称为 Pi^+演算,在原有 Pi 演算的通信机制中增加了多原语同步通信机制.证明了所有一般 Petri 网系统均可以用 Pi^+演算表达.

关键词 Petri 网;Pi 演算;π 演算;表达能力;业务过程管理

中图法分类号 TP393 **DOI 号**: 10.3724/SP.J.1016.2011.00193

The Pi^+ Calculus — An Extension of the Pi Calculus for Expressing Petri Nets

HAO Ke-Gang[1)] GUO Xiao-Qun[1)] LI Xiang-Ning[2)]

[1)](*School of Information Science and Technology, Northwest University, Xi'an* 710069)

[2)](*Laboratory of Electronic Equipment Structure, Xidian University, Xi'an* 710071)

Abstract To study the expressiveness of Pi calculus, we use it to express Petri Net systems. It is proved that some sub-classes of Petri nets can be directly expressed in Pi calculus. However, for general Petri Net systems, it turns out to be difficult. Therefore, an extension of the Pi calculus, named Pi^+ calculus, is introduced by adding a multi-primitive synchronized communication mechanism to the one of original Pi calculus. This paper proves that all general Petri net systems can be expressed in Pi^+ Calculus.

Keywords Petri net; pi calculus; π-calculus; expressiveness; BPM

1 引 言

众所周知,Petri 网[1-3]和 Pi 演算(π-calculus)[4-5]是两个很好的描写并行系统的形式化模型.随着面向服务的计算(SOC,SOA)和业务流程管理(BPM)技术的发展和广泛应用以及对软件可信性要求的提高,不少学者用它们作为服务组合和业务流程技术的理论基础[6-11].但是究竟哪个模型更合适、更好,存在着很大的争论[12-13].比较多的学者认为这两个模型各有所长各有所短,各有各的适用领域.这就引起学术界对它们表达能力(expressiveness)的探讨和研究,因为只有研究清楚了模型的表达能力,才能更准确地确定它们的适用范围.

本文用对 Petri 网系统的表达来研究 Pi 演算模型的表达能力.首先严格定义了"表达",并证明了 Petri 网的某些子类,如自由选择网等,可以直接用 Pi 演算表达.van der Aalst Wil 曾向力主使用 Pi 演算的学者提出了 7 个挑战[13],其中第 4 个挑战就是具体给出了一个例子,问它是否能用 Pi 演算方便地

收稿日期:2009-10-16;最终修改稿收到日期:2010-08-02.本课题得到国家十一五"八六三"高技术研究发展计划重点项目基金(2007AA010305)资助.**郝克刚**,男,1936 年生,教授,主要研究领域包括软件工程及相关理论、Petri 网、Pi 演算等.E-mail:hkg@nwu.edu.cn.**郭小群**,女,1971 年生,博士,讲师,主要研究方向为软件工程、面向服务的计算、业务流程管理系统等.**李向宁**,男,1976 年生,博士,讲师,主要研究方向包括工作流、业务过程管理、Pi 演算等.

表达.显然,挑战并不成功,因为这个例子属自由选择网系统,根据本文的证明,可以很容易地用 Pi 演算将其直接表达出来.本文提出了另一个例子,是一个非自由选择网系统,用它解释了一般的 Petri 网系统在 Pi 演算中表达的困难.

接着提出了一种对 Pi 演算的扩展,称为 Pi^+ 演算,在原有 Pi 演算的通信机制中增加了多原语同步通信机制.本文对 Pi^+ 演算的进程表达式形成规则、结构等价规则、归约规则以及强表达等给了严格的定义,并且证明了一般 Petri 网系统能在 Pi^+ 演算中强表达.指出 Pi^+ 演算的不同表示还能描述 Petri 网系统更加细致的动态语义.

2 Petri 网的某些子类可以用 Pi 演算直接表达

首先对"表达"给以严格的定义.设有 Petri 网系统 $N=(P,T,F,m_0)$,用 $M=\{m\mid m:P\to\mathcal{N}\}$ 表示其所有标识组成的集合,其中 $\mathcal{N}=\{0,1,2,\cdots\}$ 是自然数集合.为讨论方便,在本节中假定 Petri 网中位置的容量不受限制,弧的权值均等于 1.

定义 1(表达). 我们说 Petri 网系统 N 能被 Pi 演算表达,是指在 Pi 演算的表达式集合 E 中能定义一个关于归约封闭的子集 $SE\subseteq E$,称为状态表达式集合,并且能定义一个 SE 的结构等价类 $SE^{\equiv}$ 到 M 上的映射 $\varphi:SE^{\equiv}\to M$,使得下述两个条件成立(注:能定义一个 SE 的结构等价类 $SE^{\equiv}$ 到 M 上的映射 $\varphi:SE^{\equiv}\to M$,也可以等同地说成是定义一个 SE 到 M 上的映射 $\varphi:SE\to M$,使得如果 s 结构等价于 $s'\in SE$,即 $s\equiv s'$,则 $\varphi(s)=\varphi(s')$):

(1) 如果在 Pi 演算中有状态表达式 $s,s'\in SE$,使得 $s\to s'$,则或者 $\varphi(s)=\varphi(s')$,或者令 $\varphi(s)=m$,$\varphi(s')=m'$,在 Petri 网系统 N 中有转移 t 使得 $m[t\rangle m'$.

(2) 对于 Pi 演算中任何状态表达式 $s\in SE$,令 $\varphi(s)=m$,如果有 $t\in P^{\cdot}$ 使 $m[t\rangle m'$ 成立,则在 Pi 演算中有状态表达式序列 $s=s_1,\cdots,s_k\in SE$,使演算的归约式 $s_1\to\cdots\to s_{k-1}\to s_k$ 成立,且 $\varphi(s_1)=\cdots=\varphi(s_{k-1})=m$,$\varphi(s_k)=m'$.

定义 2(Pi 演算状态表达式). 我们在 Pi 演算中为 Petri 网系统 N 中每个位置和每个转移分别定义一系列带状态的位置表达式和带状态的转移表达式.对每个位置取一个带状态的位置表达式,对每个转移取一个带状态的转移表达式,然后把这些表达式用并行算子"|"连起来所构成的表达式称为状态表达式.SE 定义为所有状态表达式的集合的结构等价关系闭包,显然它是 Pi 演算的表达式集合 E 的子集:$SE\subseteq E$,而且从下述具体定义可知 SE 关于归约是封闭的.

具体说,我们为 N 中每个位置 P_i 定义分别带状态 0,1,2,… 的位置表达式如下:

$$(P_i,0)=!f_i\cdot\bar{g}_i\cdot 0,$$
$$(P_i,1)=!f_i\cdot\bar{g}_i\cdot 0\mid\bar{g}_i\cdot 0,$$
$$(P_i,2)=!f_i\cdot\bar{g}_i\cdot 0\mid\bar{g}_i\cdot 0\mid\bar{g}_i\cdot 0,$$
$$(P_i,3)=!f_i\cdot\bar{g}_i\cdot 0\mid\bar{g}_i\cdot 0\mid\bar{g}_i\cdot 0\mid\bar{g}_i\cdot 0,$$
$$\cdots$$

也就是说,位置表达式的状态等于表达式中含有的以 $\bar{g}_i$ 为第一前缀的子表达式的个数.

显然,根据 Pi 演算的归约规则,有下述归约式成立($k=0,1,2,\cdots$):

$$(P_i,k)\xrightarrow{\bar{f}_i}(P_i,k+1),$$
$$(P_i,k+1)\xrightarrow{g_i}(P_i,k).$$

我们为 N 中每个转移 t 定义带状态的转移表达式如下.假定 ${}^{\cdot}t=\{P_{i1},P_{i2},\cdots,P_{is}\}$,$t^{\cdot}=\{P_{j1},P_{j2},\cdots,P_{jr}\}$,

$$T\overset{\text{def}}{=}g_{i1}\cdot\cdots\cdot g_{is}\cdot\bar{f}_{j1}\cdot\cdots\cdot\bar{f}_{jr}\cdot T;$$
$$(t,\{\})=T=g_{i1}\cdot\cdots\cdot g_{is}\cdot\bar{f}_{j1}\cdot\cdots\cdot\bar{f}_{jr}\cdot T,$$
$$(t,\{g_{i1}\})=g_{i2}\cdot\cdots\cdot g_{is}\cdot\bar{f}_{j1}\cdot\cdots\cdot\bar{f}_{jr}\cdot T,$$
$$(t,\{g_{j1},g_{i2}\})=g_{j3}\cdot\cdots\cdot g_{js}\cdot\bar{f}_{j1}\cdot\cdots\cdot\bar{f}_{jr}\cdot T,$$
$$\cdots$$
$$(t,\{g_{i1},\cdots,g_{is-1}\})=g_{is}\cdot\bar{f}_{j1}\cdot\cdots\cdot\bar{f}_{jr}\cdot T,$$
$$(t,\{\bar{f}_{j1},\cdots,\bar{f}_{jr}\})=\bar{f}_{j1}\cdot\cdots\cdot\bar{f}_{jr}\cdot T,$$
$$(t,\{\bar{f}_{j2},\cdots,\bar{f}_{jr}\})=\bar{f}_{j2}\cdot\cdots\cdot\bar{f}_{jr}\cdot T,$$
$$\cdots$$
$$(t,\{\bar{f}_{jr}\})=\bar{f}_{jr}\cdot T.$$

从上述定义可知,转移表达式的状态是个集合,它可以是空集,也可以含有若干 g_i 或若干 $\bar{f}_j$.如果转移表达式以 g_j 为前缀,则转移表达式的状态是已经执行过的 g_j 的集合.如果转移表达式以 $\bar{f}_j$ 为前缀,则状态是尚未执行的 $\bar{f}_j$ 的集合.

显然,根据 Pi 演算的归约规则,有下述归约式成立:

$$(t,\{\})\xrightarrow{\bar{g}_{i1}}(t,\{g_{i1}\})\xrightarrow{\bar{g}_{i2}}(t,\{g_{i1},g_{i2}\})\cdots$$
$$\xrightarrow{\bar{g}_{is-1}}(t,\{g_{i1},\cdots,g_{is-1}\})\xrightarrow{\bar{g}_{is}}(t,\{\bar{f}_{j1},\cdots,\bar{f}_{jr}\})$$
$$\xrightarrow{f_{j1}}(t,\{\bar{f}_{j2}\cdots,\bar{f}_{jr}\})\cdots\xrightarrow{f_{jr-1}}(t,\{\bar{f}_{jr}\})\xrightarrow{f_{jr}}(t,\{\}).$$

我们把从 $(t,\{g_{i1},\cdots,g_{is-1}\})=g_{is}\cdot\bar{f}_{j1}\cdot\cdots\cdot$

$\bar{f}_{jr}\cdot T$到$(t,\{\bar{f}_{j1},\cdots,\bar{f}_{jr}\})=\bar{f}_{j1}\cdot\cdots\cdot\bar{f}_{jr}\cdot T$的归约，即转移表达式中的最后一个$g_{is}$的归约，称为$t$的激发归约.

在上述定义和后面的论述中，用到进程名的定义以及递归定义(如T，可以在表达式中出现).虽然这些机制没有在 Pi 演算的原本定义中出现，但是已经证明，它们完全可以运用 Pi 演算的重复算子等表达[5].注意文中引入的(P,k)，(t,h)只是为了论证而引入的代表某类表达式的记号，并不是新定义的带参数的进程名，归约时它们并不在表达式中出现.

定义 3(映射 φ).　SE 到 M 上的映射 φ 定义如下.假设 Petri 网系统 N 中有 m 个位置和 n 个转移，是任意状态表达式，按照定义 2，s 可以表达为 $s=(P_1,k_1)|\cdots|(P_m,k_m)|(t_1,h_1)|\cdots|(t_n,h_n)$，其中 k_i 是相应位置表达式的状态$(i=1,2,\cdots,m)$，是非负整数；h_j是相应转移表达式的状态$(j=1,2,\cdots,n)$，是个集合.

$m=\varphi(s)$定义为

$$m(P_i)=k_i+\sum_{j=1}^{n}C(g_i,h_j)+\sum_{j=1}^{n}C(\bar{f}_i,h_j),\quad i=1,2,\cdots,m,$$

其中 $C(g_i,h_j)$，$C(\bar{f}_i,h_j)$分别表示 g_i和 $\bar{f}_i$在 h_j 中出现的次数(0 或 1).

假设有一个标识 m，我们可以构造一个状态表达式$s=(P_1,m(P_1))|\cdots|(P_m,m(P_1))|(t_1,\{\})|\cdots|(t_n,\{\})$，即所有的位置表达式的状态就等于标识 m 中相应位置的 token 数，而所有的转移表达式的状态都是空集，我们称它为标准状态表达式.显然有$\varphi(s)=m$，可见 φ 是 SE 到 M 上的映射(也称满射).另外从 Pi 演算的结构等价规则不难验证此映射 φ 满足条件：如果 $s\equiv s'\in SE$，则 $\varphi(s)=\varphi(s')$.

例如，设有 Petri 网系统如图 1.

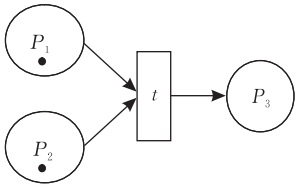

图 1　例子

$m=(1,1,0)$，$m'=(0,0,1)$，$m[t\rangle m'$，

$(P_i,0)=!f_i\cdot\bar{g}_i\cdot 0$，$i=1,2,3$，

$(P_i,1)=!f_i\cdot\bar{g}_i\cdot 0|\bar{g}_i\cdot 0$，$i=1,2,3$，

$T\overset{\text{def}}{=}g_1\cdot g_2\cdot\bar{f}_3\cdot T$，

$(t,\{\})=T=g_1\cdot g_2\cdot\bar{f}_3\cdot T$，

$(t,\{g_1\})=g_2\cdot\bar{f}_3\cdot T$，

$(t,\{\bar{f}_3\})=\bar{f}_3\cdot T$，

$s_1=(P_1,1)|(P_2,1)|(P_3,0)|(t,\{\})$，是标准状态表达式，

$s_2=(P_1,0)|(P_2,1)|(P_3,0)|(t,\{g_1\})$，

$s_3=(P_1,0)|(P_2,0)|(P_3,0)|(t,\{\bar{f}_3\})$，

$s_4=(P_1,0)|(P_2,0)|(P_3,1)|(t,\{\})$，是标准状态表达式，

$s_1\to s_2\to s_3\to s_4$，

$\varphi(s_1)=\varphi(s_2)=m$，$s_2\to s_3$是 t 的激发归约，

$\varphi(s_3)=\varphi(s_4)=m'$.

引理 1(映射 φ 满足条件 1).　如果在 Pi 演算中有状态表达式 $s,s'\in SE$，使得 $s\to s'$，令 $\varphi(s)=m$，$\varphi(s')=m'$，则或者 $m=m'$，或者在 Petri 网系统 N 中有转移 t 使得 $m[t\rangle m'$.

证明.　假定在 Pi 演算中有状态表达式 $s,s'\in SE$，使得归约式 $s\to s'$成立，且令 $\varphi(s)=m$，$\varphi(s')=m'$.由上面的定义和分析可知，s 到 s'的归约只能是某转移表达式的前缀 f 同某位置表达式的前缀 $\bar{f}$ 的通信，或者是某位置表达式的前缀 $\bar{g}$ 同某转移表达式的前缀 g 的通信.

如果是前者.假定此转移表达式是$(t,\{\bar{f}_{js},\bar{f}_{js+1},\cdots,\bar{f}_{jr}\})=\bar{f}_{js}\cdot\bar{f}_{js+1}\cdot\cdots\cdot\bar{f}_{jr}\cdot T$，$\bar{f}_{js}$在其状态中出现，位置表达式是$(P_{js},k)$.归约后转移表达式成为$(t,\{\bar{f}_{js+1},\cdots,\bar{f}_{jr}\})$，$\bar{f}_{js}$从其状态中删掉，位置表达式是$(P_{js},k+1)$，其状态加 1.所以按照 φ 的定义：

$$m(P_{js})=k_{js}+\sum_{j=1}^{n}C(g_{jsi},h_j)+\sum_{j=1}^{n}C(\bar{f}_{js},h_j).$$

归约后第 1 项加 1，第 2 项不变和第 3 项减 1，归约前后 $m(P_{js})$的总值保持不变，于是有 $\varphi(s)=\varphi(s')$.

如果是后一种情况.假定此时转移表达式具有形式：$(t,\{g_1,\cdots,g_{i-1}\})=g_i\cdot\cdots\cdot g_s\cdot\bar{f}_{j1}\cdot\cdots\cdot\bar{f}_{jr}\cdot T$，其中 $1\le i\le s$.当 $i=1$ 时状态为空集，即它是$(t,\{\})$.

我们来考察它的第一个前缀 g_i，肯定与它对应的位置表达式的状态 $k_i>0$：

$$(P_i,k_i)=!f_i\cdot\bar{g}_i\cdot 0|\bar{g}_i\cdot 0|\cdots|\bar{g}_i\cdot 0.$$

对于 $i\neq s$ 的情况，归约后对应的位置表达式的状态 k_i 减 1，而 g_i却在转移表达式的状态中新增出现，具有形式：

$$(t,\{g_1,\cdots,g_i\})=g_{i+1}\cdot\cdots\cdot g_s\cdot\bar{f}_{j1}\cdot\cdots\cdot\bar{f}_{jr}\cdot T.$$

所以按照 φ 的定义，归约后 $m(P_i)$的第 1 项减

1,第 2 项加 1 和第 3 项不变,$m(P_i)$的总值保持不变,于是有 $\varphi(s)=\varphi(s')$.

对于 $i=s$ 的情况,转移表达式具有形式:

$$(t,\{g_1,\cdots,g_{s-1}\})=g_s\cdot\bar{f}_{j1}\cdot\cdots\cdot\bar{f}_{jr}\cdot T.$$

归约后对应的位置表达式的状态 k_s 减 1,而且 $g_1,\cdots,g_{s-1}$ 都不在转移表达式的状态中出现,具有形式:

$$(t,\{\bar{f}_{js},\bar{f}_{js+1},\cdots,\bar{f}_{jr}\})=\bar{f}_{js}\cdot\bar{f}_{js+1}\cdot\cdots\cdot\bar{f}_{jr}\cdot T.$$

也就是说,在 $\varphi(s')=m$ 中对于所有的 $P_i\in{}^{\cdot}t$,按 $\varphi(s')$的定义 $m(P_i)$减 1,而对于所有的 $P_j\in t^{\cdot}$,按 $\varphi(s')$的定义 $m(P_j)$加 1. 根据 Petri 网的定义 $\varphi(s')=m'$正是 t 激发后的标识,也就是 $m[t\rangle m'$.

证毕.

定义 4(Petri 网的若干子类). Petri 网的若干类型的子类定义如下:

(1)一个 Petri 网 $N=(P,T,F)$称为是 S 网(也称状态机网 state machine),如果对于所有的 $t\in T$,有$|{}^{\cdot}t|=|t^{\cdot}|=1$ 成立.

(2)一个 Petri 网 $N=(P,T,F)$称为是无同步网,如果对于所有的 $t\in T$,有$|{}^{\cdot}t|=1$ 成立. 显然,S 网是无同步网.

(3)一个 Petri 网 $N=(P,T,F)$称为是 T 网,如果对于所有的 $p\in P$,有$|{}^{\cdot}p|\leq 1$ 而且$|p^{\cdot}|\leq 1$ 成立.

(4)一个 Petri 网 $N=(P,T,F)$称为是无竞争网,如果对于所有的 $p\in P$,有$|p^{\cdot}|\leq 1$ 成立. 显然,T 网是无竞争网.

(5)一个 Petri 网 $N=(P,T,F)$称为是自由选择网(free choice net),如果对于所有不同的 $p_1,p_2\in P$,当$p_1^{\cdot}\cap p_2^{\cdot}\neq\Lambda$ 时,有$|p_1^{\cdot}|=|p_2^{\cdot}|=1$ 成立. 显然,无同步网和无竞争网都是自由选择网. 自由选择网允许有竞争和同步,但是不允许连在一起出现. 自由选择网还有多种形式的等价定义. 例如:

①一个 Petri 网 $N=(P,T,F)$称为是自由选择网,如果对于所有的 $t\in T$,当$|{}^{\cdot}t|>1$ 时,对于所有的 $p\in{}^{\cdot}t$,有$|p^{\cdot}|=1$ 成立.

②一个 Petri 网 $N=(P,T,F)$称为是自由选择网,如果对于所有的 $p\in P$,$|p^{\cdot}|\leq 1\vee$ 对于所有的 $t\in p^{\cdot}$,有$|{}^{\cdot}t|=1$ 成立. 也就是说在自由选择网中,每个弧或者是从某位置唯一引出的弧,或者是唯一引入某转移的弧.

引理 2. 对于自由选择网,映射 φ 满足条件 2.

证明. 即要证明限制在自由选择网的条件下,对于 Pi 演算中任何状态表达式 $s\in SE$,如果有 $t\in P^{\cdot}$且 $\varphi(s)[t\rangle$,则在 Pi 演算中有状态表达式序列 $s=s_1,\cdots,s_k\in SE$,使 $s_1\to\cdots\to s_{k-1}\to s_k$, $\varphi(s)=\varphi(s_1)=\cdots=\varphi(s_{k-1})$和 $\varphi(s)[t\rangle\varphi(s_k)$成立.

在自由选择网的条件下,仅有两种情况(如图 2):$|p^{\cdot}|\leq 1$,或者对于所有的 $t\in p^{\cdot}$有$|{}^{\cdot}t|=1$.

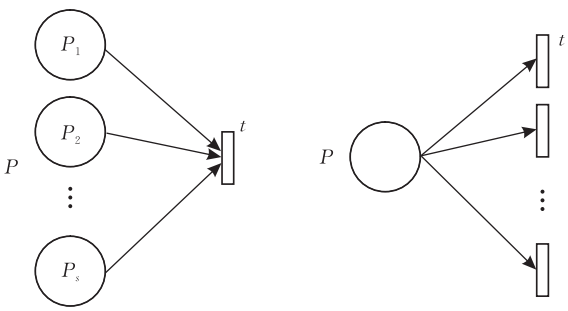

图 2 自由选择网的两种情况

第 1 种情况,如果有 $|p^{\cdot}|\leq 1$. 因为有 $t\in P^{\cdot}$且$\varphi(s)[t\rangle$,则$|p^{\cdot}|=1$,即 $p^{\cdot}=\{t\}$. 由于$|{}^{\cdot}t|\geq 1$,设 $p\in{}^{\cdot}t=\{p_1,p_2,\cdots,p_s\}$,$s\geq 1$. 根据自由选择网的特性有$|p_1^{\cdot}|=|p_2^{\cdot}|=\cdots=|p_s^{\cdot}|=1$,即 $p_1^{\cdot}=p_2^{\cdot}=\cdots=p_s^{\cdot}=\{t\}$. 我们来考察 s 中与转移 t 相应的状态表达式.

如果它的状态含有若干 $\bar{f}$,不失一般可以假定它是

$$(t,\{\bar{f}_{j1},\cdots,\bar{f}_{jr}\})=\bar{f}_{j1}\cdot\cdots\cdot\bar{f}_{jr}\cdot T.$$

于是从 s 出发经过它同与位置 $p_{j1},\cdots,p_{jr}$ 相应的状态表达式的通信,最后归约为$(t,\{\})$,而且在归约中 $\varphi(s)$保持不变. 所以我们说,s 本身或经过若干次的保持 $\varphi(s)$不变的归约,可以使其中与转移 t 相应的状态表达式的状态或者为空集,或者只含有若干 g. 不失一般性,可以假定它是下式中之一.

$$(t,\{\})=g_1\cdot\cdots\cdot g_s\cdot\bar{f}_{j1}\cdot\cdots\cdot\bar{f}_{jr}\cdot T,$$

$$(t,\{g_1\})=g_2\cdot\cdots\cdot g_s\cdot\bar{f}_{j1}\cdot\cdots\cdot\bar{f}_{jr}\cdot T,$$

$$(t,\{g_1,g_2\})=g_3\cdot\cdots\cdot g_s\cdot\bar{f}_{j1}\cdot\cdots\cdot\bar{f}_{jr}\cdot T,$$

$$\cdots$$

$$(t,\{g_1,\cdots,g_{s-1}\})=g_s\cdot\bar{f}_{j1}\cdot\cdots\cdot\bar{f}_{jr}\cdot T.$$

或者假定它具有形式:$(t,\{g_1,\cdots,g_{i-1}\})=g_i\cdot\cdots\cdot g_s\cdot\bar{f}_{j1}\cdot\cdots\cdot\bar{f}_{jr}\cdot T$,其中 $1\leq i\leq s$,当 $i=1$ 时状态为空集,即它是$(t,\{\})$. 我们来考察它的第一个前缀 g_i,与它对应的位置是 p_i. 假设此时 $\varphi(s)=m$,由于 $m[t\rangle$,有 $m(p_i)\geq 1$. 根据 φ 的定义 $m(p_i)$由三项的和组成,即

$$m(P_i)=k_i+\sum_{j=1}^{n}C(g_i,h_j)+\sum_{j=1}^{n}C(\bar{f}_i,h_j)\geq 1.$$

由于 $g_i\notin\{g_1,\cdots,g_{i-1}\}$,即 g_i 不在$(t,\{g_1,\cdots,$

$g_{i-1}\}$)的状态中出现以及 $p_i^{\cdot}=\{t\}$，g_i不可能在 t 以外的任何转移表达式的状态中出现(注：对于非自由选择网这个论断不一定成立)，所以可知上式第 2 项 $\sum_{j=1}^{n}C(g_i,h_j)=0$. 如果有某 t_j，在其状态中含有 $\bar{f}_i$，显然可以进行保持 $\varphi(s)$不变的归约使 $C(\bar{f}_i,h_j)=0$. 于是如果需要可以进行若干次保持 $\varphi(s)$不变的归约使第 3 项 $\sum_{j=1}^{n}C(\bar{f}_i,h_j)=0$. 综上所述，如果与 P_i 相应的位置表达式的状态 $k_i=0$，则可以进行若干次保持 $\varphi(s)$不变的归约使 $k_i\geq 1$.

当与 P_i相应的位置表达式的状态 $k_i\geq 1$ 时，按照定义位置表达式中就有形如 $\bar{g}_i\cdot 0$ 的并行子表达式可以同与转移 t 相应的状态表达式进行归约. 归约后转移表达式的第一个前缀成为 g_{i+1}. 同理可以依次归约，直至表达式的第一个前缀成为 g_s，而且这些归约都保持 $\varphi(s)$不变. 但是接下来的对于前缀 g_s的归约则是激发归约，归约后的状态表达式 s'正好满足 $\varphi(s)[t\rangle\varphi(s')$.

第 2 种情况，假定 $p^{\cdot}=\{t=t_1,t_2,\cdots,t_s\}$，则 $|{}^{\cdot}t|=|{}^{\cdot}t_1|=\cdots=|{}^{\cdot}t_s|=1$. 我们知道与这些转移 t_i相应的状态表达式的状态或者为空集，或者只含有若干 g，或者只含有若干 $\bar{f}$. 对于只含有若干 $\bar{f}$ 的情况，我们已经在前面证明过状态表达式 s 可以经过若干次的保持 $\varphi(s)$不变的归约，使其中与转移 t 相应的状态表达式的状态成为空集. 又由于 $|{}^{\cdot}t_i|=1$，t_i相应的状态表达式的前缀中最多只能有一个 g，所以状态只能是空集(注：对于非自由选择网这个论断不一定成立)，因而转移 t_i相应的状态表达式只能具有形式

$$(t,\{\})=g\cdot\bar{f}_{j1}\cdot\cdots\cdot\bar{f}_{jr}\cdot T.$$

我们考虑位置 p 对应的状态表达式. 假设此时 $\varphi(s)=m$，由于 $m[t\rangle$，有 $m(p)\geq 1$. 根据 φ 的定义，

$$m(P)=k+\sum_{j=1}^{n}C(g,h_j)+\sum_{j=1}^{n}C(\bar{f},h_j)\geq 1.$$

由于转移 t 相应的状态表达式的状态是空集，因而第 2 项$\sum_{j=1}^{n}C(g,h_j)=0$. 又由于前面已经证明，如果需要可以进行若干次保持 $\varphi(s)$不变的归约使第 3 项$\sum_{j=1}^{n}C(\bar{f}_i,h_j)=0$. 也就是说，如果需要可以进行若干次保持 $\varphi(s)$不变的归约使 $k\geq 1$，于是位置表达式中就有形如 $\bar{g}_i\cdot 0$ 的并行子表达式能同转移 t 相应的状态表达式进行激发归约，从而归约后的状态表达式 s'满足 $\varphi(s)[t\rangle\varphi(s')$.　　证毕.

定理 1.　自由选择网系统可以用 Pi 演算表达.

证明.　设有一任意自由选择网系统 $N=(P,T,F,m_0)$. 按照定义 2，在 Pi 演算中定义关于归约封闭的状态表达式集合 $SE\subseteq E$. 再按照定义 3 定义一个 SE 到 M 上的映射 $\varphi:SE\to M$. 由引理 1 可知 φ 满足条件 1. 再根据引理 2，对于自由选择网，映射 φ 满足条件 2. 从而根据定义 1 可知 N 可在 Pi 演算中表达.　　证毕.

由于 S 网是无同步网的子类，T 网是无竞争网的子类，而无同步网和无竞争网都是自由选择网的子类，所以由定理 1 可以推出下述定理.

定理 2.　S 网、无同步网、T 网和无竞争网系统都可以用 Pi 演算表达.

在以上讨论中都限定 Petri 网中弧的权值等于 1. 当弧的权值允许大于 1 时，有下述定理，证明从略.

定理 3.　当允许弧的权值大于 1 时，T 网和无竞争网系统可以用 Pi 演算表达.

3　用 Pi 演算直接表达 Petri 网子类的应用例子

van der Aalst Wil 向力主使用 Pi 演算的学者提出了 7 个挑战[13]，其中第 4 个挑战就是问对于图 3 具体给出的例子，是否能方便地用 Pi 演算表达.

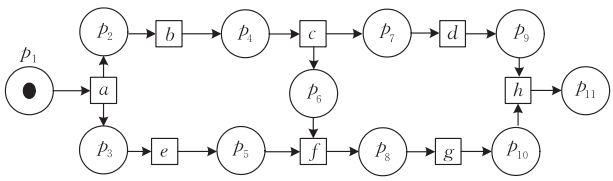

图 3　第 4 个挑战

根据 Petri 网的定义，图 3 是一个 T 网系统，因此由定理 2 可知，它能方便地直接用 Pi 演算表达. 与图中的每一位置 P_i对应的位置状态表达式可以表示为

$$(P_1,1)=!f_1\cdot\bar{g}_1\cdot 0|\bar{g}_1\cdot 0,$$
$$(P_i,0)=!f_i\cdot\bar{g}_i\cdot 0,\ i=2,\cdots,11.$$

这里 f_i，g_i是 Pi 演算中的通信通道名，分别表示相应位置的输入边和输出边. 与图中的转移$a,\cdots,h$对应的转移状态表达式分别表达如下：

$$(A,\{\})=T_A\stackrel{\text{def}}{=}g_1\cdot\bar{f}_2\cdot\bar{f}_3\cdot T_A,$$
$$(B,\{\})=T_B\stackrel{\text{def}}{=}g_2\cdot\bar{f}_4\cdot T_B,$$
$$(C,\{\})=T_C\stackrel{\text{def}}{=}g_3\cdot\bar{f}_6\cdot\bar{f}_7\cdot T_C,$$
$$(D,\{\})=T_D\stackrel{\text{def}}{=}g_7\cdot\bar{f}_9\cdot T_D,$$
$$(E,\{\})=T_E\stackrel{\text{def}}{=}g_3\cdot\bar{f}_5\cdot T_E,$$

$$(F,\{\})=T_F\stackrel{\text{def}}{=}g_5\cdot g_6\cdot\bar{f}_8\cdot T_F,$$

$$(G,\{\})=T_G\stackrel{\text{def}}{=}g_8\cdot\bar{f}_{10}\cdot T_G,$$

$$(H,\{\})=T_H\stackrel{\text{def}}{=}g_9\cdot g_{10}\cdot\bar{f}_{11}\cdot T_H.$$

于是整个 Petri 网系统可以表达为进程表达式 $(P_1,1)|(P_2,0)|\cdots|(P_{11},0)|(A,\{\})|\cdots|(H,\{\})$. 按照上述定义根据 Pi 演算归约规则不难看出,Petri 网中转移的每一次激发,从前趋位置中取 token,向后连位置中送 token,都可以看作是 Pi 演算中相对于转移的进程同相对于前趋位置以及后连位置的进程间的若干次通信归约.

显然,Aalst 的挑战用图 3 作为例子并不成功. 因为这个例子是个 T 网系统(属于自由选择网系统),根据上面所述的定理,用 Pi 演算可以很容易地将其直接表达出来. 要想选个很难用 Pi 演算表达的例子至少必须是个非自由选择网系统. 例如下节图 4 所示的例子,就很难用 Pi 演算表达. 不过后面将证明,只要将 Pi 演算加以扩充就能表达.

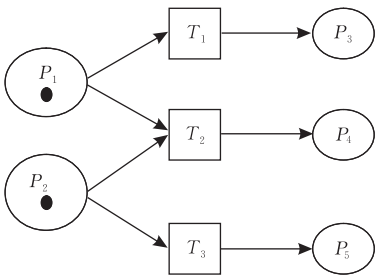

图 4 一个非自由选择网

4 用 Pi 演算表达一般的 Petri 网时遇到的困难

下面我们来解释,为什么对于一般的 Petri 网,不能用上述方法在 Pi 演算中表达. 例如这样一个实际例子:有两个不同的任务,可以分别由两个单位单独处理完成,也可以由第三单位将其合起来同时处理完成. 此例可以表达为一个 Petri 网系统(非自由选择网),如图 4,其中 $m_0(P_1)=1, m_0(P_2)=1, m_0(P_3)=m_0(P_4)=m_0(P_5)=0$. 相应的 Pi 演算进程 P 可表示如下:

$$P=(P_1,1)|(P_2,1)|(P_3,0)|(P_4,0)|(P_5,0)|(T_1,\{\})|(T_2,\{\})|(T_3,\{\}),$$

$$(P_1,1)=!f_1\cdot\bar{g}_1\cdot 0|\bar{g}_1\cdot 0,$$

$$(P_2,1)=!f_2\cdot\bar{g}_2\cdot 0|\bar{g}_2\cdot 0,$$

$$(P_3,0)=!f_3\cdot\bar{g}_3\cdot 0,$$

$$(P_4,0)=!f_4\cdot\bar{g}_4\cdot 0,$$

$$(P_5,0)=!f_5\cdot\bar{g}_5\cdot 0,$$

$$(T_1,\{\})=T_1\stackrel{\text{def}}{=}g_1\cdot\bar{f}_3\cdot T_1,$$

$$(T_2,\{\})=T_2\stackrel{\text{def}}{=}g_1\cdot g_2\cdot\bar{f}_4\cdot T_2,$$

$$(T_3,\{\})=T_3\stackrel{\text{def}}{=}g_2\cdot\bar{f}_5\cdot T_3.$$

知 $m_0=(11000)$. 设 $m_1=(01100)$,

$s_1=(P_1,1)|(P_2,1)|(P_3,0)|(P_4,0)|(P_5,0)|(T_1,\{\})|(T_2,\{\})|(T_3,\{\})$. 根据 φ 的定义和 Pi 演算的归约规则,显然有可能 $(P_1,1)$ 中的 $\bar{g}_1$ 同 $(T_1,\{\})$ 中的 g_1 通信:$s_1\rightarrow s_2\rightarrow s_3$,而且 $\varphi(s_1)=m_0$, $\varphi(s_2)=\varphi(s_3)=m_1$, $m_0[T_1\rangle m_1$,其中 $s_2=(P_1,0)|(P_2,1)|(P_3,0)|(P_4,0)|(P_5,0)|(T_1,\{\bar{f}_3\})|(T_2,\{\})|(T_3,\{\})$, $s_3=(P_1,0)|(P_2,1)|(P_3,1)|(P_4,0)|(P_5,0)|(T_1,\{\})|(T_2,\{\})|(T_3,\{\})$.

如果此时 $(P_2,1)$ 中的 $\bar{g}_1$ 同 $(T_2,\{\})$ 中的 g_1 通信,有归约:$s_3\rightarrow s_4$,而且 $\varphi(s_3)=\varphi(s_4)=m_1$. 其中 $s_4=(P_1,0)|(P_2,0)|(P_3,1)|(P_4,0)|(P_5,0)|(T_1,\{\})|(T_2,\{g_1\})|(T_3,\{\})$,显然此时 $m_1[T_3\rangle$,但是 s_4 已经再无法归约,出现了阻塞,违背了表达的条件 2. 直观地讲,问题出在 Pi 演算的通信是一个一个串行执行的,而不是一次性同步执行. 上例中 T_2 不是一次性地从 P_1 与 P_2 同时取出 token,而是一个一个地去取. T_2 与 P_2 进行通信时,T_1 已经与 P_1 进行了通信,从而 T_2 已无法再与 P_1 进行通信,而且 T_3 也已无法与 P_2 进行通信出现了阻塞.

为了表达一般的 Petri 网系统(包括位置的容量受限制,弧的权值大于 1 的情形),本文在原有 Pi 演算的通信机制中增加了多原语同步通信机制,提出了一种对 Pi 演算的扩展,称为 Pi$^+$ 演算. 并且证明了用 Pi$^+$ 演算的归约可以较准确地表达一般 Petri 网系统的动态行为,而且能更加细致地描述它丰富的动态语义.

5 Pi$^+$ 演算

Pi$^+$ 演算是在传统的 Pi 演算的基础上,增加多原语同步通信机制(multi-primitive synchronized communication mechanism). 语法作如下改动.

定义 5(Pi$^+$ 演算进程表达式). Pi$^+$ 演算的进程表达式语法(用巴科斯范式 BNF 表达):

$$P::=M|P|P|\nu zP|!P,$$

$$M::=0|\pi\cdot P|\tau\cdot P|M+M,$$

$$\pi::=\bar{x}\langle y\rangle|x(z)|\pi\pi.$$

我们称 $\bar{x}\langle y\rangle$ 为发送原语(send primitive), $x(z)$ 为接收原语(receive primitive), x 是通道名, y 是沿通道发送的数据, z 是接收变量. 在 π 的语法表

达式中，π 的并置 $\pi\pi$ 称为同步组合，在结构等价的意义上它满足结合率和交换率，所以实际上 π 是一个由若干个发送原语和接收原语组成的同步通信原语组合. 在传统的 Pi 演算中，π 仅仅是一个发送原语或接收原语. π 的归约仅相当于一对通信原语执行的一次通信. 而在我们增加了多原语同步通信机制的 Pi$^+$ 演算中，π 是由一系列发送原语和接收原语组成的同步通信原语组合. Pi$^+$ 演算中 π 的归约相当于多个进程的前缀 π 中所含有的多个通信原语成对地同步执行. 另外，对 $\bar{x}\langle y\rangle$ 和 $x(z)$，在 Pi 演算中规定，如果参数为空(不带参数的通信)，可以省略写为 $\bar{x}$ 和 x. 在 Pi$^+$ 演算中也做同样规定.

定义 6(Pi$^+$ 演算结构等价规则). Pi$^+$ 演算的结构等价规则和 Pi 演算基本相同，在其中增加：

(1) $\pi_1\pi_2\equiv\pi_2\pi_1$；

(2) $(\pi_1\pi_2)\pi_3\equiv\pi_1(\pi_2\pi_3)$；

(3) 如果 $\pi_1\equiv\pi_2$，则 $\pi_1\cdot P\equiv\pi_2\cdot P$.

定义 7(Pi$^+$ 演算前缀集合完全匹配). 设 ς 是由若干个前缀 π 组成的集合 $\varsigma=\{\pi_1,\pi_2,\cdots,\pi_n\}$，我们考虑其中所有 π 含的发送原语和接收原语，如果满足下述条件则称 ς 是完全匹配的(fully complementary).

(1) 对何任通道 x，ς 中以 x 为通道的发送原语的个数和以 x 为通道的接收原语的个数相同；

(2) ς 中所有通道名相同的发送和接收原语或者都带参数，或者都不带参数；

(3) ς 中所有通道名相同的带参数的发送原语，其发送的数据必须相同；

(4) 在 ς 的任何 π_i 中，如果 π_i 中带参数的接收原语有多个，则其接收变量不得相同.

定义 8(Pi$^+$ 演算归约规则). Pi$^+$ 演算归约规则以 Pi 演算的归约规则为基础，把 Pi 演算中的 R-react：

$$(x(y)\cdot P+M\mid\bar{x}\langle z\rangle\cdot Q+N)\to P[z/y]\mid Q$$

替换成归约规则(π). 设 $\varsigma=\{\pi_1,\cdots,\pi_s\}$ 是完全匹配的，则

$$(\pi_1\cdot P_1+M_1\mid\cdots\mid\pi_s\cdot P_s+M_s)\to P_1'\mid\cdots\mid P_s'.$$

其中 $P_i'(i=1,2,\cdots,s)$ 定义如下：如果 π_i 中的接收原语全不带参数，则 $P_i'=P_i$；否则，设 π_i 中带参数的接收原语有 $x_1(y_1),\cdots,x_k(y_k)$，ς 的所有 π 中所有沿通道 $x_j(j=1,2,\cdots,k)$ 的发送原语发送的数据分别是 z_j，则 P_i' 是由 P_i 将其中 $y_1,y_2,\cdots,y_k$ 的自由出现分别替换为 $z_1,z_2,\cdots,z_k$ 而得到的，表示为 $P_i'=P_i[z_1/y_1,z_2/y_2,\cdots,z_k/y_k]$. 由于 ς 是完全匹配的，上述完全匹配的定义 3、4 保证了这里的替换不会出现混乱.

例如：

$\bar{x}\langle y\rangle w(u)\cdot P+Q\mid x(z)\bar{w}\langle v\rangle\cdot R\to P[v/u]\mid R[y/z]$，

$\bar{x}\langle y\rangle\bar{x}\langle y\rangle\cdot P\mid x(z)\cdot Q\mid x(z)\cdot R\to P\mid Q[y/z]\mid R[y/z]$.

但是，下面的例子由于不满足完全匹配的条件，从而不能归约

$\bar{x}\langle y\rangle\bar{x}\langle y\rangle\cdot P\mid x(z)\cdot Q$(以 x 为通道的发送原语的个数是 2，同接收原语的个数 1 不同)；

$\bar{x}\langle y\rangle\cdot P\mid x\cdot Q$(发送原语带参数，接收原语不带参数)；

$\bar{x}\langle y\rangle\bar{x}\langle v\rangle\cdot P\mid x(z)\cdot Q\mid x(z)\cdot R$(发送原语发送的数据不同)；

$\bar{x}\langle y\rangle\bar{w}\langle v\rangle\cdot P\mid x(z)w(z)\cdot Q$(在第 2 个进程中接收原语接收变量相同).

这里要说明的是，原 Pi 演算中归约规则 R-react 是这里 Pi$^+$ 演算中归约规则(π)的特例($s=2$，π_1，π_2 都只含一个通信原语)，所以，把 R-react 换成归约规则(π)，实际上是对原 Pi 演算中归约的扩展. 也就是说，原 Pi 演算中的归约关系也是 Pi$^+$ 演算中归约关系.

6　用 Pi$^+$ 演算表达一般的 Petri 网

假设有一个具有边的权重和位置容量限制的 Petri 网(P/T 网)系统

$$N=(P,T,F,W,K,m_0),$$

其中 $P=\{P_1,\cdots,P_m\}$(位置集)，

$T=\{T_1,\cdots,T_n\}$(转移集)，满足 $P\cap T=\wedge$，

$F\subseteq(P\times T)\cup(T\times P)$(边集)，

W：$F\to\mathcal{N}^+$(边的权重)，$\mathcal{N}^+=\{1,2,\cdots\}$ 是大于零的自然数集合，

K：$P\to\mathcal{N}^+\cup\{\infty\}$(位置的容量)，

m_0：$P\to\mathcal{N}$(初始标识)，$\mathcal{N}=\{0,1,2,\cdots\}$ 是自然数集合.

为了后面论述方便，我们用 F 和 W 定义扩展的权重映射 A：$(P\times T)\cup(T\times P)\to\mathcal{N}$，如果 $e\in F$，则 $A(e)=W(e)$，否则 $A(e)=0$.

定义 9(强表达). 我们说 Petri 网系统 N 能在 Pi$^+$ 演算中强表达，是指在 Pi$^+$ 演算的表达式集合 E 中能定义一个关于归约封闭的子集 $SE\subseteq E$，称为状态表达式集合，并且能定义一个 SE 的结构等价类 $SE^{\equiv}$ 到所有标识组成的集合 M 上的一一映射 ψ：$SE^{\equiv}\leftrightarrow M$，使得下述两个条件成立，

(1) 如果在 Pi$^+$ 演算中有状态表达式 $s,s'\in SE$，使得 $s\to s'$，则在 Petri 网系统 N 中有由转移组成的

多重集 T^b，使得 $m[T^b\rangle m'$，其中 $m=\psi(s)$，$m'=\psi(s')$.

(2) 对于 Pi$^+$ 演算中任何状态表达式 $s\in SE$，令 $\psi(s)=m$，如果在 Petri 网系统 N 中有转移的多重集 T^b，使得 $m[T^b\rangle m'$，则在 Pi$^+$ 演算中有状态表达式 $s'\in SE$，使 $s\rightarrow s'$，$\psi(s')=m'$ 成立.

注. 多重集(Multset，也称袋子 Bag)是集合概念的一种扩展，允许其中元素重复出现并标记重复的次数(重数). 多重集通常用和式 $T^b=b_1T_1+\cdots+b_mT_m$ 来表示，其中 b_j 是重数. Petri 网系统的执行中，在不互斥可激发的条件下允许一个转移的多重集的所有成员一步同时激发.

定义 10(Pi$^+$ 演算状态表达式). 我们为 Petri 网系统 N 中每个位置在 Pi$^+$ 演算中定义一系列带状态的位置表达式，为每个转移定义一个转移表达式. 我们对每个位置取一个带状态的位置表达式 (P_i,u_i)，同所有的转移表达式 T_j 用并行算子"|"连起来，把所构成的表达式 $P=(P_1,u_1)|\cdots|(P_m,u_m)|T_1|\cdots|T_n$ 称为状态表达式. SE 定义为所有状态表达式的集合的结构等价关系闭包，显然它是 Pi$^+$ 演算的表达式集合 E 的子集：$SE\subseteq E$，而且从下述具体定义可知 SE 关于归约是封闭的.

具体说，我们为 N 中每个位置 P_i，分别针对 $K(P_i)=\infty$ 和 $K(P_i)=k$ 两种情况，定义带状态 0，1，2，…的位置表达式如下：

如果 $K(P_i)=\infty$，则

$$(P_i,0)=!f_i\cdot\bar{g}_i\cdot 0,$$

$$(P_i,1)=!f_i\cdot\bar{g}_i\cdot 0|\bar{g}_i\cdot 0,$$

$$(P_i,2)=!f_i\cdot\bar{g}_i\cdot 0|\bar{g}_i\cdot 0|\bar{g}_i\cdot 0,$$

$$\cdots,\ (P_i,r)=!f_i\cdot\bar{g}_i\cdot 0|^r\bar{g}_i\cdot 0,\cdots$$

可以看出，位置表达式的状态等于表达式中含有的以 $\bar{g}_i$ 为第一前缀的子表达式的个数. 显然，根据 Pi 演算的归约规则，有下述归约式成立($r=0,1,2,\cdots$)：

$$(P_i,r)\xrightarrow{\bar{f}_i^v}(P_i,r+v),$$

$$(P_i,r+u)\xrightarrow{g_i^u}(P_i,r).$$

如果 $K(P_i)=k$，则令 $Q_i\overset{\text{def}}{=}f_i\cdot R_i$，$R_i\overset{\text{def}}{=}\bar{g}_i\cdot Q_i+\bar{g}_if_i\cdot R_i$，带状态 0，1，2，…，$k$ 的位置表达式定义为 $(P_i,0)=|^kQ_i$，即 $(P_i,0)=Q_i|\cdots|Q_i$(共 k 个 Q_i 并行)，$(P_i,1)=|^{k-1}Q_i|R_i$，$(P_i,2)=|^{k-2}Q_i|R_i|R_i$，…，$(P_i,r)=|^{k-r}Q_i|^rR_i$，…，$(P_i,k)=|^kR_i$.

可以看出，表达式的状态等于表达式中含有 R_i 的个数，即以 $\bar{g}_i$ 或 $\bar{g}_if_i$ 为第一前缀的子表达式的个数. 在位置容量为 $K(P_i)=k$ 的情况下，如果表达式中含有 r 个 R_i，则表达式中含有 $k-r$ 个 Q_i，以 f_i 为第一前缀的子表达式. 根据 Pi 演算的归约规则，在 $r\geq u$ 和 $r-u+v\leq k$ 的条件下，有下述归约式成立：

$$(P_i,r)\xrightarrow{g_i^u\bar{f}_i^v}(P_i,r-u+v).$$

为了定义与转移 T_j($j=1,2,\cdots,n$)对应的表达式，我们令 $uij=A(P_i,T_j)$，$vji=A(T_j,P_i)$($i=1,2,\cdots,m$，$j=1,2,\cdots,n$). 与转移 T_j 对应的转移表达式定义如下

$$T_j=!g_1^{u1j}\cdots g_m^{umj}\bar{f}_1^{vj1}\cdots\bar{f}_m^{vjm}\cdot 0.$$

注. x^u 表示 u 个相同的原语 x 构成的同步通信原语组合，若 $u=0$，则 x 不在组合中出现. 从 T_j 的结构可以看出，在机构等价的意义下 T_j 在归约中保持不变，即 $T_j\xrightarrow{\bar{g}\cdots\bar{g}f\cdots f}T_j$.

在上述表达式中使用了进程的标志符(表达式名)和递归(表达式中出现表达式名). 正如 Milner 在他的著作中指出的，我们没有必要在基本的 Pi 演算中引入进程标志符和递归，因为它们可以从 Pi 演算中导出[5]. 对于 Pi$^+$ 演算也同样.

定义 11(映射 ψ). SE 的结构等价类 $SE^{\equiv}$ 到 M 上的一一映射 $\psi:SE^{\equiv}\leftrightarrow M$，定义如下.

假设 Petri 网系统 N 中有 m 个位置和 n 个转移，s 是任意状态表达式，按照定义 10，s 可以表达为 $s=(P_1,u_1)|\cdots|(P_m,u_m)|T_1|\cdots|T_n$，其中 u_i 是相应位置表达式的状态. $m=\psi(s)$ 定义为 $m(P_i)=u_i(i=1,2,\cdots,m)$. 反之，对 N 中任意标识 m 可以令与其对应的状态表达式中的 $u_i=m(P_i)(i=1,2,\cdots,m)$. 显然，如果 s 结构等价于 s'，则有 $\psi(s)=\psi(s')$. ψ 是 SE 的结构等价类 $SE^{\equiv}$ 到 M 上的一一映射.

引理 3(映射 ψ 满足强表达条件 1). 如果在 Pi$^+$ 演算中有状态表达式 $s,s'\in SE$，使得 $s\rightarrow s'$，则在 Petri 网系统 N 中有转移的多重集 T^b，使得 $m[T^b\rangle m'$，其中 $m=\psi(s)$，$m'=\psi(s')$.

证明. 按照 Pi$^+$ 演算的归约规则和 SE 的定义，如果有归约 $s\rightarrow s'$ 必然是应用了归约规则(π). 设完全匹配的 $\varsigma=\{\pi_1,\pi_2,\cdots,\pi_s\}$ 是归约中应用的 s 子表达式的前缀集合. 按照 SE 的定义，这些 $\pi_1,\cdots,\pi_s$ 均来自某些转移表达式和与其相应转移的前驱位置对应的位置表达式的子表达式. 从转移表达式的定义可知，转移表达式具有形式 $T_j=!\tau_j$，其中 $\tau_j=g_1^{u1j}\cdots g_m^{umj}\bar{f}_1^{vj1}\cdots\bar{f}_m^{vjm}\cdot 0$. 同 T_j 通信实际上是同它的结构等价的表达式

$$T_j \equiv !\tau_j \mid \tau_j \mid \cdots \mid \tau_j,$$

中的子表达式 τ_j 的通信. 假定 T_j 参与通信的子表达式的个数等于 b_j，即在 ς 中有 b_j 个 τ_j 的前缀. 我们要证的是对于转移的多重集

$$T^b = b_1 T_1 + \cdots + b_m T_m,$$

有 $\psi(s)[T^b\rangle\psi(s')$. 由于 ς 是完全匹配的，ς 中发送原语 $\bar{g}_i$ 的个数和接收原语 g_i 的个数相同，发送原语 $\bar{f}_i$ 的个数和接收原语 f_i 的个数相同从而可知，对于任何位置表达式 P_i，设它在 s 中的状态值是 r_i，它在 s' 中的状态值是 r'_i，有

$$r_i \geq \sum_{j=1}^{m} b_j u_{ij},$$

如果 $K(P_i)=k$，还有

$$r_i - \sum_{j=1}^{m} b_j u_{ij} + \sum_{j=1}^{m} b_j v_{ji} \leq k,$$

$$r'_i = r_i - \sum_{j=1}^{m} b_j u_{ij} + \sum_{j=1}^{m} b_j v_{ji}.$$

按照 Petri 网系统的激发规则可见，T^b 在标识 $\psi(s)$ 中是不互斥可激发的，而且激发的结果标识是 $\psi(s')$，即 $\psi(s)[T^b\rangle\psi(s')$. 证毕.

引理 4(映射 ψ 满足强表达条件 2). 对于 Pi$^+$ 演算中任何状态表达式 $s\in SE$，令 $\psi(s)=m$，如果在 Petri 网系统 N 中有转移的多重集 T^b，使得 $m[T^b\rangle m'$，则在 Pi$^+$ 演算中有状态表达式 $s'\in SE$，使 $s\to s'$，$\psi(s')=m'$ 成立.

证明. 设在 Petri 网系统 N 中有转移的多重集 T^b，使得 $m[T^b\rangle m'$，$T^b = b_1 T_1 + \cdots + b_m T_m$. 按照 Petri 网系统的激发规则，在 $\psi(s)=m$ 中 T^b 是不冲突可激发的. 对于任何位置 P_i，若 $m(P_i)=r_i$ 则有

$$r_i \geq \sum_{j=1}^{m} b_j u_{ij},$$

如果 $K(P_i)=k$，还有

$$r_i - \sum_{j=1}^{m} b_j u_{ij} + \sum_{j=1}^{m} b_j v_{ji} \leq k.$$

根据定义 11 关于映射 ψ 的定义，转移表达式具有形式 $T_j = !\tau_j$，其中 $\tau_j = g_1^{u1j}\cdots g_m^{umj}\,\bar{f}_1^{vj1}\cdots\bar{f}_m^{vjm}\cdot 0$. 在 s 中位置表达式 P_i 的状态值是 r_i，具有形式 $(P_i,r) = !f_i\cdot\bar{g}_i\cdot 0 \mid {}^r\bar{g}_i\cdot 0$，或者在 $K(P_i)=k$ 的情况下，具有形式

$$(P_i,r) = \mid {}^{k-r}Q_i \mid {}^r R_i,$$

其中 $Q_i \overset{\text{def}}{=} f_i\cdot R_i$，$R_i \overset{\text{def}}{=} \bar{g}_i\cdot Q_i + \bar{g}_i f_i\cdot R_i$，此时我们可以选择前缀组合 $\varsigma=\{\pi_1,\cdots,\pi_s\}$ 如下.

首先按照 T^b 选择所有结构等价的转移表达式 $T_j \equiv !\tau_j \mid \cdots \mid \tau_j$ 中 b_i 个子表达式的前缀，然后按照 $\tau_j = g_1^{u1j}\cdots g_m^{umj}\,\bar{f}_1^{vj1}\cdots\bar{f}_m^{vjm}\cdot 0$ 选择与其前缀 g 和 $\bar{f}$ 分别匹配的等量的 $\bar{g}$ 和 f 在位置表达式前缀中的出现. 由于 $r_i \geq \sum_{j=1}^{m} b_j u_{ij}$，位置表达式 P_i 中有足够多个 $\bar{g}_i$ 与其匹配，并且在 $K(P_i)=\infty$ 时，在结构等价的位置表达式 P_i 中有任意多个 f_i 与其匹配，即使在 $K(P_i)=k$ 时由于在 r_i 个 R_i 中有 $\bar{g}_i f_i$ 同时出现的前缀，加之

$$r_i - \sum_{j=1}^{m} b_j u_{ij} + \sum_{j=1}^{m} b_j v_{ji} \leq k,$$

位置表达式 P_i 中也有足够多个 $\bar{g}_i$ 与其匹配，由以上状态可知这样选择的前缀组合 $\varsigma=\{\pi_1,\cdots,\pi_s\}$ 是完全匹配的. 在按 ς 归约 $s\to s'$ 后，s' 中所有的位置表达式 P_i 的状态

$$r'_i = r_i - \sum_{j=1}^{m} b_j u_{ij} + \sum_{j=1}^{m} b_j v_{ji}.$$

而根据 Petri 网系统的激发规则，这正是 T^b 激发后 $m'=\psi(s)$ 中所有位置 P_i 的 token 数，于是证明了 $m[T^b\rangle m'$. 证毕.

定理 4. 任何具有边的权重和允许位置容量限制的一般 Petri 网系统能在 Pi$^+$ 演算中强表达.

证明. 设有一任意的具有边的权重和允许位置容量限制的 Petri 网系统 $N=(P,T,F,W,K,m_0)$. 按照定义 10，在 Pi$^+$ 演算的表达式集合 E 中定义一个状态表达式子集合 $SE\subseteq E$，它关于归约是封闭的. 再按照定义 11 定义一个 SE 的结构等价类 $SE^{\equiv}$ 到所有标识组成的集合 M 上的一一映射 ψ：$SE^{\equiv}\leftrightarrow M$. 根据引理 3～4，可知 ψ 满足条件 1 和条件 2. 从而根据定义 9 可知 N 能在 Pi$^+$ 演算中强表达. 证毕.

前面我们对 Petri 网中转移这样表示：

$$T_j = !g_1^{u1j}\cdots g_m^{umj}\;\bar{f}_1^{vj1}\cdots\bar{f}_m^{vjm}\cdot 0.$$

显然，采用这种表示，如果在转移多重集 T^b 中某转移满足能够激发若干次的条件，就可在一步中并行执行. 如果我们限制同一转移不得多于 $k(k\in\mathcal{N}^+)$ 个进程并行执行，就可表示为

$$T_j = \mid^k S_j,\quad S_j = g_1^{u1j}\cdots g_m^{umj}\;\bar{f}_1^{vj1}\cdots\bar{f}_m^{vjm}\cdot S_j.$$

如果我们限制同一转移在一步中只能执行一次，就可表示为

$$T_j = g_1^{u1j}\cdots g_m^{umj}\;\bar{f}_1^{vj1}\cdots\bar{f}_m^{vjm}\cdot T_j.$$

转移在满足激发条件后，激发执行一次. 执行后如果仍满足激发条件，转移再执行激发，但是必须在前次执行之后. 因而，即使某转移满足能够激发若干次的条件，也必须一个个串行地顺序执行，而不能并行执行. 从以上分析可知用 Pi$^+$ 演算可以更加细致地描

述和研究 Petri 网的丰富的动态语义.

7 结束语

本文用对 Petri 网系统的表达来研究 Pi 演算模型的表达能力.证明了 Petri 网的某些子类,如自由选择网等,可以直接用 Pi 演算表达.然而对于一般的 Petri 网,却很难用 Pi 演算直接表达.于是提出了一种对 Pi 演算的扩展,称为 Pi^+ 演算,将原有 Pi 演算的单原语通信机制扩充成多原语同步通信机制.证明了用 Pi^+ 演算可以较强地表达一般 Petri 网系统,而且还能更加细致地描述它丰富的动态语义.本文的重点是比较 Pi 演算和 Petri 网系统的表达能力,从理论上说明 Pi 演算和 Petri 网表达能力之间的关系.这种关系可以用图 5 来示意.

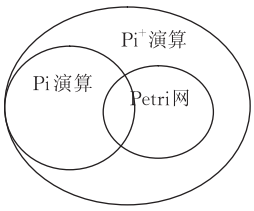

图 5 各模型表达能力的关系

之所以一般的 Petri 网不能用原有的 Pi 演算直接表达,是由于 Petri 网的转移可以有多个输入边,从而允许多个流程的同步.可是原有的 Pi 演算一次只允许一个发送进程和一个接受进程进行通信,所以只允许两个进程的同步.这可以说是原有的 Pi 演算的不足,也正是我们引入 Pi^+ 演算,扩充成多原语同步通信机制的原由.

本文指出某些 Petri 网系统很难用 Pi 演算表达,只要将 Pi 演算加以扩充就能表达一般的 Petri 网.这是否意味着 Petri 网模型的表达能力比 Pi 演算更强呢?事实上并非如此.虽然某些 Petri 网系统很难用 Pi 演算表达,但起码 Petri 网的某些子集可以直接用 Pi 演算模型表达,而且只要将 Pi 演算稍微加以自然地扩充,就能表达所有的一般 Petri 网系统.反过来,用 Petri 网模型来表达 Pi 演算则比较难.我们还不知道 Pi 演算的怎样的子集可以直接用 Petri 网模型表达.起码接收语句将接收的值代入到后继进程、通道名的传送和重复算子的使用等,都很难用 Petri 网模型直接表达.至于对 Petri 网模型做怎样的扩展就能表达一般的 Pi 演算,现在还实在想不出个眉目来.所以总体比较,Pi 演算的表达能力看来还是比 Petri 网模型更强些.

Petri 网模型在简单、直观、有严格的分析方法等方面有其优势.但是由于 Pi 演算中通道名可以作为名字传送,因而能够表达结构松散的动态耦合系统,如 Mobile 系统为保证通信的逻辑双方不变,而通信中介的物理实体可以在移动中改变;在 E-mail 系统中 mail 地址可以传送,可以根据接收到的 mail 地址进行通信;面向服务的结构中可以查询服务提供方的 URL,然后按照选择的 URL 进行绑定和链接,这都属于动态耦合.然而,Petri 网模型中由于位置和转移的连接是固定的,较适合于表达结构固定的耦合系统,很难表达松散的动态耦合系统.

由于本文证明了 Pi^+ 演算可以表达一般的 Petri 网,所以对于 Petri 网的那些向后兼容的扩展(backwards-compatible,即扩展后仍可用 Petri 网表达的)如着色网等,肯定也可以用 Pi^+ 演算表达.至于 Petri 网的那些非向后兼容的扩展,Pi^+ 演算就不一定有足够的表达能力.例如我们曾证明[14],对于带抑止弧的 Petri 网,如果抑止位置容量有穷,则可以用 Pi^+ 演算表达;如果抑止位置容量无穷,则需将 Pi^+ 演算再加以扩展,使其增加优先原语功能才能表达.今后我们将继续开展用 Pi^+ 演算对各种 Petri 网的扩展表达的研究工作.此外,为了探讨 Pi^+ 演算的表达能力,我们还研究了用它对图灵机模型的表达方法[15].

需要说明一点,本文证明了如果把 Pi 演算扩展成 Pi^+ 演算,就能表达一般的 Petri 网.但是,对论断"Pi 演算不能表达一般的 Petri 网"并没有严格证明,只能算是我们的一个直觉判断.要严格证明一个否定的论题往往相当困难,我们只是论证了不能用我们提出的用以表达自由选择网的方法在 Pi 演算中表达一般的 Petri 网.

本文只是从理论上比较了 Pi 演算和 Petri 网系统的表达能力,提出了一种比 Pi 演算表达能力更强的扩展:Pi^+ 演算,至于 Pi^+ 演算在其它领域和实际中的应用,还有待进一步的研究和探讨.

参 考 文 献

[1] Petri C A. Kommunikation mit automaten[Ph. D. dissertation]. Fakultät für Mathematik und Physik, Technische Hochschule Darmstadt, Darmstadt, Germany, 1962

[2] Reisig W, Rozenberg G eds. Lectures on Petri Nets I: Basic Models. Lecture Notes in Computer Science: 1491, Berlin: Springer-Verlag, 1998

[3] Peterson J L. Petri Net Theory and the Modeling of Systems. NJ, USA: Prentice-Hall, 1981

[4] Milner R, Parrow J, Walker D. A calculus of mobile processes, Part I/II. Information and Computation, 1992, 100(1): 1-77

[5] Milner R. Communicating and Mobile Systems: The Pi-Calculus. Cambridge, UK: Cambridge University Press, 1999
(林惠民等译. 通信与移动系统:π-演算. 北京: 清华大学出版社, 2009)

[6] Puhlmann Frank, Weske Mathias. Using the Pi-calculus for formalizing workflow patterns//Proceedings of the BPM 2005. Lecture Notes in Computer Science: 3649, Berlin: Springer-Verlag, 2005: 153-168

[7] Xue Gang, Lu Joan, Yao Shao-Wen. Investigating workflow patterns in term of Pi-calculus//Proceedings of the 11th International Conference on Computer Supported Cooperative Work in Design (CSCWD). Melbourne, Australia, 2007: 823-827

[8] Lucchi R, Mazzara M. A Pi-calculus based semantics for ws-bpel. Journal of Logic and Algebraic Programming, 2007, 70(1): 96-118

[9] Abouzaid Faisal, Mullins John. A calculus for generation, verification and refinement of BPEL. Specifications Electronic Notes in Theoretical Computer Science, 2008, 200(3): 43-65

[10] Huang Yu, Wang Han-Pin, Yu Peng, Xia Yun-Ni. Property-transition-net-based workflow process modeling and verification. Electronic Notes in Theoretical Computer Science, 2006, 159(24): 155-170

[11] Chi Yu-Liang, Lee Hsun-Ming. A formal modeling platform for composing web services. Expert Systems with Applications, 2008, 34(2): 1500-1507

[12] Smith H, Fingar P. Workflow is just a Pi process, V2.1, http://www.bpm3.com/picalculus, November 2003

[13] van der Aalst Wil. Pi calculus versus Petri nets: Let us eat "humble pie" rather than further inflate the "Pi hype". http://is.tm.tue.nl/research/atterns/download/pi-hype.pdf, May 31, 2005

[14] Guo Xiao-Qun, Hao Ke-Gang, Hou Hong, Ding Jan-Jie. Describing Petri net with inhibitor arcs using Pi calculus. Journal of System Simulation, 2009, 20(Supplement): 9-12 (in Chinese)
(郭小群, 郝克刚, 侯红, 丁剑洁. 用 Pi+ 演算表示带抑止弧的 Petri 网. 系统仿真学报, 2009, 20(增刊): 9-12)

[15] Hao Ke-Gang, Guo Xiao-Qun. Expression of turing machines with Pi calculus. Computer Engineering and Science, 2009, 31(10): 53-55(in Chinese)
(郝克刚, 郭小群. Pi 演算对图灵机的表达. 计算机工程与科学, 2009, 31(10): 53-55)

HAO Ke-Gang, born in 1936, professor. His research interests include software engineering and related theories, Petri net, Pi calculus.

GUO Xiao-Qun, born in 1971, Ph. D., lecturer. Her research interests include software engineering, service oriented computation, business process management.

LI Xiang-Ning, born in 1976, Ph. D., lecturer. His research interests include workflows, business process management, pi calculus.

Background

It is well known that Petri nets and π-calculus are two perfect formal models to describe concurrent systems. With advancement and widespread usage of service-oriented computing (SOC) and business process management (BPM) along with increasing demand for software dependability, many scholars started to use those formal models as the theoretical basis for service composition and business process technology. However, there has been a lot of debates on which of these models is more suitable and better. This leads to the research of their express power (expressiveness) in academic community.

In this paper, we study the expressiveness of the Pi calculus model by using it to express Petri net systems. First, we gave "express" a strict definition and proved that some sub-class of Petri nets, such as the free choice nets, can be expressed in Pi calculus directly. Wil van der Aalst posted 7 challenges to those who advocate Pi calculus. The fourth one is to challenge to express a given example of Petri net system in Pi calculus. This challenge can be met. The given example is of free choice net system. Based on the theorem proved in this paper, it can be expressed in Pi calculus easily. However, this paper presents another example of a non-free choice net system, and uses it to explain the difficulties of expressing general Petri net systems in Pi calculus.

Hence, we present an extension of Pi calculus, named Pi^+ calculus, by adding the so-called multi-primitive synchronized communication mechanism to the one of the original Pi calculus. The formal definitions of process expression syntax, structural congruence, reduction (reaction) rules of Pi^+ calculus and the concept of strong expression are presented in this paper. It is shown that general Petri net systems can be strongly expressed in Pi^+ calculus. It is also pointed out that the variousexpressions of Pi^+ calculus can express more detailed dynamic semantics of Petri net systems.

This paper is one of the research works supported by the National 11th Five-Year Plan High Technology Research and Development (863) Key Program of China under grant No. 2007AA010305.