计算机科学 1996. Vol. 23 №. 6

22-25

Petri 网语言与传统形式语言的关系

陈 军 王元元

(南京通信工程学院 南京 210016)

TP301.2

摘 要 本文介绍了一种形式语言—Petri 网语言,并讨论了 Petri 网语言与传统形式语言(正规语言、上下文无关语言、上下文有关语言以及递归枚举语言)的关系。

关键词 Petri 网语言 正规语言 上下文无关语言 上下文有关语言 递归枚举语言

Petri 网是德国学者 C. A. Petri 于 1962 年在其博士论文中首次提出的一种自动机模型,同时也是一种可并发的状态变迁模型。之后经过 Peterson 和 Hack 等人的不断研究,Petri 网理论日臻完善和成熟。由 Petri 网产生的一类形式语言称为 Petri 网语言,它与传统的形式语言之间存在某些关系,本文就此进行揭示。

一、传统的形式语言

传统的形式语言[1]一般指正规语言(RL)、上下文无关语言(CFL)、上下文有关语言(CSL)和递归枚举语言(REL),生成这些语言的相应文法分别为正规文法(又称 3 型文法)、上下文无关文法(又称 2 型文法)、上下文有关文法(又称 1 型文法)和 0 型文法。相应的自动机分别为有限状态自动机(FSM)、下推自动机(PA)、线性界限自动机(LBA)和图灵机(TM)。这些语言类的大小是不同的,根据 Chomsky 对这些语言类的大小(文法的强弱)所做的研究,它们有类似于洋葱皮一样的结构,如图 0 所示,外面的语言包含里面的语言。

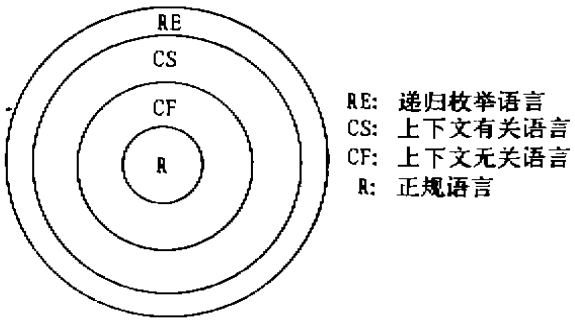


图 0 传统形式语言的大小

二、Petri 网语言

2.1 Petri 网

通常用 Petri 网结构来表示一个 Petri 网(PN)[2,3],一个 Petri 网结构可以定义为一个四元组

C=(P,T,I,O),其中:

P={p1,p2,…,pn},是位置的有穷集合,n≥0;

T={t1,t2,…,tn},是变迁的有穷集合,n≥0,P∩T=Φ;

I 是输入函数,是变迁到位置包(Bag)的映射,I: T→P∞;

O 是输出函数,也是变迁到位置包的映射,O: T→P∞。

下面给出了一个 Petri 网结构的例子:

C={P,T,I,O}

P={p1,p2,p3,p4,p5}　　T={t1,t2,t3,t4}

I(t1)={p1}　　O(t1)={p2,p3,p5}

I(t2)={p2,p3,p5}　　O(t2)={p5}

I(t3)={p3}　　O(t3)={p4}

I(t4)={p5}　　O(t4)={p2,p3}

Petri 网也可以用更加直观的图形来表示,称为 Petri 网图。同时,为了分析的方便,还可将 Petri 网用矩阵形式定义为(P,T,D⁻,D⁺),在此就不做更多的讨论了。

2.2 Petri 网语言

前面已经提到,Petri 网语言可以定义为由 Petri 网产生的一类语言,更加形式化的定义与其它几类语言的定义非常相似。除了已定义的 Petri 网结构外,还需要定义一个初始状态、一个标签(Labeling)函数和一个终止状态集合。下面就简要介绍这几个概念,但不做过多的细致讨论。

2.2.1 初始状态 是 Petri 网初始运行(Execution)时的标注(Marking)μ,可以有三种方法来定义不同的初始状态,但它们本质上都是一样的。在本文中,初始状态定义为一个任意的标注 μ。

2.2.2 标签函数 σ 是变迁 T 到字母表 Σ 中符号的映射，$\sigma: T \to \Sigma$。对 σ 的不同限制可以产生三种不同的 Petri 网语言，但稍后就可以看到，这些不同对我们的讨论并没有什么影响。

2.2.3 终止状态 是终止标注的集合，对终止状态的不同定义可以产生四种 Petri 网语言（L 型、G 型、T 型和 P 型），但本文只讨论 L 型语言，因为 L 型语言是最大的一种语言，包含了其它三类语言，对 L 型语言的讨论也最具普遍意义。下面给出 L 型 Petri 网语言的定义：

如果存在一个 Petri 网结构 (P,T,I,O)，一个变迁的标签函数 $\sigma: T \to \Sigma$，一个初始标注 μ 和一个有限的终止标注集合 F，使得 $L=\{\sigma(\beta)\in\Sigma^* \mid \beta\in T^* \text{ 且 } \delta(\mu,\beta)\in F\}$，那么语言 L 是一个 L 型 Petri 网语言。

2.2.4 标准形式的 Petri 网 上文中曾提到过，由于对标签和终止状态定义的不同，共可以定义出十二种 Petri 网语言。虽然 L 型语言是最大的一种 Petri 网语言，但由于标签定义的不同，仍可有三种不同的 L 型语言（L^l，L 和 L^λ）。这也没有什么关系，因为存在一个标准形式的 Petri 网，可以产生出任何一种 L 型语言，并且每一个 Petri 网等价于一个 Petri 网标准形。下面就给出标准形式 Petri 网的定义。

如果一个具有语言 $L(\gamma)$ 的、标记的 Petri 网 $\gamma=(C,\sigma,\mu,F)$ 具有下列特性，它就是标准形式的：

1）初始标注 μ 只包含一个具有标记（Token）的初始位置 p_s，在其它位置中不含标记。对于所有 $t_i\in T$，$p_s\in O(t_i)$。

2）存在一个位置 $p_f\in P$ 使得：(a)如果 $\lambda\in L(\gamma)$，$F=\{p_f\}$；否则 $F=\{p_f,p_s\}$。(b)对于所有 $t_i\in T$，$p_f\in I(t_i)$。(c)对所有 $t_i\in T$，若 $\mu'\in R(C,\mu)$ 且 $\mu'(p_f)>0$，则 $\delta(\mu',t_i)$ 无定义。

三、Petri 网语言与其它语言的关系

在介绍完传统的几种语言与 Petri 网语言之后，就可以来讨论它们之间的关系。本文将从最简单的正规语言开始，逐一讨论四种传统语言与 Petri 网语言的关系，最后将 Petri 网语言也纳入 Chomsky 分层结构中去。

3.1 正规语言

RL 是最简单和研究得最多的形式语言之一，由正规语法或有限状态机产生，用正规表达式表示。PN 是一种允许并发的状态变迁模型，直观上感觉，PNL 至少应不小于 RL，事实上也是如此。

定理 1 正规语言都是 Petri 网语言。

证明：首先是基于这样一个事实：任一 FSM（有限状态机）都可以转化为一等价的 PN，下面给出这种转换的方法。

首先用 PN 位置模拟 FSM 的输入和输出，并将 FSM 的每一状态都用 PN 中的一位置来表示，将当前状态用一个标记来标注，其它所有位置内无标记。然后用变迁定义状态的改变和输出；对每一对状态和输入符号定义一个变迁，其输入位置与此状态和输入符号相对应，输出位置与下一状态和输出符号相对应。更加形式化地，对每一 $FSM(Q,\Sigma,\triangle,\delta,\Gamma)$，定义一个 $PN(P,T,I,O)$：

$P=Q\cup\Sigma\cup\triangle$

$T=\{t_q,\delta \mid q\in Q,\delta\in\Sigma\}$

$I(t_q,\delta)=\{q,\delta\}$

$O(t_q,\delta)=\{\delta(q,\delta),\Gamma(q,\delta)\}$

此 PN 与被模拟的 FSM 等价。例如，图 1 所示的 PN 与图 2 所示 FSM 等价。

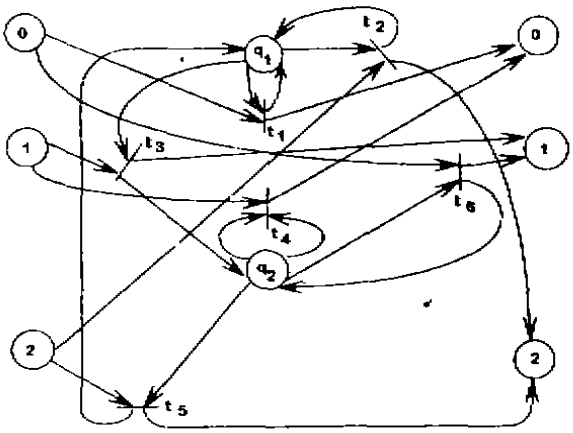


图 1 与图 2 的 FSM 相等的一个 PN

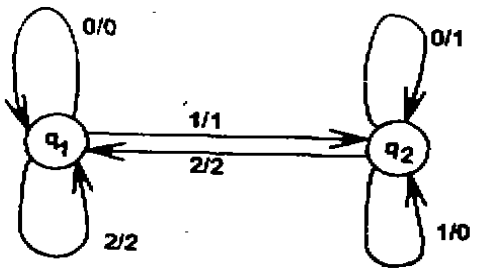


图 2 一个有限状态机

定理 1 说明了 RL 是 PNL 的子集，是不是真子集呢？回答也是肯定的。证明非常简单，可以构造一个产生语言 $L=\{a^nb^n \mid n>1\}$ 的 PN，如图 3 所示，这

是一种 CFL，而不是 RL，于是可以得出如下结论：

定理 2 正规语言是 Petri 网语言的真子集。

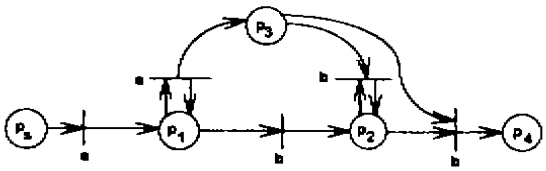


图 3 产生语言 $L=\{a^nb^n|n>1\}$ 的 Petri 网

3.2 上下文无关语言

图 3 说明了 PN 能产生 CFL，对图 3 稍作改动，就能产生语言 $L=\{a^nb^nc^n|n>0\}$，这是一种 CSL，如图 4 所示。这容易让人想到一个问题，CFL 是否也是 PNL 的子集呢？不幸的是，事实并非如此。例如 CFL $\{WW^R|W\in\Sigma^*\}$ 就不是 PNL，于是得出下面的结论：

定理 3 存在不是 Petri 网语言的上下文无关语言。

证明：假设存在两个产生语言 WW^R 的具有 n 个位置和 m 个变迁的 PN，k 为 Σ 中的符号数（$k>1$）。对一个输入字符串 XX^R，令 $r=|X|$（X 的长度）。由于存在 k^r 个可能的输入串，为了记住整个字符串 X，就必须有 k^r 个可达的状态。否则，对于两个不同的字符串 X_1 和 X_2，有 $\delta(\mu,X_1)=\delta(\mu,X_2)$，于是：

$$\begin{aligned}\delta(\mu,X_1X_2^R)&=\delta(\delta(\mu,X_1),X_2^R)\\&=\delta(\delta(\mu,X_2),X_2^R)\\&=\delta(\mu,X_2X_2^R)\in F\end{aligned}$$

此 PN 错误地产生了语言 $X_1X_2^R$。

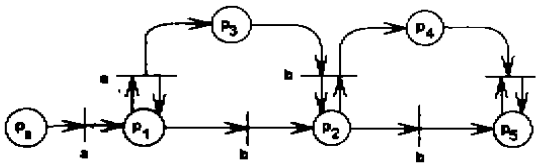


图 4 产生语言 $L=\{a^nb^nc^n|n>0\}$ 的 Petri 网

使用矩阵形式的定义 (P,T,D^-,D^+) 来表示 PN 时，对每一变迁 t_j，存在一向量 $v_j=e[j]\cdot D$，如果 $\delta(q,t_j)$ 是有定义的，则其值是 $q+v_j$。在 r 次输入后，就有一个变迁序列 $t_{j1},t_{j2}\cdots t_{jr}$ 和一个状态 $q_r=\mu+\sum_i V_{ji}$。另外一种表示 q_r 的方法为 $q_r=\mu+\sum_i f_iV_i$，其中 f_i 是变迁 t_i 在序列 $t_{j1}t_{j2}\cdots t_{jr}$ 中出现的次数（$f=(f_1,f_2,\cdots,f_m)$ 是点火向量），$\sum_i f_i=r$。

最好的情况下，向量 $v_1,v_2,\cdots,v_m$ 是线性无关的，并且每一点火向量 $(f_1,f_2,\cdots,f_m)$ 都表示唯一的状态 q_r。由于 $\sum_i f_i=r$，向量 $(f_1,f_2,\cdots,f_m)$ 将整数 r 划分成 m 个部分。Knuth 在 1973 年证明了将整数 r 划分成 m 个部分时，共可有 $\begin{bmatrix}r+m-1\\m-1\end{bmatrix}$ 种划分[5]，由于 $\begin{bmatrix}r+m-1\\m-1\end{bmatrix}=\frac{(r+m-1)\cdots(r+1)}{(m-1)!}<(r+m)^m$，所以在 r 个输入之后，可到达的状态少于 $(r+m)^m$ 个。当 r 足够大时，$\begin{bmatrix}r+m-1\\m-1\end{bmatrix}<(r+m)^m<k^r$。所以对每一有 k^r 种可能的输入字符串，不可能存在 k^r 个不同的状态，于是 PN 也不可能产生语言 WW^R。

由以上的证明可以看出，PN 不能记住任意长度的任意字符序列，故而不能模拟下推自动机来产生 CFL。同时，由于 PN 内状态连接比下推自动机状态间严格的路径具有更多的灵活性，所以 PN 能产生并非 CFL 的语言。

既然 CFL 和 PNL 互不为子集，那么什么样的语言既是 CFL 又是 PNL 呢？到目前为止，我们还不能完全回答这个问题，但可以找到一些属于 CFL 和 PNL 交集的成员。RL 显然是这样的一种语言，另外，界限上下文无关语言（BCFL）也是这样的一种语言。

3.3 上下文有关语言

到目前为止，我们已经知道 PNL 与 CFL 之间不存在相互包含关系，PNL 与 CFL 间的交集不空，而 CFL 是 CSL 的真子集，CSL 显然不是 PNL 的子集，那么 PNL 是否是 CFL 的子集呢？定理 4 回答了这个问题。

定理 4 所有 Petri 网语言都是上下文有关语言。

有两种方法来证明一种语言是 CSL：要么构造一种产生此语言的上下文有关文法，要么说明一种产生此语言的不确定线性界限自动机。Peterson 于 1973 年定义了一种能产生 PNL 的上下文有关文法，其证明过程非常复杂，在此不打算详述此证明过程，只分析一下为什么线性界限自动机能产生 PNL。

线性界限自动机与图灵机是非常相似的，与图灵机的不同之处在于其带长只能是输入字符串的线性函数。从这个意义上说，线性界限自动机与下推自动机有些相似，但下推自动机只能访问堆栈顶部的存贮器单元，而线性界限自动机能随机地访问它的

存贮器。

为了产生 PNL，线性界限自动机可以在每次输入之后记住每一位置内的标记数，以此方式来模拟 PN。那么标记数作为输入字符串的函数，其增长速度有多快呢？考虑在 r 次变迁点火之后 PN 内的标记总数 C，设变迁点火序列为 $t_{i1}t_{i2}\cdots t_{ir}$，则 $C=1+\sum_{i=1}^{r}(|O(t_{ij})|-|I(t_{ij})|)$。由于 $\#(p_k,O(t_j))$ 和 $\#(p_k,I(t_j))$ 是由 PN 的结构决定的，进而 $|O(t_j)|$ 和 $|I(t_j)|$（输出包和输入包的基数）也是有界的，因此存在一个极大值 $l=\max_{t_j\in T}(|O(t_j)|-|I(t_j)|)$，于是 $C=1+\sum_{i=1}^{r}(|O(t_{ij})|-|I(t_{ij})|)\leq 1+r\cdot l$。标记总数只能是输入长度的线性函数，所以记住这些标记数所需的存贮空间也是输入长度的线性函数。因此 PNL 能由线性界限自动机来产生。

3.4 递归枚举语言

有了以上的讨论结果，PNL 与 RNL 的关系就变得十分明显。由于 *PNL 是 CSL 的真子集，而 CSL* 是 RNL 的真子集，故 PNL 也是 RNL 的真子集。

小结 本文到此为止已讨论了 PNL 与几种传统形式语言的关系，对于大部分结论，本文给出了形式的证明，没有详述证明过程的结论，本文也作了一些易于理解的解释。最后，本文用一个文氏图来总结 *PNL* 与几种传统形式语言的关系，使读者对这些关系有一个明了而直观的认识，如图 5 所示。

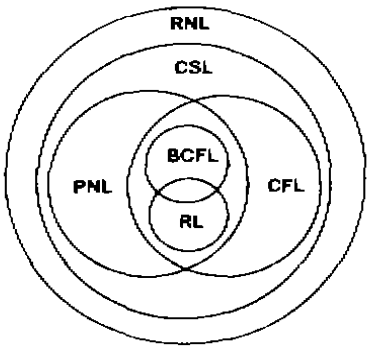


图 5 Petri 网语言与传统形式语言的关系

参考文献

[1]王元元，可计算性引论，东南大学出版社，1990

[2]Peterson J.，Petri Net and the Modeling of System，Prentece-Hall，1981

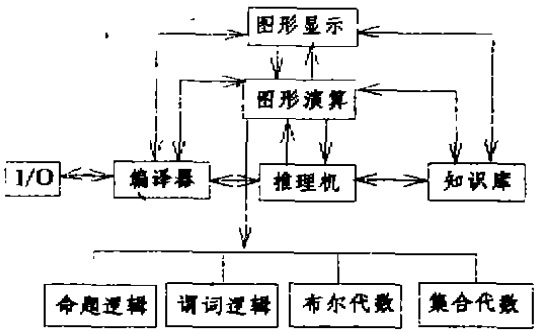
[3]陈军，Petri 网与网络协议的描述及验证，第十届南京地区研究生通信学术年会论文集，1995

[4]Peterson J.，Modeling of Parallet System，Ph. D. dissertation，Staoford University. Dec，1973

[5]Knuth D.，The Art of Computer Programming，Volume 3：Sorting and Searching，Reading，Massachusetts，Addison-Wesley，1973

（上接第 48 页）

(2)系统的实现框图：



结束语 随机问题本来就是个很复杂的问题，要在计算机上实现就显得更困难了，但是人们并不会在此止步。本文给出的随机类比推理模型及基于该模型的实现系统 DFGCS 就是关于此类问题的一种尝试。

参考文献

[1]李波，类比推理理论及实现研究，北京航空航天大学博士论文，4，1993

[2]伊波，类似模型和类似对应，南京大学博士论文，11，1989

[3]蔡庆生等，一种机器学习发现系统，CJCA90

[4]李凡长等，类比推理的数学分析，*CAAI'94*，浙江大学

[5]李凡长等，一种基于类比空间的类比推理的定义及数学模型，《计算机科学》 5. 1994

[6]李凡长，一种动态模糊逻辑及其应用的研究，中国科技大学硕士论文，1995

[7]S. Owen，Analogy for Automated Reasoning，Academic Press Limited，London，1990

[8]D. Genter，The mechanisms of Analogical Learning，In S. Vosuiadou and A. Ortony (Eds.)，Similarity and Analogical Reasoning，Cambridge University Press，1989