下推自动机的状态转换图与下推自动机的化简

张继军[1,2]　吴哲辉[2]

(山东农业大学信息学院　泰安 271018)[1]　(山东科技大学信息学院　青岛 266510)[2]

摘　要　参照有限状态自动机图形表示方式的思想方法，研究了标准下推自动机的图形表示——PAD状态转换图，证明了下推自动机与标准下推自动机的等价性。给出了对标准下推自动机进行化简的原则，并给出了化简算法，实现了下推自动机的化简。

关键词　状态转换图，标准下推自动机，化简，行为等价，状态等价

State Transition Diagram of Pushdown Automata and Simplification of Pushdown Automata

ZHANG Ji-Jun[1,2]　WU Zhe-Hui[2]

(College of Information, Shandong University Agriculture, Tai'an 271018)[1]

(College of Information Shandong, University of Science and Technology, Qingdao266510)[2]

Abstract　According to the method of finite automation diagram description, the diagram description of normal form of pushdown automata, the PAD state transition diagram is discussed. It is proofed that pushdown automat is equipollence with normal form of pushdown automata. The rules that normal form of pushdown automata is simplified and the algorithm of simplification are given. The simplification of pushdown automata is implemented.

Keywords　State transition diagram, Normal form of pushdown automata, Simplification, Behavior equivalence, State equivalence

1　引言

有限状态自动机是正规语言的识别系统，而有限状态自动机可以用状态图直观的描述，借助于状态图可以更直观、明确、容易地理解正规语言的性质及其语言的形成过程，并进一步对有限自动机确定化及化简的认识和理解，利用最简的确定有限自动机，可以更简单、容易分析对应的正规语言的性质。而对于下推自动机及其可以识别的上下文无关语言，却缺少与此对应的状态转换图，为理解和解释上下文无关语言的性质带来了不便。为此，本文参照有限状态自动机的图形表示方式的思想方法，讨论研究了标准下推自动机的图形表示——PAD状态转换图，借助于PAD状态图对下推自动机及对应的上下文无关语言有了直观的认识和了解，并实现了下推自动机的化简。从而提供了一种方便、简单的对上下文无关语言进行性质分析、描述的工具。

2　标准下推自动机与状态转换图

2.1　标准下推自动机

文[1]中，定义了下推自动机(PDA)并证明了以"空栈"和"进入终止状态"两种形式接受语言的下推自动机的等价性。为了便于讨论，本文假设使用的下推自动机是以进入终止状态时接受语言的下推自动机。

定义 1[1]　一个下推自动机是一个7元组 $M=(Q,\Sigma,\Gamma,\delta,q_0,Z_0,F)$，其中这里 Q,Σ,Γ 和 F 都是有限集合，并且：

① Q 是状态集合；②Σ 是输入字母表，并记：$\Sigma_\varepsilon=\Sigma\cup\{\varepsilon\}$；③$\Gamma$ 是栈字母表；④$\delta:Q\times\Sigma_\varepsilon\times\Gamma\to Q\times\Gamma^*$ 是转移函数；⑤$q_0\in Q$ 是起始状态；⑥$Z_0\in\Gamma$ 是栈的起始符号；⑦$F\subseteq Q$ 是接受状态集。

定义 2[1]　下推自动机 $M=(Q,\Sigma,\Gamma,\delta,q_0,Z_0,F)$，该下推自动机是确定的下推自动机(DPDA)。是指：

①对 $\forall q\in Q,\forall X\in\Gamma:\delta(q,\varepsilon,X)\neq\phi$，则 $\forall a\in\Sigma:\delta(q,a,X)=\phi$；

②对 $\forall q\in Q,\forall a\in\Sigma_\varepsilon,\forall X\in\Gamma:\delta(q,a,X)$ 至多有一个值。

文[1]针对确定的下推自动机给出了标准确定下推自动机的描述：

定义 3[1]　若确定的下推自动机的每步操作中，栈的操作最多只能弹出(或压进)一个栈顶元素，则称它为标准确定下推自动机。

由于确定下推自动机是下推自动机的真子集[1]，对定义3进行修改、扩充，给出标准下推自动机的形式定义。

定义 4　一个标准下推自动机是一个6元组 $M=(Q,\Sigma,\Gamma,\delta,q_0,q_e)$，这里 Q,Σ,Γ 和 q_0，其含义与定义1相同；并且

1)记：$\Sigma_\varepsilon=\Sigma\cup\{\varepsilon\},\Gamma_\varepsilon=\Gamma\cup\{\varepsilon\}$；

2)$\delta:Q\times\Sigma_\varepsilon\times\Gamma_\varepsilon\to Q\times\Gamma_\varepsilon$ 是转移函数；

3)$q_e\in Q$ 是接受状态。

由定义4可以得到：一个标准的下推自动机 M，只有4种基本操作(其中：$S,T\in\Gamma,p,q\in Q,a\in\Sigma$)：

① $(q,S)\in\delta(p,\varepsilon,\varepsilon)$：字符 S 进栈，状态由 p 变为 q；

② $(q,\varepsilon)\in\delta(p,\varepsilon,T)$：字符 T 出栈，状态由 p 变为 q；

③ $(q,\varepsilon)\in\delta(p,a,\varepsilon)$：从输入串中读取字符 a，状态由 p 变为 q；

④ $(q,\varepsilon)\in\delta(p,\varepsilon,\varepsilon)$：状态由 p 改变为 q。

为了便于理解和计算，对于相邻的顺序动作，将它们组合，产生另外2种操作：

⑤ $(q,S)\in\delta(p,a,\varepsilon)$：读入字符 a，S 进栈，状态由 p 变为

张继军　副教授，主要研究方向：Petri网理论及应用、软件工程。吴哲辉　教授，博士生导师，主要研究方向：Petri网理论及应用、算法设计与分析等。

q；

⑥ $(q,\varepsilon)\in\delta(p,a,T)$：读入字符 a，T 出栈，状态由 p 变为 q；

对于操作⑤：由动作 $(q_1,\varepsilon)\in\delta(p,a,\varepsilon)$，接着执行动作 $(q,S)\in\delta(q_1,\varepsilon,\varepsilon)$ 而形成。

对于操作⑥：由动作 $(q_1,\varepsilon)\in\delta(p,a,\varepsilon)$，接着执行动作 $(q,\varepsilon)\in\delta(q_1,\varepsilon,T)$ 而形成。

定理 1 任意一个由定义 1 给出描述的下推自动机，都可以转化为由定义 4 给出描述的标准下推自动机。

证明：设 $M=(Q,\Sigma,\Gamma,\delta,q_0,Z_0,F)$ 是一个以“终止状态”接受语言的下推自动机，构造一个标准下推自动机 $M'=(Q',\Sigma',\Gamma',\delta',q_s,q_e)$，使得 $L(M)=L(M')$。其构造过程为：

1)引入两个新状态 q_s,q_e，并令 $Q'=Q\cup\{q_s\}\cup\{q_e\}$；$\Sigma'=\Sigma$；$\Gamma'=\Gamma$；

2)将 $\delta'(q_s,\varepsilon,\varepsilon)=\{(q_0,Z_0)\}$ 添加到 M' 转换函数集中；

3)$\forall q\in F$，添加 $\delta'(q,\varepsilon,\varepsilon)=\{(q_e,\varepsilon)\}$ 到 M' 转换函数集中；

4)对于所有使 $\delta(q,a,Z_0)=\{(q_1,\varepsilon)\}$ 存在的 $q,q_1\in Q$ 及 $a\in\Sigma'$，添加 $\delta'(q,\varepsilon,Z_0)=\{(q_1,\varepsilon)\}$ 到 M' 的转换函数集中，并从 M 中删除 $\delta(q,a,Z_0)=\{(q_1,\varepsilon)\}$；

5)对于 M 中余下的任意一个转移关系：$(p,\alpha)\in\delta(q,a,Z)$，字符 Z 出栈，而符号串 $\alpha=u_1u_2u\cdots u_k(u_i\in\Sigma,1\leqslant i\leqslant k)$，整个过程可分解为下列序列动作完成：引入新状态 $p_1,p_2,\cdots,p_{k-1}$，将它们添加到集合 Q'，并且令转移函数：

$(p_1,u_k)\in\delta'(q,\varepsilon,Z)$，

$\delta'(p_1,\varepsilon,\varepsilon)=\{(p_2,u_{k-1})\}$

$\delta'(p_2,\varepsilon,\varepsilon)=\{(p_3,u_{k-2})\}$

……

$\delta'(p_{k-1},a,\varepsilon)=\{(q,u_1)\}$

将这些转换函数添加到转换函数集合中。

从而形成标准下推自动机 M'。

通过归纳证明，可得 $L(M)=L(M')$。

2.2 下推自动机的状态转换图

定义 5 PDA 状态转换图：设 $M=(Q,\Sigma,\Gamma,\delta,q_s,q_e)$ 是一个标准下推自动机，则 M 对应的状态转换图是一个带边标注的有向图 $G=(V,E,H,\Phi)$，其中：

①$V=Q$ 是顶点集合，

②$E=\{(p,q)\mid(p,A)\in\delta(q,a,Z)$，其中：$p,q\in Q,a\in\Sigma_\varepsilon,A,Z\in\Gamma_\varepsilon\}$ 是有向边的集合；

③$H=\Sigma\cup\Gamma\cup\{\varepsilon\}$ 是边的标注字符集合；

④$\Phi:Q\times Q\to\Sigma_\varepsilon\times\Gamma_\varepsilon\times\Gamma_\varepsilon$ 是边标注映射函数，其中：若 $(p,A)\in\delta(q,a,Z)$，则 $\Phi(p,q)=(a,Z,A)$，且 $(p,q)\in E$，记：$(a,Z\to A)$ 为边 (p,q) 的标注，“$Z\to A$”表示 Z 出栈，A 进栈；

⑤在图中用“→◎”表示起始状态，“◎”表示终止状态。

说明：定义 5 虽然是按标准下推自动机模型给出的，但同样也适用于定义 1 所给出的下推自动机定义模型。所不同的只是在标准下推自动机模型中，标注 $(a,Z\to A)$ 中，A 必须为单字符，即 $A\in\Gamma\cup\{\varepsilon\}$；而在定义 1 所给出的下推自动机定义模型中，标注 $(a,Z\to\alpha)$ 中，α 可以为字符串，即 $\alpha\in\Gamma^*$。

例 1 该例给出了接受语言 $\{0^n1^n\mid n\geqslant0\}$ 的标准下推自动机 M 及对应的 PDA 状态图。

设计构造识别语言 $\{0^n1^n\mid n\geqslant0\}$ 的一个标准下推自动机是 $M=(Q,\Sigma,\Gamma,\delta,q_1,q_4)$，其中：$Q=\{q_1,q_2,q_3,q_4\}$，$\Sigma=\{0,1\}$，$\Gamma=\{A,\$\}$，转换函数 δ：

$\delta(q_1,\varepsilon,\varepsilon)=\{(q_2,\$)\}$，

$\delta(q_2,0,\varepsilon)=\{(q_2,A)\}$，$\delta(q_2,\varepsilon,\varepsilon)=\{(q_3,\varepsilon)\}$，

$\delta(q_3,1,A)=\{(q_3,\varepsilon)\}$，$\delta(q_3,\varepsilon,\$)=\{(q_4,\varepsilon)\}$。

图 1 所示是该自动机 M 对应的 PDA 状态图。

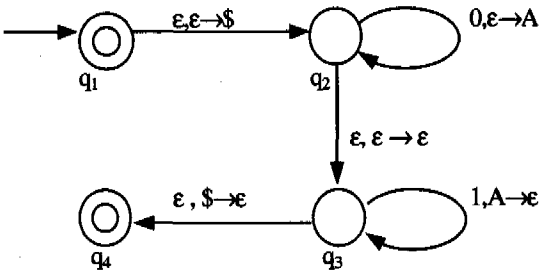

图 1 下推自动机 M 的 PDA 状态

由图 1 可以清楚地看到，下推自动机 M 的运行过程：

1)从 q_1 及空栈开始；

2)第一步：$ 进栈(作为栈空的识别，当再一次遇到 $ 时，栈空)，并进入状态 q_2；

3)第二步：识别字符‘0’，而字符‘A’进栈，重复执行；

4)第三步：由状态 q_2 转换为状态 q_3；

5)第四步：识别字符‘1’，而字符‘A’出栈，重复执行；

6)遇到字符‘$’，字符 $ 出栈，进入终态，栈又空，运行结束。

从上面运行过程分析，可以清楚地理解和认识了语言 $\{0^n1^n\mid n\geqslant0\}$ 的形成过程及性质特点。

例 2 将接受语言 $\{ww^R\mid w\in\{0,1\}^*\}$(其中 w^R 表示 w 的回文)的一个下推自动机 M，转换为一个标准下推自动机 M'。

$M=(\{q_1,q_2\},\{0,1\},\{R,B,G\},\delta,q_1,R,\varphi)$ 其中转换函数如下：

1)$\delta(q_1,0,R)=\{(q_1,BR)\}$

2)$\delta(q_1,1,R)=\{(q_1,GR)\}$

3)$\delta(q_1,0,B)=\{(q_1,BB),(q_2,\varepsilon)\}$

4)$\delta(q_1,0,G)=\{(q_1,BG)\}$

5)$\delta(q_1,1,B)=\{(q_1,GB)\}$

6)$\delta(q_1,1,G)=\{(q_1,GG),(q_2,\varepsilon)\}$

7)$\delta(q_2,0,B)=\{(q_2,\varepsilon)\}$

8)$\delta(q_2,1,G)=\{(q_2,\varepsilon)\}$

9)$\delta(q_1,\varepsilon,R)=\{(q_2,\varepsilon)\}$

10)$\delta(q_2,\varepsilon,R)=\{(q_2,\varepsilon)\}$

该自动机 M 对应的 PAD 转换图如图 2 所示。

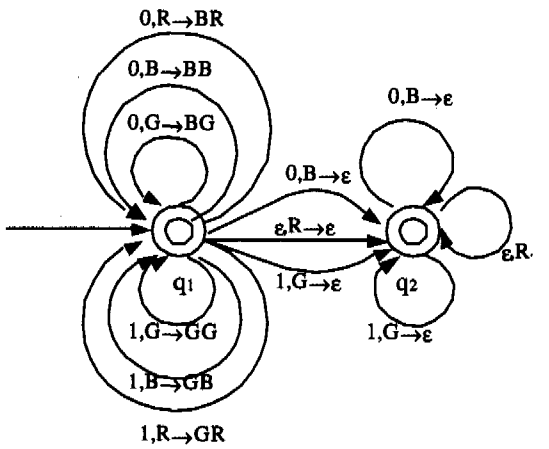

图 2 例 2 下推自动机 M 对应的 PAD 状态图

对于定理 1 的构造过程的第四步的变换，可以用图 3 形

象地描述其转换动作的实现。

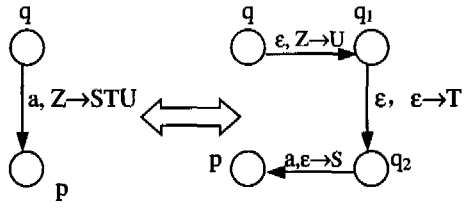

图 3 将$(P,STU)\in\delta(q,a,z)$等价变换的实现

根据定理 1 可以将例 2 的下推自动机，转换为标准下推自动机，对应的状态转换图如图 4 所示。

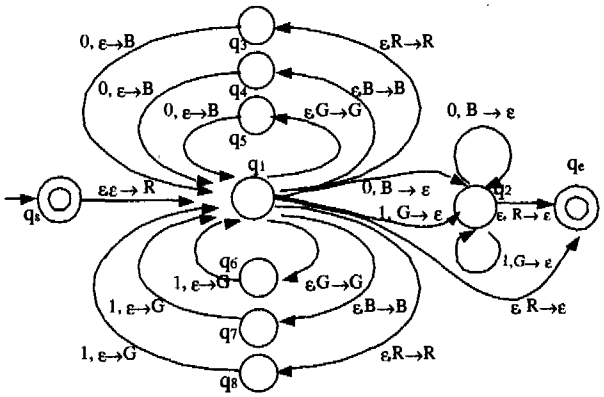

图 4 例 2 的下推自动机对应的标准下推自动机的 PAD 图

3 下推自动机的化简

图 4 看起来比图 3 要复杂得多，但是，图 4 显示了下推自动机的每一步的执行情况和运行规律。根据标准下推自动机的结构特点，可以进行下推自动机的化简，形成结构简单、易于理解和分析的下推自动机，下面给出了有关的变换定义和变换规则。

3.1 标准下推自动机的化简规则

定义 6 设$M=(Q,\Sigma,\Gamma,\delta,q_0,q_e)$是一个标准下推自动机，存在$q_1,q_2\in Q$，对$\forall a\in\Sigma$，存在$Z\in\Gamma$，使得$\delta(q_1,a,Z)=\delta(q_2,a,Z)$或$\delta(q_1,a,Z)$与$\delta(q_2,a,Z)$均不存在，则称状态$q_1$，$q_2$是等价的。

根据该定义，可得如下规则。

规则 1(等价合并规则) 设$M=(Q,\Sigma,\Gamma,\delta,q_0,q_e)$是一个标准下推自动机，若$q_1,q_2(q_1,q_2\in Q)$等价，则删除状态$q_2$，删除所有形如：$(r,\alpha)\in\delta(q_2,a,Z)$的转换函数，而将形如：$(q_2,\alpha)\in\delta(q,a,Z)$的转换函数，改写为：$(q_1,\alpha)\in\delta(q,a,Z)$，所形成的下推自动机$M'$与$M$是等价的。

其等价合并变换，见图 5 所示(图中两个状态q_1、q_2等价)。

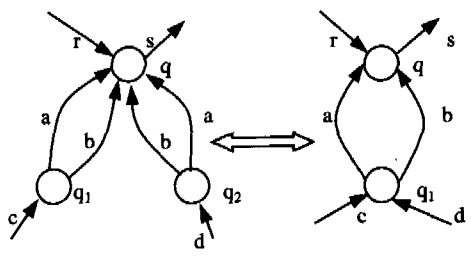

图 5 等价合并变换原则的图示

定义 7 设$M=(Q,\Sigma,\Gamma,\delta,q_0,q_e)$是一个标准下推自动机，$q_1,q_2\in Q,Z\in\Gamma$；若对于$\forall a\in\Sigma$，使得$\delta(q_2,a,Z)\subset\delta(q_1,a,Z)$或$\delta(q_1,a,Z)$与$\delta(q_2,a,Z)$均不存在，则称状态$q_1$包含了状态$q_2$。

规则 2(分解消除原则) 设$M=(Q,\Sigma,\Gamma,\delta,q_0,q_e)$是一个标准下推自动机，若状态$q_1$包含了状态$q_2(q_1,q_2\in Q)$，对于$q_1$，删除使$\delta(q_1,a,Z)=\delta(q_2,a,Z)$的转换，并添加转换：$\delta(q_1,\varepsilon,\varepsilon)=\{(p_2,\varepsilon)\}$，所形成的下推自动机$M'$与$M$是等价的。

其分解消除变换，见图 6 所示(图中两个状态$q_1\supset q_2$)。

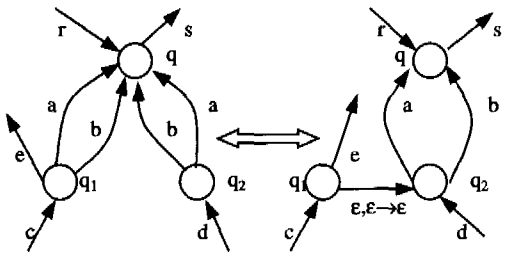

图 6 分解消除原则图示

定义 8 设$M=(Q,\Sigma,\Gamma,\delta,q_0,q_e)$是一个标准下推自动机，$q_1,q_2\in Q$；

1) $\exists a\in\Sigma,Z,A\in\Gamma$，使得：

$(q,A)\in\delta(q_2,a,Z)\wedge(q,A)\in\delta(q_1,a,Z)$，

2) $\exists b\in\Sigma,Z_1\in\Gamma$，使得：

$\delta(q_2,b,Z_1)$存在，$\delta(q_1,b,Z_1)$不存在，

3) $\exists c\in\Sigma,Z_2\in\Gamma$，使得：

$\delta(q_2,b,Z_2)$不存在，$\delta(q_1,c,Z_2)$存在，满足上述三个条件，则称状态q_1与q_2含有公共项$(a,Z\rightarrow A)\Rightarrow q$(表示由$q_1$指向$q$的边标注与$q_2$指向$q$的边标注均为$(a,Z\rightarrow A)$)。

规则 3(提取公共项原则) 设$M=(Q,\Sigma,\Gamma,\delta,q_0,q_e)$是一个标准下推自动机，对于$q_1,q_2\in Q$，若状态$q_1$与$q_2$含有公共项$(a,Z\rightarrow A)\Rightarrow q$，则引入状态$q_3$，及状态转换函数$\delta(q_3,a,Z)=(q,A)$、$\delta(q_1,\varepsilon,\varepsilon)=(q_3,\varepsilon)$和$\delta(q_2,\varepsilon,\varepsilon)=(q_3,\varepsilon)$，并删除$\delta(q_1,a,Z)=(q,A)$及$\delta(q_2,a,Z)=(q,A)$，所形成的下推自动机$M'$与$M$是等价的。

其提取公共项变换，见图 7(图中由q_1到q的两条边a，b与由q_2到q的两条边a，b是公共项)。

定义 9 设$M=(Q,\Sigma,\Gamma,\delta,q_0,q_e)$是一个标准下推自动机，$q_1,q_2\in Q,Z\in\Gamma$；则两转换关系$\delta(p_1,\varepsilon,Z)=(p_2,Z)$和$\delta(p_1,\varepsilon,\varepsilon)=(p_2,\varepsilon)$是行为等价的，即：它们只完成了状态由$p_1$到$p_2$的变换。

规则 4(行为等价原则) 设$M=(Q,\Sigma,\Gamma,\delta,q_0,q_e)$是一个标准下推自动机，$p_1,p_2\in Q,Z\in\Gamma$，将$\delta(p_1,\varepsilon,Z)=(p_2,Z)$用$\delta(p_1,\varepsilon,\varepsilon)=(p_2,\varepsilon)$替换所形成的下推自动机$M'$与$M$是等价的。

3.2 下推自动机的化简过程

根据以上的讨论，可以给出标准下推自动机的化简过程：

1)首先利用定理 1 的构造方法，将下推自动机转换为标准下推自动机；

2)根据定义 6，求出等价类，利用等价合并规则进行变换；

3)根据行为等价性，对状态转换动作进行替换；

4)根据定义 7，求出状态之间的包含关系，利用分解消除

原则进行变换；

5)根据定义8,求出状态之间的最大公共项,利用提取公共项原则进行变换；

6)重复执行2)、3)、4)直到不再有满足定义6、7、8的状态为止。

例3 对图4所表示的标准下推自动机M'进行化简,可得到图8的PAD状态图及对应的标准下推自动机M_1。

化简后的标准下推自动机$M_1=(Q,\Sigma,\Gamma,\delta,q_s,q_e)$,其中:$Q=\{q_1,q_2,q_s,q_e\}$,$\Sigma=\{0,1,\varepsilon\}$,$\Gamma=\{B,G,R,\varepsilon\}$。转换函数$\delta$:

$\delta(q_s,\varepsilon,\varepsilon)=\{(q_1,R)\}$,

$\delta(q_1,0,\varepsilon)=\{(q_1,B)\}$,$\delta(q_1,1,\varepsilon)=\{(q_1,G)\}$,

$\delta(q_1,\varepsilon,\varepsilon)=\{(q_2,\varepsilon)\}$,$\delta(q_2,1,G)=\{(q_2,\varepsilon)\}$,

$\delta(q_2,0,B)=\{(q_2,\varepsilon)\}$,$\delta(q_2,\varepsilon,R)=\{(q_e,\varepsilon)\}$。

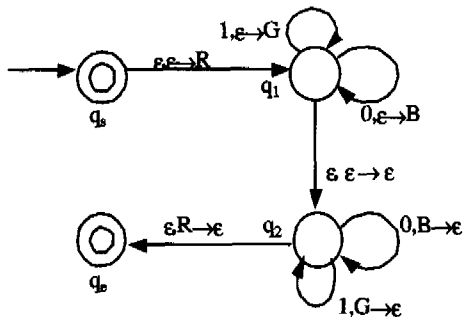

图8 化简图4所得的PDA状态图

图3与图8相比较,图8结构简单,更有利于直观地理解和认识语言$\{ww^R\mid w\in\{0,1\}^*\}$的形成过程及性质特点。

结束语 本文通过对下推自动机的基本操作的分析,定义了标准下推自动机及PAD状态图,可以直观地理解和认识下推自动机的工作机制;借助于状态图,给出了下推自动机的化简规则,实现了下推自动机的化简,将复杂的结构等价转换为简单结构,从而可以更简单、方便地分析下推自动机的性质特点,为描述、分析上下文无关语言提供了一种更简单方便的方法。

进一步的研究工作应该是:上下文无关语言的结构本质及其性质特点,为上下文无关语言提供一种并行的语法分析技术和方法。

参 考 文 献

1 Hopcroft J E, Ullman J D. Introduction to Automata Theory, Languages and Computation[M]. Addison-Wesleg, 1979

2 张立昂,等译. 计算理论导引[M]. 机械工业出版社,2000

3 吕映芝. 上下文无关文法与无限状态自动机[J]. 电子学报,1996,24(8):23～27

4 蒋立源,康慕宁. 编译原理[M]. 西安:西北工业大学出版社,1999

5 陈火旺,等著. 程序设计语言编译原理(第三版)[M]. 国防工业出版社,2000

(上接第243页)

运行像SPEC CPU这样的标准程序的SMT处理器中,可以采用各线程拥有独立分支预测器和独立BTB的结构,并且每个独立的预测器可以做得很小,从而可以有效地减少硬件开销;6)在一个主要运行包含大量不规则分支行为程序的SMT处理器中,各线程拥有独立分支预测器的结构将变得不实际,因为每个独立的预测器必须做得很大,从而带来的硬件开销太大。

需要指出的是,我们实验中的模拟器采用了较大的发射宽度,各线程都有充足的指令进入指令队列,使得各线程均等地竞争可用资源,导致了各线程的IPC相比与单个线程时有较大下降,这是我们实验的不足之处。进一步的工作中,我们将对更多的SMT结构、不同的工作负载做更多的测试分析。我们希望建立一个分支预测器可动态配置的实验环境,可以根据实际的应用和具体的线程数目动态地调整预测器配置。如果是预测器独立的策略,则线程越多时各线程的预测器就越小。当只有一个线程时,则它能使用整个预测器空间,从而充分利用资源。同时,我们希望尝试一些其它的分支处理办法,例如等待分支结果的办法,处理器遇到分支只能继续从其它线程取指,直到分支目标地址确定。

参 考 文 献

1 Marr D T, Binns F, Hill D L, et al. Hyper-Threading Technology Architecture and Microarchitecture. Intel Technical Journal, 2002,6(01)

2 Tullsen D, Eggers S, Emer J, et al. Exploiting Choice: Instruction Fetch and Issue on an Implementable Simultaneous Multithreading Processor. In: Proc. 23rd Annual Int'l Symposium on Computer Architecture, 1996. 191～202

3 Hily S, Seznec A. Branch Prediction and Simultaneous Multithreading. International Conference on Parallel Architecture and Compilation Techniques, 1996

4 Ramsay M, Feucht Ch, Lipasti M H. Exploring Efficient SMT Branch Predictor Design. University of Wisconsin-Madsin, Department of Electrical and Computer Engineering, 2003

5 Seznec A, Felix S, Krishnan V, et al. Design Tradeoffs for theAlpha EV8 Conditional Branch Predictor. Proceedings of the 29th Annual Symposium on Computer Architecture, May 25-29, 2002

6 Michaud P, Seznec A, Uhlig R. Trading conflict and capacity aliasing in conditional branch predictors. proceedings of the 24th annual International Symposium on Computer Architecture(ISCA-97), June 1997

7 Lee J K F, Smith A J. Branch Prediction Strategies and Branch Target Buffer Design. IEEE Computer, 1984,17(1):6～22

8 Yeh T Y, Patt Y N. Two-Level Adaptive Training Branch Prediction. In: Proceedings of the 24th Annual International Symposium on Microarchitecture, Nov. 1991. 51～61

9 McFarling S. Combining branch predictors. [Technical Report TN-36]. Digital Western Research Laboratory, June 1993

10 Goncalves R, Ayguadé E, Valero M, et al. A Simulator for SMT Architectures: Evaluating Instruction Cache Topologies. Supported by the Spanish Ministry of Education(TIC98-511), 2000

11 Burger D, austin T M. The SimpleScalar Tool Set, Version 2.0. http://www.simplescalar.com, 1997

12 KleinOsowski A J, Flynn J, Meares N, et al. Adapting the SPEC2111 Benchmark Suite for Simulation-Based Computer Architecture Research. http://www.arctic.umn.edu, 2002