序佩特里(Petri)网计算能力分析

王永革
(南开大学数学研究所,天津 300071)

ON THE COMPUTING POWER OF ORDERED PETRI NETS

Wang Yongge
(*Nankai Institute of Mathematics, Nankai University, Tianjin* 300071)

Abstract Based on the parallel computation model Petri Net, we'll introduce a kind of Ordered Petri Nets, whose computing power is much more stronger compared with Petri Net, by giving a partial order on the transition set of Petri Net, and define a parallel model of computing machine: Ordered Petri Net Computing Machine. In the last section, it is proved that this kind of machine can compute all the general recursive functions, and so it can be regarded as the basic model of computing machine.

摘要 本文通过引进序 Petri 网的概念,定义了序 Petri 网计算机,最后讨论该机器的计算能力:它可以计算所有的一般递归函数.

§1. 引 言

图灵机一直被认为是最基本的计算模型,人们也相信图灵论题的正确性,即形式可计算的一定是图灵机可计算的,反过来也成立.为证明这一论题,数学家与计算机科学家引进了多种计算模型,诸如 URM 机器、Post 产生式系统等等.不过这些模型都是串行模型,目前并行计算机迅速发展,人工智能与神经网络的研究结果也表明,人脑是并行思维的.本文借助于并发处理中的 Petri 网,定义了序 Petri 网,从而大大加强了 Petri 网作为计算模型时的计算能力与计算的确定性.最后,我们引进并行计算模型:序 Petri 网计算机,并讨论这种机器可与图灵机一样作为基本计算模型的依据.

§2. 定义及预备知识

定义 1.1: 设 S 是任一集合,S 上的一个二元关系,如果适合反身性、反对称性和传递性,就说是 S 的一个偏序关系,通常用符号"$\leqslant$"表示.即"$\leqslant$"是 S 上的一个二元关系,适合

1) $\forall x \in S: x \leqslant x$,

2) $\forall x, y \in S: (x \leqslant y) \wedge (y \leqslant x) \rightarrow x = y$,

3) $\forall x, y, z \in S: (x \leqslant y) \wedge (y \leqslant z) \rightarrow x \leqslant z$.

定义 1.2: 具有偏序关系$\leqslant$的集合 S 记作$(S, \leqslant)$,叫做偏序集.

例 1.1: 设 A 是任意集合,$S = 2^A$(A 的所有子集的集合),"$\leqslant$"表示 S 的子集间的包含关

本文 1990 年 12 月 6 日收到,1991 年 3 月 23 日定稿.本项目得到国家自然科学基金资助.作者王永革,1991 年硕士毕业于南开大学,现在 Heidelberg 大学读博士,主要研究领域为应用逻辑,形式语言,并行计算模型,Petri 网.

系，即$\forall x,y\in S: x\leqslant y\Leftrightarrow x\subseteq y$，则$(S,\leqslant)$是一个偏序集.

定义 1.3：设S是任一集合，R为S上的满足反对称性的二元关系，则称关系$\leqslant_R$为由R生成的S上的偏序关系，记为$\leqslant_R=R^*$；如果$\leqslant_R$满足

1) $\forall x\in S: x\leqslant_R x$；

2) $\forall x,y\in S: (x,y)\in R\rightarrow x\leqslant_R y$；

3) $\forall x,y,z\in S: (x\leqslant_R y)\wedge(y\leqslant_R z)\rightarrow x\leqslant_R z$.

定义 1.4：三元式$(P,T;F)$称为一个有向网，当且仅当$P\cap T=\Phi$，$P\cup T\neq\Phi$，$F\subseteq P\times T\cup T\times P$，$dom(F)\cup cod(F)=P\cup T$. 同时，P中的元素称为库所，T中的元素称为变迁. 对于$x\in P\cup T$，记$^\cdot x=\{y|(y,x)\in F\}$，$x^\cdot=\{y|(x,y)\in F\}$.

定义 1.5：八元式$(P, T; F, In, out, on, off; \leqslant_T)$称为一个序Petri网计算机，当且仅当$(P,T;F)$是一个有向网，$In\subseteq P$是输入库所集合，$out\in P$是输出库所，$on\in P$是启动开关，$off\in P$是结束开关，$\leqslant_T\subseteq T\times T$是变迁集T上的一个偏序，即$(T;\leqslant_T)$是一个偏序集.

通常我们用一个有向图来表示有向网或序Petri网计算机，如图1便是一个简单的序Petri网计算机，其中$P=\{In, out, on, off, p_1, p_2\}$；$T=\{t_1, t_2, t_3, t_4, t_5\}$；F为各条有向弧；In，on，off，out以自身标记；$\leqslant_T=\{(t_2, t_3),(t_5, t_4),(t_4, t_3)\}^*$. 亦即，$\leqslant_T$的含义是：当$(t_i, t_j)\in\leqslant_T$，$t_i\neq t_j$且$t_i$与$t_j$同时满足点火条件时，$t_i$点火，$t_j$不点火.

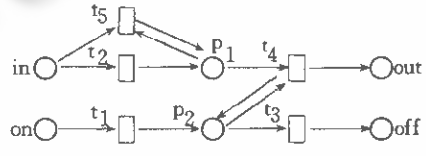

图1

为了叙述简单，本文以后用有向图的形式给出有向网或序Petri计算机时不再给出P，T，F，In，on，off和out等的形式描述，而只是在图上标明，以圆标记库所，矩形小块标记变迁.

定义 1.6：序Petri网计算机$(P, T; F, In, out, on, off; \leqslant_T)$的动作是以如下规则进行的：

(1) 计算开始时，on中有一个托肯(token)，In的各个库所中的托肯数分别为各变量的输入值，其余库所为空.

(2) 称变迁t满足点火条件，如果t中每个库所的托肯数大于零.

(3) 变迁t每点火一次，则$^\cdot t$中每个库所的托肯数减少一个，$t^\cdot$中每个库所的托肯数增加一个.

(4) 变迁t_1与t_2同时满足点火条件，且$t_1\not\leqslant_T t_2$，$t_2\not\leqslant_T t_1$时，t_1与t_2可被同时点火.

(5) 两个变迁t_1与t_2同时满足点火条件，但$t_1\leqslant_T t_2$或$t_2\leqslant_T t_1$时，如$t_1\leqslant_T t_2$，则t_1点火，t_2不点火；如$t_2\leqslant_T t_1$，则t_2点火，t_1不点火. 否则，计算非正常结束.

(6) 当处所off中托肯数由0变1，除out外，其余库所中托肯数全为零时，计算正常结束，且out中的托肯数就是运算结果.

例1.2：图1中的序Petri网计算机是以如下方式运行的：

运算开始时，如in为空，则t_2不满足点火条件，故t_1，t_3依次被点火，在处所out中输出1，正常结束运算；如in非空，则t_2，t_3都满足点火条件，但t_2的优先级大于t_3(即$t_2\leqslant_T t_3$)，故t_2被点火，接着利用t_5将in中的托肯耗尽后，点火t_4，正常结束运算，并在处所out中输出.

在Petri网的变迁上引入了偏序的概念之后，使Petri网的运行具有一定的确定性，同时也加强了计算能力. 如上述例1.2的分析表明图1中的序Petri网就具有判断输入是否为零的功能. 由于序Petri网具有这种判断功能，所以很容易理解它可计算所有一般递归函数. 下节

我们将证明这一结论.

定义 1.4：函数 $f(c_1,\cdots,c_n)$ 是序 Petri 网可计算的，如果存在一个序 Petri 网计算机 PM_f，使得对于任意输入 $(c_1,\cdots,c_n)$，PM_f的运算可正常结束当且仅当 $f(c_1,\cdots,c_n)$ 有定义，且运算结果就是函数值 $f(c_1,\cdots,c_n)$.

注：在 PM_f作为计算 n 元函数的模型时，如 PM_f的 In 中输入库所多于 n 个，则第 n+1 以后的库所中输入为空.

§3. 序 Petri 网计算能力

在本节，我们将证明序 Petri 网计算机的计算能力与图灵机相当，为此，我们只需证明序 Petri 网能够计算所有一般递归函数.

引理 1：后继函数 $f(x)=x+1$ 是序 Petri 网可计算的.

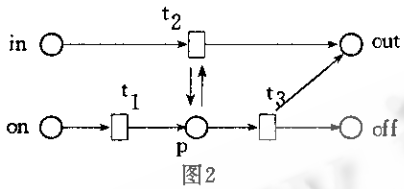

图2

证明：图 2 的序 Petri 网计算机可计算后继函数 $f(x)$. 其中 $\leqslant_T=\{(t_2,t_3)\}^*$.

其点火方式是 t_1先点火，on 中控制托肯进入 p. 接着，由于 $t_2\leqslant_T t_3$，即 t_2优先于 t_3，故可用变迁 t_3及 p 中的控制托肯将 in 中的所有托肯搬到 out 中. 最后，t_3被点火，在 off 与 out 中各放入一个托肯，运算结束，输出为库所 out 中的托肯数即 x+1. 证毕.

引理 2：减 1 函数 $f(x)=x-1$ 是序 Petri 网可计算的.（x=0 时 f(x)=0）

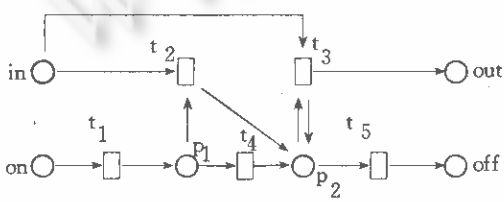

图3

证明：图 3 的序 Petri 网可计算函数 $f(x)$. 其中 $\leqslant_T=\{(t_2,t_3),(t_2,t_5),(t_3,t_5),\}^*$.

其点火方式类似于引理 1，读者可自行补出. 证毕.

引理 3：投影函数 $I_i(x_1,\cdots,x_n)=x_i$ 是序 Petri 网可计算的.

证明：图 4 的序 Petri 网可计算该投影函数. 其中 $\leqslant_T=\{(t_1,t_2),(t_2,t_3),\cdots,(t_{i-1},t_{i+1}),\cdots,(t_{n-1},t_n),(t_n,t_i)\}^*$.

其点火方式留作练习，请读者自己给出. 证毕.

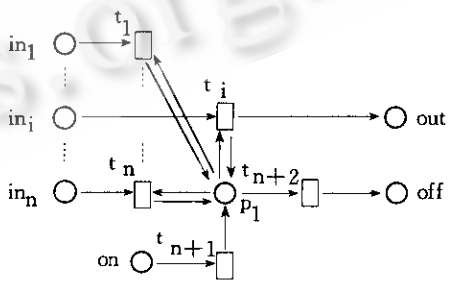

图4

引理 4：设函数 $f_1(x_1,\cdots,x_m),\cdots,f_n(x_1,\cdots,x_m)$ 和 $g(x_1,\cdots,x_n)$都是序 Petri 网可计算的，则复合函数 $F(x_1,\cdots,x_m)=g(f_1,\cdots,f_n)$ 也是序 Petri 网可计算的.

证明：我们只考虑 m=1，n=1 时的特殊情况，对于多元的情况可类似给出. 即考虑函数 $F(x)=g(f(x))$ 的序 Petri 网可计算性. 设图 5 中 (a)、(b)分别为计算 f(x) 与 g(x) 的序

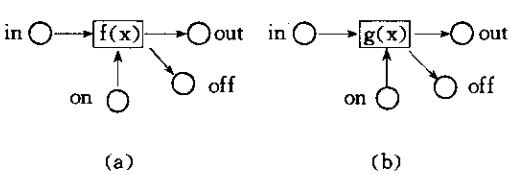

图5

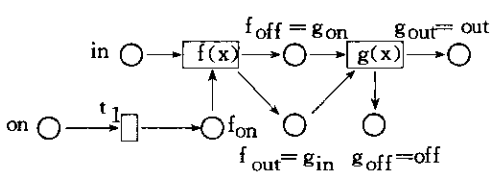

图6

Petri 网计算机，则图 6 中的序 Petri 网计算机可计算复合函数 $F(x)=g(f(x))$. 其中$\leqslant_T=(\leqslant_{T_f}\cup\leqslant_{T_g})^*$，这里$\leqslant_{T_f}$与$\leqslant_{T_g}$分别为图 5 中序 Petri 网 (a)、(b)变迁集上的偏序.

图 6 中序 Petri 网的运行过程可分为以下两个阶段：

(1) t_1点火，控制托肯进入 f(x) 的启动开关 f_{on}中. 接着计算 f(x) 的序 Petri 网被启动，f(x)计算完毕后，计算结果保存在 f 的输出库所也就是 g 的输入库所 $f_{out}=g_{in}$中，控制托肯进入 f 的结束开关也就是 g 的启动开关 $f_{off}=g_{on}$中.

(2)上述计算完成后，由于计算函数 g 的序 Petri 网满足启动条件，其输入为库所 $f_{out}=g_{in}$中的托肯数 f(x)，故这部分序 Petri 网可被启动. g 计算结束后，在 g_{out}=out 中输出g(f(x))，并置 g_{off}=off 中托肯数为 1，正常结束整个计算. 证毕.

引理 5：如果 g(x)，h(x,y,z) 是序 Petri 网可计算的，则下述函数

$$f(x,0)=g(x),f(x,y+1)=h(x,y+1,f(x,y))$$

也是序 Petri 网可计算的.

证明：图 7 的序 Petri 网可计算函数 f(x, y)：

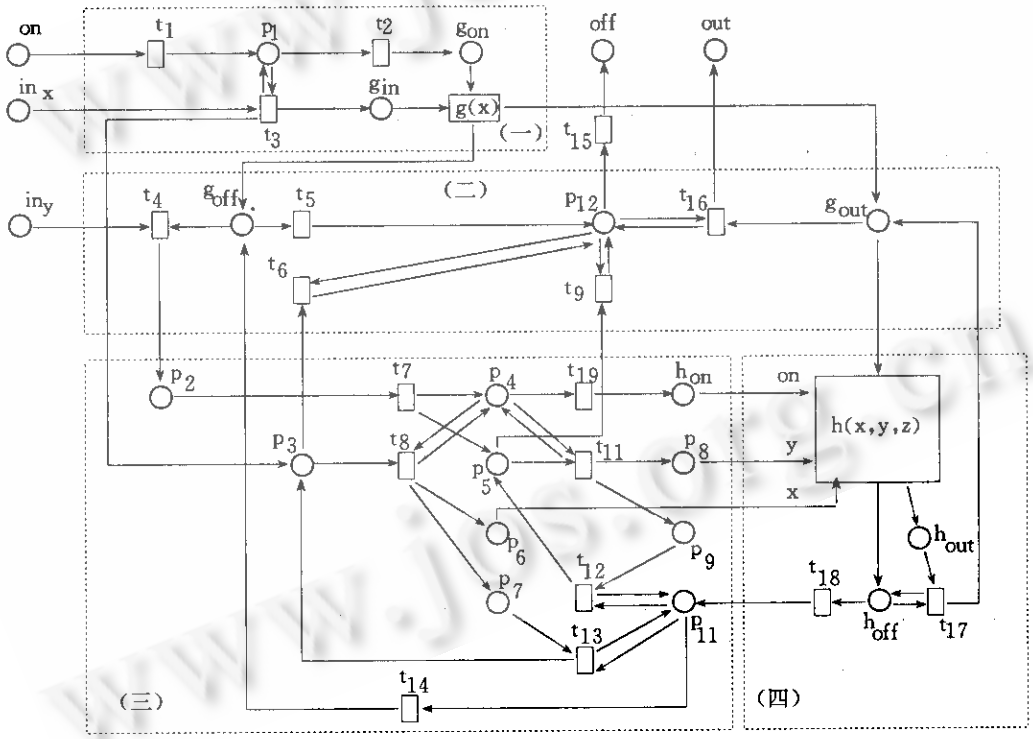

图7

图中$\leqslant_T=(\{(t_3,t_2),(t_4,t_5),(t_{16},t_{15}),(t_6,t_9),(t_9,t_{16}),(t_{11},t_8),(t_8,t_{19}),(t_{17},t_{18}),(t_{12},t_{13}),(t_{13},t_{14})\}\cup\leqslant_{T_f}\cup\leqslant_{T_g})^*$，这里$\leqslant_{T_f}$与$\leqslant_{T_g}$分别为计算函数 f 与 g 的序 Petri 网的变迁集上的偏序关系.

图 7 中的序 Petri 网由四大块组成，其功能分别如下：

(一)计算函数 g(x)；

(二)判断迭代次数是否已与输入值 y 相等，若相等，则将计算结果输出，并结束计算；若不等，则进行下一次迭代；

(三)为进行迭代运算术 h(x,y+1, f(x,y)) 准备数据及迭代运算结束后负责数据的善后处理；

(四)迭代求值 h(x,y+1,f(x,y)).

相应于以上四大模块，该序 Petri 网的运行机制可描述如下：

(一)模块一运行：

(1) 运算开始，t_1点火，on 中控制托肯进入 p_1；

(2) t_3优先级大于 t_2，故 t_3点火若干次，在 p_2与 p_5中各拷贝一份 in_x中的输入值 x；

(3) t_2点火，控制托肯进入 g_{on}，开始求 g(x)，g(x) 计算结束后，运算结果存于库所 g_{out}中，控制托肯也随之进入库所 g_{off}，启动模块二，转入下一阶段.

(二)模块二运行

(1) 此时如果 in_y为空，则表明 y=0，或是迭代若干步之后迭代次数已与输入值 y 相等. 不管何种情况，都表明运算结束，该进入数据输出阶段了，转下(2)执行；如果 in_y非空，则表明迭代次数还不够，故用变迁 t_4(注意：$t_4 \leqslant_T t_5$)将 in_y中的托肯数减少一个，并且控制托肯进入 p_2，开始下一次迭代运算数据准备阶段，即转(三)；

(2) t_5点火，控制托肯进入 p_{12}，随即用变迁 t_6、t_9清除库所 p_3、p_5 中的备份托肯，并点火 t_{16}将 g_{out}中所保存的值作为输出(注：g_{out}中保存的值或为 g(x)，或为前次迭代结果 h(x,y+1, f(x,y)))，正常结束整个运算.

(三)模块三运行之一：

(1) 为计算 h(x,y,z) 准备 y 值：t_7点火，控制托肯进入 p_4，同时将 p_5中保存的用于前次迭代的 y 值加 1. 利用偏序 $t_{11} \leqslant_T t_8$、$t_8 \leqslant_T t_{19}$，点火 t_{11}若干次，将 p_5中的值拷贝一份到 p_8中作为本次迭代用的 y 输入值，同时拷贝一份到 p_9中作为备份以便迭代结束后恢复 p_5 中用于本次迭代的 y 值；

(2) 为计算 h(x,y,z) 准备输入值 x：点火 t_3若干次，将 p_3中保存的输入值 x 拷贝一份到 p_6中作为本次迭用的 x 输入值，同时也拷贝一份到 p_7 中作为备份以便迭代结束后恢复 p_3中的 x 值；

(3) t_{19}点火，控制托肯进入 h_{on}，准备进入迭代运算，即转(四).

(四)模块四运行：

(1) 利用 h_{on}中的控制托肯启动计算 h(x,y,z) 的序 Petri 网，运算结束后结果保存在库所 h_{out}中，控制托肯进入 h_{off}；

(2) 由于 $t_{17} \leqslant_T t_{18}$，利用 t_{17}将本次迭代结果转入 g_{out}中；

(3) 点火 t_{18}，控制托肯进入 p_{11}，开始迭代结束后的数据善后处理工作，即转(五).

(五)模块三运行之二：

依次点火 t_{12}，t_{13}与 t_{14}，分别恢复 p_5与 p_3 中的 y 值与 x 值，并且控制托肯进入 g_{off}，转(二)执行. 证毕.

引理 6：如果 f(x,y) 是序 Petri 网可计算的，则 $g(x)=\mu y(f(x,y)=0)$ 也是序 Petri 网可计算的.

证明：图 8 的序 Petri 网可计算函数 g(x).

图中 $\leqslant_T = (\{(t_1,t_6), (t_5,t_7), (t_9,t_{11}), (t_{12},t_{13}), (t_8,t_4), (t_2,t_{17}), (t_{15},t_{16}), (t_{16},t_{14})\} \cup \leqslant_{T_f})^*$，这里 $\leqslant_{T_f}$是计算 f(x,y) 的序 Petri 网变迁集上的偏序关系.

图中的序 Petri 网由以下四大块组成：(一)左上角一块，为计算 f(x,y) 的值准备输入值

x;(二)左下角一块,为计算 f(x,y) 的值准备输入值 y;(三)右上角一块,求值 f(x,y);(四)右下角一块,判断 f(x,y) 的值是否为零,并决定下一步该干什么.

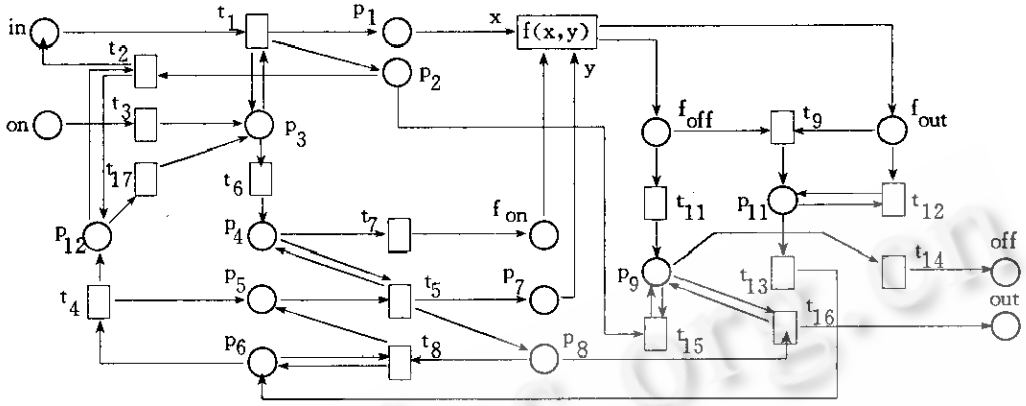

图8

相应的,该 Petri 网的运行机制如下:

(一)模块一的运行:

t_3点火,开始运算. t_1点火若干次,将 in 中的输入值拷贝一份到 p_1 中作为计算 f(x,y) 的输入值 x;同时也拷贝一份到 p_2中作为备份,以便下次运算时恢复 in 中的值.

(二)模块二的运行:

(1) t_6点火,控制托肯进入 p_4;

(2) 利用偏序关系 $t_5 \leqslant_T t_7$,点火 t_5若干次,将 p_5中的 y 值(开始第一次运算时其值为 0,以后每运算一次其值增加 1)拷贝一份到 p_7中作为计算 f(x,y) 的输入值,同时拷贝一份到 p_8中作为备份,以便下次运算时恢复 p_5的 y 值;

(3) 点火 t_7,控制托肯进入 f_{on}.

(三)模块三的运行:

利用 f_{on}中的控制托肯启动计算 f(x,y) 的序 Petri 网,f(x,y) 计算结束后,结果存于 f_{out}中,控制托肯进入 f_{off}.

(四)模块四的运行:

(1) 如果上次运算结果不为零,即 f_{out}中托肯数不为零,则转 (2),否则转 (5);

(2) t_9点火,控制托肯进入 p_{11},利用变迁 t_{12},耗尽 f_{out}中剩余的托肯;

(3) t_{13}点火,控制托肯进入 p_6,接着 t_8点火若干次,恢复 p_5中的 y 值,并利用 t_4将 p_5中的 y 值加 1,控制托肯进入 p_{12};

(4) 利用 t_2恢复 in 中的 x 值,点火 t_{17},控制托肯进入 p_3,转(一)执行(即进行下一次计算);

(5) t_{11}点火控制托肯进入 p_9,并利用 t_{15}耗尽 p_2中所保留的备份值 x;

(6) t_{16}点火若干次,将 p_8中的 y 值移到 out 中作为输出,点火 t_{14}正常结束整个计算. 证毕.

由以上六个引理,我们很容易得到:

定理:序 Petri 网计算机能够计算所有的一般递归函数.

§4. 尾　声

定义 4.1：序 Petri 网计算机 PM 的描述难度 D_{PM} 定义为 PM 中变迁集 T 的势.

定义 4.2：图灵机 TM 的描述难度 D_{TM} 定义为其指令条数.

通过仔细的对比与分析，我们发现对于三个基本函数，计算它们的序 Petri 网计算机与图灵机有大致相同的描述难度. 同样，从已知的计算 $f(x)$，$g(x)$ 与 $h(x)$ 的序 Petri 网与图灵机出发构造出的计算由这些函数经过复合、递归、μ 运算而得到的函数的序 Petri 网计算机与图灵机也具有大致相同的描述难度，因此从描述难度的角度讲，序 Petri 网也有理由作为基本的计算模型. 值得指出的是，如果用更高级的网络系统，如谓词/变迁系统(Predicate/Transition)来定义计算模型，则由于库所具有谓词功能，变迁具有判断、运算功能，所以定义的模型将显得非常简单.

结束本文前，作者感谢徐书润老师与胡久稔老师的辛勤指导，同时也对编辑与审稿同志提出的建设性修改意见表示衷心的感谢！

参考文献

[1] 袁崇义，Petri 网，1989，东南大学出版社.

[2] 张鸣华，可计算性理论，1984，清华大学出版社.

[3] 胡国定，On the Mathematical Model of Computing Machine，J. of Computer Sci. and Technology，Vol. 3，No. 4，1988.

[4] M. D. Davis Computability，Complexity，and Languages，1983，Academic Press.

[5] W. Brauer(editor)，Net Theory and Applications，Springer LNCS，Vol. 84(1980).