

Chapter 6

下推自动机

6.1 下推自动机

正如正则表达式有一个等价的自动机 — 有穷自动机一样, 上下文无关文法也有其相应的机器 — 下推自动机. 这里的等价性不太令人满意, 因为下推自动机是一个非确定的装置, 对应的确定装置只能接受全部 CFL 的一个子集. 幸运的是这个子集包含了绝大多数的程序设计语言的文法.

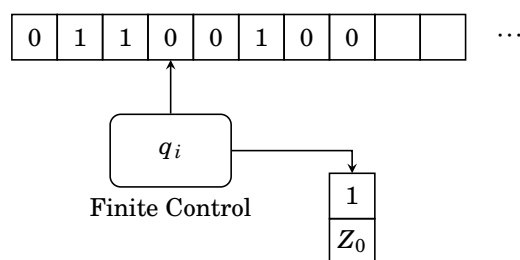
下推自动机实质上是一种能控制一条输入带和一个栈的有穷自动机, 可以看作带有栈的 ϵ -NFA. 工作方式类似 ϵ -NFA, 有一个有穷控制器, 并能够以非确定的方式进行状态转移, 并读入输入字符; 增加的堆栈, 用来存储无限的信息, 但只能以后进先出的方式使用.

$$\epsilon\text{-NFA} + \text{栈} = \text{PDA}$$

ϵ -NFA: 有限状态, 非确定, ϵ 转移

栈: 后进先出, 只用栈顶, 长度无限

- *pop*: 仅弹出栈顶的一个符号
- *push*: 可压入一串符号



6.1.1 形式定义

定义. 下推自动机 (PDA, Pushdown Automata) P 为七元组

$$P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$$

1. Q , 有穷状态集;

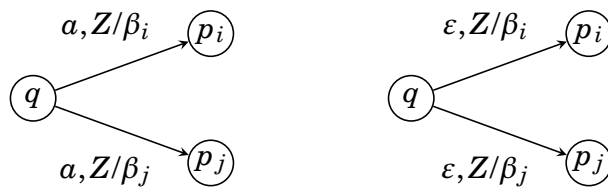
2. Σ , 有穷输入符号集 (即字母表);
3. Γ , 有穷栈符号集 (或栈字母表);
4. $\delta: Q \times (\Sigma \cup \{\epsilon\}) \times \Gamma \rightarrow 2^{Q \times \Gamma^*}$, 状态转移函数;
5. $q_0 \in Q$, 初始状态;
6. $Z_0 \in \Gamma - \Sigma$, 栈底符号, *PDA* 开始时, 栈中包含这个符号的一个实例, 用来表示栈底, 最初的栈底符号之下无任何内容;
7. $F \subseteq Q$, 接收状态集或终态集.

PDA 的动作和状态转移图

如果 $q, p_i \in Q$ ($1 \leq i \leq m$), $a \in \Sigma$, $Z \in \Gamma$, $\beta_i \in \Gamma^*$, 可以有动作:

$$\delta(q, a, Z) = \{(p_1, \beta_1), (p_2, \beta_2), \dots, (p_m, \beta_m)\}, \text{ 或}$$

$$\delta(q, \epsilon, Z) = \{(p_1, \beta_1), (p_2, \beta_2), \dots, (p_m, \beta_m)\}.$$



如果 q 和 p_i 是状态 ($1 \leq i \leq m$), 输入符号 $a \in \Sigma$, 栈符号 $Z \in \Gamma$, 栈符号串 $\beta_i \in \Gamma^*$, 那么

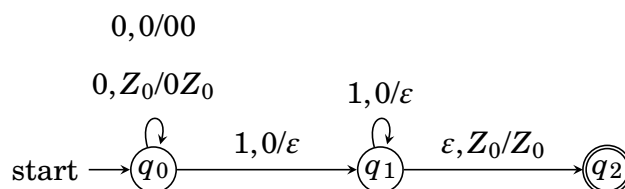
$$\delta(q, a, Z) = \{(p_1, \beta_1), (p_2, \beta_2), \dots, (p_m, \beta_m)\}$$

的意思是: 输入符号是 a , 栈顶符号 Z 的情况下, 处于状态 q 的 *PDA* 能够进入状态 p_i , 且用符号串 β_i 替换栈顶的符号 Z , 这里的 i 是任意的, 然后输入头前进一个符号. (约定 β_i 的最左符号在栈最上.) 但是若 $i \neq j$, 不能同时选择 p_i 和 β_j . 而

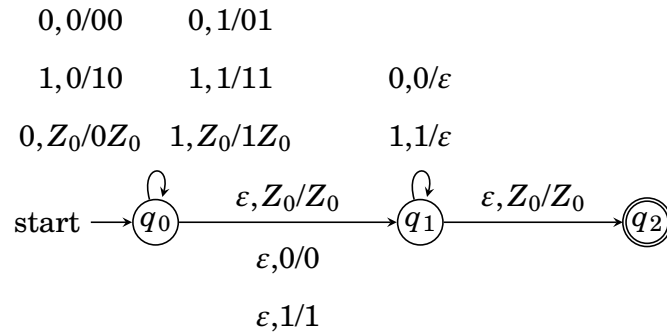
$$\delta(q, \epsilon, Z) = \{(p_1, \beta_1), (p_2, \beta_2), \dots, (p_m, \beta_m)\}$$

的意思是: 与扫描的输入符号无关, 只要 Z 是栈符号, 处于状态 q 的 *PDA*, 就可以进行上面的动作, 输入头不移动.

例 1. 设计识别 $L_{01} = \{0^n 1^n \mid n \geq 1\}$ 的 *PDA*.



例 2. 设计识别 $L_{ww^R} = \{ ww^R \mid w \in (0+1)^* \}$ 的 PDA.



1. 初始状态 q_0 , 栈顶 Z_0 , 无论输入 0 或 1 都直接压栈;
2. 继续压栈状态 q_0 , 则对不同输入 (0/1) 和不同栈顶 (0/1), 都直接压栈;
3. 非确定的转到弹栈状态 q_1 , 不论栈顶是 Z_0 , 0, 或 1, 开始匹配后半部分;
4. 保持弹栈状态 q_1 , 弹出的栈顶符号必须和输入一致;
5. 扫描到串结尾且只有看到栈底符号了, 才允许转移到接受状态 q_2 .

6.1.2 瞬时描述和转移符号

定义. 为形式描述 PDA 在一个给定瞬间的格局 (Configuration), 定义 $Q \times \Sigma^* \times \Gamma^*$ 中三元组

$$(q, w, \gamma)$$

为瞬时描述 (ID, Instantaneous Description), 表示此时 PDA 处于状态 q , 输入带上剩余输入串 w , 栈中的符号串为 γ .

定义. 在 PDA P 中如果 $(p, \beta) \in \delta(q, a, Z)$, 由 $(q, aw, Z\alpha)$ 到 $(p, w, \beta\alpha)$ 的变化, 称为 ID 的转移 \vdash_P , 记为

$$(q, aw, Z\alpha) \vdash_P (p, w, \beta\alpha)$$

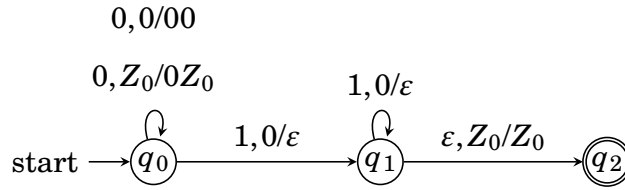
其中 $w \in \Sigma^*$, $\alpha \in \Gamma^*$.

若有 ID I, J 和 K , 递归定义 \vdash_P^* 为:

1. $I \vdash_P^* I$;
2. 若 $I \vdash_P J$, $J \vdash_P^* K$, 则 $I \vdash_P^* K$.

若 P 已知, 可省略, 记为 \vdash 和 \vdash^* .

续例 1. 语言 $L_{01} = \{0^n 1^n \mid n \geq 1\}$ 的 PDA, 识别 0011 时的 ID 序列.



有关 ID 的序列

对 PDA P 的一个合法 ID 序列 (计算):

1. 把相同字符串加到所有 ID 的输入串末尾, 得到的计算合法;
2. 把相同栈符号串加到所有 ID 的栈底之下, 得到的计算合法;
3. 把所有 ID 中都未消耗的部分输入串去掉, 得到的计算合法.

定理 23. 对 $\forall w \in \Sigma^*, \forall \gamma \in \Gamma^*$, 如果

$$(q, x, \alpha) \stackrel{*}{\vdash}_P (p, y, \beta),$$

那么

$$(q, xw, \alpha\gamma) \stackrel{*}{\vdash}_P (p, yw, \beta\gamma).$$

定理 24. 对 $\forall w \in \Sigma^*$, 如果

$$(q, xw, \alpha) \stackrel{*}{\vdash}_P (p, yw, \beta),$$

那么

$$(q, x, \alpha) \stackrel{*}{\vdash}_P (p, y, \beta).$$

6.2 下推自动机接受的语言

定义. PDA $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$, 以两种方式接受语言:

- P 以终态方式接受的语言, 记为 $\mathbf{L}(P)$, 定义为

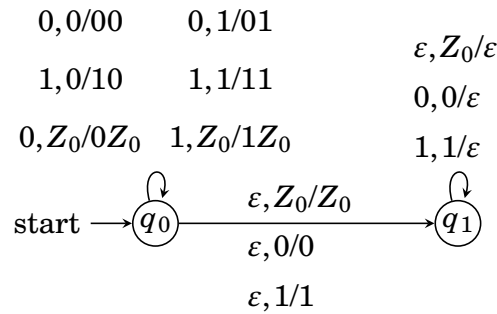
$$\mathbf{L}(P) = \{w \mid (q_0, w, Z_0) \stackrel{*}{\vdash} (p, \varepsilon, \gamma), p \in F\}.$$

- P 以空栈方式接受的语言, 记为 $\mathbf{N}(P)$, 定义为

$$\mathbf{N}(P) = \{w \mid (q_0, w, Z_0) \stackrel{*}{\vdash} (p, \varepsilon, \varepsilon)\}.$$

续例 2. 识别 L_{wwr} 的 PDA P , 从终态方式, 改为空栈方式接受.

用 $\delta(q_1, \varepsilon, Z_0) = \{(q_1, \varepsilon)\}$ 代替 $\delta(q_1, \varepsilon, Z_0) = \{(q_2, Z_0)\}$ 即可.

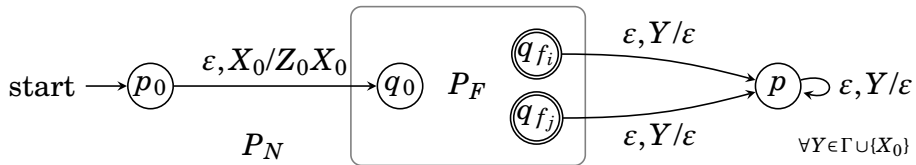


6.2.1 从终态方式到空栈方式

定理 25. 如果 PDA P_F 以终态方式接受语言 L , 则存在 PDA P_N 以空栈方式接受 L .

证明: 设 $P_F = (Q, \Sigma, \Gamma, \delta_F, q_0, Z_0, F)$, 构造 PDA P_N ,

$$P_N = (Q \cup \{p_0, p\}, \Sigma, \Gamma \cup \{X_0\}, \delta_N, p_0, X_0, \emptyset).$$



其中 δ_N 定义如下:

1. P_N 首先将 P_F 的栈底符号压栈, 开始模拟 P_F :

$$\delta_N(p_0, \varepsilon, X_0) = \{(q_0, Z_0X_0)\};$$

2. P_N 模拟 P_F 的动作: $\forall q \in Q, \forall a \in \Sigma \cup \{\varepsilon\}, \forall Y \in \Gamma$:

$$\delta_N(q, a, Y) \text{ 包含 } \delta_F(q, a, Y) \text{ 的全部元素};$$

3. 从 $q_f \in F$ 开始弹出栈中符号, 即 $\forall q_f \in F, \forall Y \in \Gamma \cup \{X_0\}$:

$$\delta_N(q_f, \varepsilon, Y) \text{ 包含 } (p, \varepsilon);$$

4. 在状态 p 时, 弹出全部栈中符号, 即 $\forall Y \in \Gamma \cup \{X_0\}$:

$$\delta_N(p, \varepsilon, Y) = \{(p, \varepsilon)\}.$$

对 $\forall w \in \Sigma^*$ 有

$$\begin{aligned}
 w \in \mathbf{L}(P_F) &\Rightarrow (q_0, w, Z_0) \vdash_{P_F}^* (q_f, \varepsilon, \gamma) \\
 &\Rightarrow (q_0, w, Z_0 X_0) \vdash_{P_F}^* (q_f, \varepsilon, \gamma X_0) && \text{定理23} \\
 &\Rightarrow (q_0, w, Z_0 X_0) \vdash_{P_N}^* (q_f, \varepsilon, \gamma X_0) && P_N \text{模拟 } P_F \\
 &\Rightarrow (p_0, w, X_0) \vdash_{P_N} (q_0, w, Z_0 X_0) \vdash_{P_N}^* (q_f, \varepsilon, \gamma X_0) && \delta_N \text{构造 } p_0 \text{ 部分} \\
 &\Rightarrow (p_0, w, X_0) \vdash_{P_N}^* (q_f, \varepsilon, \gamma X_0) \vdash_{P_N}^* (p, \varepsilon, \varepsilon) && \delta_N \text{构造 } q_f \text{ 和 } p \text{ 部分} \\
 &\Rightarrow w \in \mathbf{N}(P_N)
 \end{aligned}$$

即 $\mathbf{L}(P_F) \subseteq \mathbf{N}(P_N)$.

对 $\forall w \in \Sigma^*$ 有

$$\begin{aligned}
 w \in \mathbf{N}(P_N) &\Rightarrow (p_0, w, X_0) \vdash_{P_N}^* (p, \varepsilon, \varepsilon) && \text{其他状态不可能空栈} \\
 &\Rightarrow (p_0, w, X_0) \vdash_{P_N} (q_0, w, Z_0 X_0) \vdash_{P_N}^* (p, \varepsilon, \varepsilon) && \text{第一个动作必然到 } q_0 \\
 &\Rightarrow (q_0, w, Z_0 X_0) \vdash_{P_N}^* (q_f, \varepsilon, \gamma X_0) \vdash_{P_N}^* (p, \varepsilon, \varepsilon) && \text{必经 } q_f \in F \text{ 消耗完 } w \\
 &\Rightarrow (q_0, w, Z_0) \vdash_{P_N}^* (q_f, \varepsilon, \gamma) && P_N \text{ 中未用过栈底的 } X_0 \\
 &\Rightarrow (q_0, w, Z_0) \vdash_{P_F}^* (q_f, \varepsilon, \gamma) && \text{均为模拟 } P_F \\
 &\Rightarrow w \in \mathbf{L}(P_F)
 \end{aligned}$$

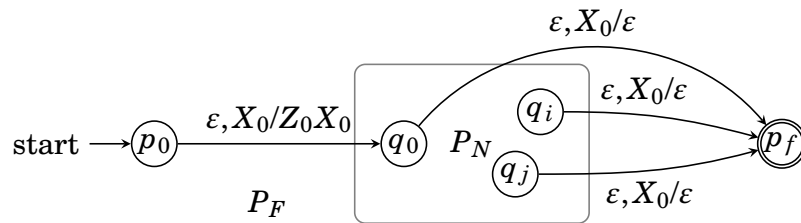
即 $\mathbf{N}(P_N) \subseteq \mathbf{L}(P_F)$. 所以 $\mathbf{N}(P_N) = \mathbf{L}(P_F)$. □

6.2.2 从空栈方式到终态方式

定理 26. 如果 PDA P_N 以空栈方式接受语言 L , 则存在 PDA P_F 以终态方式接受 L .

证明: 设 $P_N = (Q, \Sigma, \Gamma, \delta_N, q_0, Z_0, \emptyset)$. 构造 PDA P_F ,

$$P_F = (Q \cup \{p_0, p_f\}, \Sigma, \Gamma \cup \{X_0\}, \delta_F, p_0, X_0, \{p_f\})$$



其中 δ_F 定义如下:

1. P_F 开始时, 将 P_N 栈底符号压入栈, 并开始模拟 P_N ,

$$\delta_F(p_0, \varepsilon, X_0) = \{(q_0, Z_0 X_0)\};$$

2. P_F 模拟 P_N , $\forall q \in Q, \forall a \in \Sigma \cup \{\varepsilon\}, \forall Y \in \Gamma$:

$$\delta_F(q, a, Y) = \delta_N(q, a, Y);$$

3. 在 $\forall q \in Q$ 时, 看到 P_F 的栈底 X_0 , 则转移到新终态 p_f :

$$\delta_F(q, \varepsilon, X_0) = \{(p_f, \varepsilon)\}.$$

对 $\forall w \in \Sigma^*$ 有

$$\begin{aligned} w \in \mathbf{N}(P_N) &\Rightarrow (q_0, w, Z_0) \stackrel{*}{\vdash}_{P_N} (q, \varepsilon, \varepsilon) \\ &\Rightarrow (q_0, w, Z_0 X_0) \stackrel{*}{\vdash}_{P_N} (q, \varepsilon, X_0) && \text{定理23} \\ &\Rightarrow (q_0, w, Z_0 X_0) \stackrel{*}{\vdash}_{P_F} (q, \varepsilon, X_0) && P_F \text{ 模拟 } P_N \\ &\Rightarrow (p_0, w, X_0) \vdash_{P_F} (q_0, w, Z_0 X_0) \stackrel{*}{\vdash}_{P_F} (q, \varepsilon, X_0) && \delta_F \text{ 构造, } p_0 \text{ 部分} \\ &\Rightarrow (p_0, w, X_0) \stackrel{*}{\vdash}_{P_F} (q, \varepsilon, X_0) \vdash_{P_F} (p_f, \varepsilon, \varepsilon) && \delta_F \text{ 构造, } p_f \text{ 部分} \\ &\Rightarrow (p_0, w, X_0) \stackrel{*}{\vdash}_{P_F} (p_f, \varepsilon, \varepsilon) \\ &\Rightarrow w \in \mathbf{L}(P_F) \end{aligned}$$

即 $\mathbf{N}(P_N) \subseteq \mathbf{L}(P_F)$.

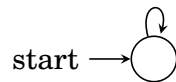
对 $\forall w \in \Sigma^*$ 有

$$\begin{aligned} w \in \mathbf{L}(P_F) &\Rightarrow (p_0, w, X_0) \stackrel{*}{\vdash}_{P_F} (p_f, \varepsilon, \varepsilon) \\ &\Rightarrow (p_0, w, X_0) \stackrel{*}{\vdash}_{P_F} (q, \varepsilon, X_0) \vdash_{P_F} (p_f, \varepsilon, \varepsilon) && \text{经 } q \text{ 才可达 } p_f \\ &\Rightarrow (p_0, w, X_0) \vdash_{P_F} (q_0, w, Z_0 X_0) \stackrel{*}{\vdash}_{P_F} (q, \varepsilon, X_0) && P_F \text{ 第一个动作} \\ &\Rightarrow (q_0, w, Z_0 X_0) \stackrel{*}{\vdash}_{P_F} (q, \varepsilon, X_0) && \text{即上式} \\ &\Rightarrow (q_0, w, Z_0) \stackrel{*}{\vdash}_{P_N} (q, \varepsilon, \varepsilon) && P_N \text{ 与 } X_0 \text{ 无关} \\ &\Rightarrow w \in \mathbf{N}(P_N) \end{aligned}$$

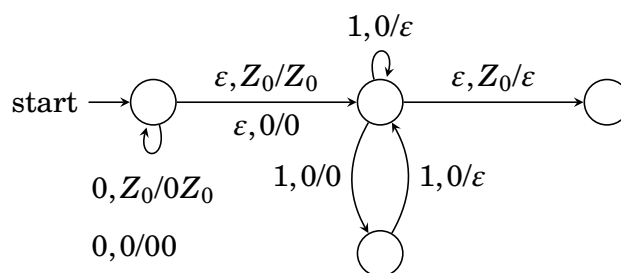
即 $\mathbf{N}(P_F) \subseteq \mathbf{L}(P_N)$. 所以 $\mathbf{L}(P_F) = \mathbf{N}(P_N)$. □

例 3. 接受 $L_{\text{eq}} = \{w \in \{0, 1\}^* \mid w \text{ 中字符 } 0 \text{ 和 } 1 \text{ 的数量相同}\}$ 的 PDA.

$$\begin{array}{lll} 0, Z_0/0Z_0 & 1, 0/10 & 0, 0/00 \\ 1, Z_0/1Z_0 & 1, 1/11 & 0, 1/01 \\ \varepsilon, Z_0/\varepsilon & 1, 0/\varepsilon & 0, 1/\varepsilon \end{array}$$



例 4. 接受 $L = \{0^n 1^m \mid 0 \leq n \leq m \leq 2n\}$ 的 PDA.



课堂练习. Design PDA for $L = \{a^i b^j c^k \mid i, j, k \geq 0, j = i + k\}$.

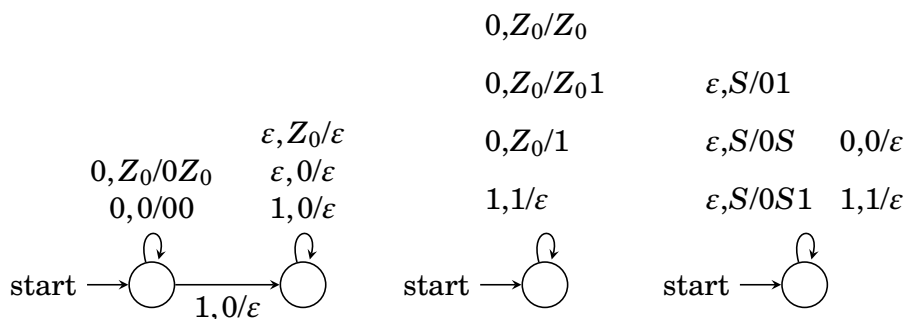
例. Design PDA for $L = \{a^i b^j c^k \mid i, j, k \geq 0, i = j \text{ or } j = k\}$.

例. Design PDA for the set of strings of 0's and 1's such that no prefix has more 1's than 0's.

6.3 下推自动机与文法的等价性

6.3.1 由 CFG 到 PDA

例 5. 设计语言 $L = \{0^n 1^m \mid 1 \leq m \leq n\}$ 的 PDA.



CFG G :

$$S \rightarrow AB$$

$$A \rightarrow 0A \mid \varepsilon$$

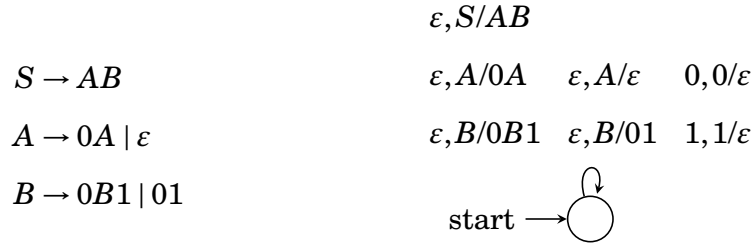
$$B \rightarrow 0B1 \mid 01$$

字符串 00011 的最左派生:

$$S \xRightarrow{\text{lm}} AB \xRightarrow{\text{lm}} 0AB \xRightarrow{\text{lm}} 0B \xRightarrow{\text{lm}} 00B1 \xRightarrow{\text{lm}} 00011$$

用 PDA 栈顶符号的替换, 模拟文法的最左派生:

PDA		CFG	
PDA 的 ID 转移	PDA 的动作	产生式	最左派生
$(q_0, 00011, S)$			S
$\vdash (q_0, 00011, AB)$	$\varepsilon, S/AB$	$S \rightarrow AB$	$\Rightarrow_{\text{lm}} AB$
$\vdash (q_0, 00011, 0AB)$	$\varepsilon, A/0A$	$A \rightarrow 0A$	$\Rightarrow_{\text{lm}} 0AB$
$\vdash (q_0, 0011, AB)$	$0, 0/\varepsilon$		
$\vdash (q_0, 0011, B)$	$\varepsilon, A/\varepsilon$	$A \rightarrow \varepsilon$	$\Rightarrow_{\text{lm}} 0B$
$\vdash (q_0, 0011, 0B1)$	$\varepsilon, B/0B1$	$B \rightarrow 0B1$	$\Rightarrow_{\text{lm}} 00B1$
$\vdash (q_0, 011, B1)$	$0, 0/\varepsilon$		
$\vdash (q_0, 011, 011)$	$\varepsilon, B/01$	$B \rightarrow 01$	$\Rightarrow_{\text{lm}} 00011$
$\vdash (q_0, 11, 11)$	$0, 0/\varepsilon$		
$\vdash (q_0, 1, 1)$	$1, 1/\varepsilon$		
$\vdash (q_0, \varepsilon, \varepsilon)$	$1, 1/\varepsilon$		



想要证明 CFG 和 PDA 的等价性, 需要思考如何使用 PDA 模拟文法的推导. 对任意属于某 CFL 的串 w , 其文法的推导过程, 就是使用产生式去匹配 (产生) w , 如果 w 放在某 PDA 的输入带上, 我们的目的就是通过文法构造动作, 让 PDA 能从左到右的扫描输入串, 利用栈来模拟文法的最左派生过程即可.

定理 27. 任何 CFL L , 一定存在 PDA P , 使 $L = \mathbf{N}(P)$.

构造与文法等价的 PDA

如果 CFG $G = (V, T, P', S)$, 构造 PDA

$$P = (\{q\}, T, V \cup T, \delta, q, S, \emptyset),$$

其中 δ 为:

1. $\forall A \in V$:

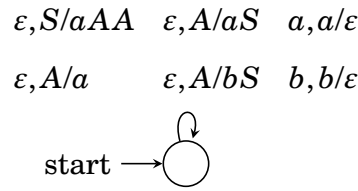
$$\delta(q, \varepsilon, A) = \{(q, \beta) \mid A \rightarrow \beta \in P'\},$$

2. $\forall a \in T$:

$$\delta(q, a, a) = \{(q, \varepsilon)\},$$

那么 $L(G) = N(P)$.

例 6. 为文法 $S \rightarrow aAA, A \rightarrow aS \mid bS \mid a$ 构造 PDA.



证明:

PDF P 可以模拟 CFG G 的最左派生, 每个动作只根据栈顶的符号确定: 如果是终结符则与输入串匹配, 如果是非终结符用产生式来替换. 为了完成定理, 只需往证

$$S \xRightarrow{*} w \iff (q, w, S) \vdash^* (q, \varepsilon, \varepsilon).$$

最左派生 $S \xRightarrow{*} w$ 的每个左句型都可写作 $xA\alpha$ 的形式, 其中 A 是最左变元, x 是它之前的所有终结符串, 而 α 是它右边的符号串. 而在 P 中, 每个左句型的 $A\alpha$ 部分都会出现在栈中, 并且当 A 处于栈顶时, x 刚好是输入带上被扫描过的 (消耗完的) 部分. 那么当 PDA 处于 $ID(q, y, A\alpha)$ 时, 刚好有 $xy = w$ 成立.

[充分性] 往证

$$S \xRightarrow{*} w \implies (q, w, S) \vdash^* (q, \varepsilon, \varepsilon).$$

设 $S \xRightarrow{*} w$ 中第 i 个左句型为 $x_i A_i \alpha_i$, 其中 $x_i \in \Sigma^*$, $A_i \in V$, $\alpha_i \in (V \cup T)^*$. 并将 S 看作第 0 个左句型 $x_0 A_0 \alpha_0 = S$, 那么

$$x_0 = \varepsilon, A_0 = S, \alpha_0 = \varepsilon.$$

将 w 看作为第 n 个左句型 $x_n A_n \alpha_n = w$, 那么

$$x_n = w, A_n = \varepsilon, \alpha_n = \varepsilon.$$

再对派生步骤 i 归纳, 往证

$$S \xRightarrow{i} x_i A_i \alpha_i \wedge w = x_i y_i \implies (q, w, S) \vdash^* (q, y_i, A_i \alpha_i).$$

归纳基础: 最左派生在第 0 步时, 显然成立

$$(q, w, S) \vdash^* (q, y_0, A_0 \alpha_0) = (q, w, S).$$

归纳递推: 假设第 i 步时成立, 当第 $i+1$ 步时, 一定是 $A_i \rightarrow \beta$ 应用到 $x_i A_i \alpha_i$

$$S \xRightarrow{i} x_i A_i \alpha_i \xRightarrow{} x_i \beta \alpha_i = x_{i+1} A_{i+1} \alpha_{i+1}.$$

即第 $i+1$ 个左句型的最左变元 A_{i+1} 一定在 $\beta\alpha_i$ 中, 设 A_{i+1} 之前的终结符为 x' , 那么由

$$\begin{aligned} x_i\beta\alpha_i &= x_ix'A_{i+1}\alpha_{i+1} = x_{i+1}A_{i+1}\alpha_{i+1} \\ x_iy_i &= x_ix'y_{i+1} = x_{i+1}y_{i+1} = w \end{aligned}$$

则有

$$\begin{aligned} \beta\alpha_i &= x'A_{i+1}\alpha_{i+1}, \\ y_i &= x'y_{i+1}. \end{aligned}$$

那么, 在 PDA 中从 ID $(q, y_i, A_i\alpha_i)$ 模拟最左派生, 用产生式 $A_i \rightarrow \beta$ 替换栈顶 A_i 后, 有

$$\begin{array}{ll} (q, w, S) \vdash^* (q, y_i, A_i\alpha_i) & \text{归纳假设} \\ \vdash (q, y_i, \beta\alpha_i) & A_i \rightarrow \beta \\ = (q, x'y_{i+1}, x'A_{i+1}\alpha_{i+1}) & \\ \vdash^* (q, y_{i+1}, A_{i+1}\alpha_{i+1}) & \text{弹出栈顶终结符} \end{array}$$

因此 $S \xRightarrow{*} w \implies (q, w, S) \vdash^* (q, y_n, A_n\alpha_n) = (q, \varepsilon, \varepsilon)$, 即充分性得证.

[必要性] 往证更一般的, 对任何变元 A , 都有:

$$(q, x, A) \vdash^* (q, \varepsilon, \varepsilon) \implies A \xRightarrow{*} \varepsilon.$$

这个过程, 可以看作“从输入带中消耗掉 x ”与“从栈中弹出 A ”两种作用相互抵消. 对 ID 转移 $(q, x, A) \vdash (q, \varepsilon, \varepsilon)$ 的次数 i 归纳证明.

归纳基础: 当 $i = 1$ 步时, 只能是 $x = \varepsilon$ 且 $A \rightarrow \varepsilon$ 为产生式, 所以 $A \xRightarrow{*} \varepsilon$. 因为即使 $x = a$ 和产生式 $A \rightarrow a$, 也需要先替换栈顶 A 为 a 再弹出 a 两步才能清空栈.

归纳递推: 假设 $i \leq n$ ($n \geq 1$) 步时上式成立. 当 $i = n+1$ 时, 因为 A 是变元, 其第 1 步转移一定是应用某产生式 $A \rightarrow Y_1Y_2\cdots Y_m$

$$(q, x, A) \vdash (q, x, Y_1Y_2\cdots Y_m)$$

其中 Y_i 是变元或终结符. 而其余的 n 步转移

$$(q, x, Y_1Y_2\cdots Y_m) \vdash^* (q, \varepsilon, \varepsilon)$$

中每个 Y_i 从栈中被完全弹出时, 将消耗掉的那部分 x 记为 x_i , 那么显然有

$$x = x_1x_2\cdots x_m.$$

而每个 Y_i 从栈中被完全弹出时, 都不超过 n 步, 所以由归纳假设,

$$(q, x_i, Y_i) \vdash^* (q, \varepsilon, \varepsilon) \implies Y_i \xRightarrow{*} \varepsilon.$$

再由产生式 $A \rightarrow Y_1 Y_2 \cdots Y_m$, 有

$$\begin{aligned} A &\Rightarrow Y_1 Y_2 \cdots Y_m \\ &\stackrel{*}{\Rightarrow} x_1 Y_2 \cdots Y_m \\ &\stackrel{*}{\Rightarrow} x_1 x_2 \cdots Y_m \\ &\stackrel{*}{\Rightarrow} x_1 x_2 \cdots x_m = x. \end{aligned}$$

因此当 $A = S$, $x = w$ 时,

$$(q, w, S) \vdash^* (q, \varepsilon, \varepsilon) \implies S \stackrel{*}{\Rightarrow} w$$

成立, 即必要性得证.

所以, 任何 CFL 都可由 PDA 识别. □

构造与 GNF 格式文法等价的 PDA

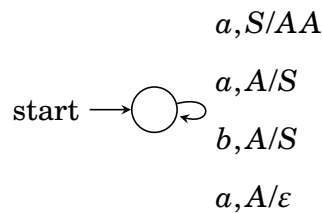
如果 GNF 格式的 CFG $G = (V, T, P', S)$, 那么构造 PDA

$$P = (\{q\}, T, V, \delta, q, S, \emptyset),$$

为每个产生式, 定义 δ 为:

$$\delta(q, a, A) = \{(q, \beta) \mid A \rightarrow a\beta \in P'\}.$$

续例 6. 文法 $S \rightarrow aAA$, $A \rightarrow aS \mid bS \mid a$ 为 GNF 格式, 构造等价的 PDA.



6.3.2 由 PDA 到 CFG

定理 28. 如果 PDA P , 有 $L = N(P)$, 那么 L 是上下文无关语言.

构造与 PDA 等价的 CFG

如果 PDA $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, \emptyset)$, 那么构造 CFG $G = (V, \Sigma, P', S)$, 其中 V 和 P' 为

1. $V = \{[qXp] \mid p, q \in Q, X \in \Gamma\} \cup \{S\};$

2. 对 $\forall p \in Q$, 构造产生式 $S \rightarrow [q_0 Z_0 p]$;
3. 对 $\forall (p, Y_1 Y_2 \cdots Y_n) \in \delta(q, a, X)$, 构造 $|Q|^n$ 个产生式

$$[q X r_n] \rightarrow a[p Y_1 r_1][r_1 Y_2 r_2] \cdots [r_{n-1} Y_n r_n]$$

其中 $a \in \Sigma \cup \{\epsilon\}$, $X, Y_i \in \Gamma$, 而 $r_i \in Q$ 是 n 次 $|Q|$ 种状态的组合; 若 $n = 0$, 为 $[q X p] \rightarrow a$.

证明: 通过以上方法所构造的文法中, $[q X p] \Rightarrow^* w$ 的含义, 可以理解为 PDA 从状态 q 将栈符号 X 完全弹出后, 转移到了状态 p , 同时从输入带上消耗掉了 w . 那么, 只需证明

$$(q, w, X) \vdash^* (p, \epsilon, \epsilon) \iff [q X p] \Rightarrow^* w.$$

并令 $X = Z_0$, $q = q_0$, 与开始符号 S 的产生式一起, 即可完成定理的证明.

[充分性] 对 PDA 中 $(q, w, X) \vdash^* (p, \epsilon, \epsilon)$ 的转移次数 i 归纳证明.

归纳基础: 当 $i = 1$ 时, P 的输入带上只能消耗不超过一个的字符, 即 $w = a$

$$(q, w, X) = (q, a, X) \vdash (p, \epsilon, \epsilon),$$

其中 $a \in \Sigma \cup \{\epsilon\}$ 且 $(p, \epsilon) \in \delta(q, a, X)$, 则由文法的构造会有

$$[q X p] \rightarrow a,$$

因此 $[q X p] \Rightarrow^* a = w$.

归纳递推: 假设当 $i \leq m$ ($m \geq 1$) 时命题成立. 那么, 当 $i = m + 1$ 时, 转移的第 1 步, 一定由某个 $(r_0, Y_1 Y_2 \cdots Y_n) \in \delta(q, a, X)$ 开始

$$(q, ax, X) \vdash (r_0, x, Y_1 Y_2 \cdots Y_n),$$

其中 $a \in \Sigma \cup \{\epsilon\}$, $w = ax$. 而其余的 m 步为

$$(r_0, x, Y_1 Y_2 \cdots Y_n) \vdash^* (p, \epsilon, \epsilon).$$

而这些转移, 会从栈中依次弹出 Y_i , 并相应的消耗掉输入带上的部分 x . 若分别记为 x_i , 则有

$$w = ax = ax_1 x_2 \cdots x_n.$$

若设从栈中完全弹出 Y_i (并消耗掉 x_i) 之前和之后的状态分别是 r_{i-1} 和 r_i , 这里 $i = 1, 2, \cdots, n$, 那么有

$$(r_{i-1}, x_i, Y_i) \vdash^* (r_i, \epsilon, \epsilon),$$

且转移步数都不会超过 m . 那么, 由归纳假设有

$$(r_{i-1}, x_i, Y_i) \vdash^* (r_i, \epsilon, \epsilon) \implies [r_{i-1} Y_i r_i] \Rightarrow^* x_i.$$

而由动作 $(r_0, Y_1 Y_2 \cdots Y_n) \in \delta(q, a, X)$ 所构造的产生式会包含

$$[qXr_n] \rightarrow a[r_0Y_1r_1][r_1Y_2r_2] \cdots [r_{n-1}Y_nr_n].$$

而显然弹出 X 后的状态 p 与弹出 Y_n 后的状态 r_n 是同一个, 所以

$$[qXp] = [qXr_n] \Rightarrow a[r_0Y_1r_1][r_1Y_2r_2] \cdots [r_{n-1}Y_nr_n] \xRightarrow{*} ax_1x_2 \cdots x_n = w$$

因此充分性得证. 那么当 $X = Z_0, q = q_0$ 时有

$$(q_0, w, Z_0) \vdash^* (p, \varepsilon, \varepsilon) \Longrightarrow [q_0Z_0p] \xRightarrow{*} w,$$

以及产生式 $S \rightarrow [q_0Z_0p]$ 有 $S \xRightarrow{*} w$, 即 PDA 接受的串可由文法派生得到.

[必要性]: 略. □

例 7. 将 PDA $P = (\{p, q\}, (0, 1), \{X, Z\}, \delta, q, Z)$ 转为 CFG, 其中 δ 如下:

- (1) $\delta(q, 1, Z) = \{(q, XZ)\}$
- (2) $\delta(q, 1, X) = \{(q, XX)\}$
- (3) $\delta(q, 0, X) = \{(p, X)\}$
- (4) $\delta(q, \varepsilon, Z) = \{(q, \varepsilon)\}$
- (5) $\delta(p, 1, X) = \{(p, \varepsilon)\}$
- (6) $\delta(p, 0, Z) = \{(q, Z)\}$

解:

δ	产生式		
(0)	$S \rightarrow [qZq]$ $S \rightarrow [qZp]$		
(1)	$[qZq] \rightarrow 1[qXq][qZq]$ $[qZq] \rightarrow 1[qXp][pZq]$ $[qZp] \rightarrow 1[qXq][qZp]$ $[qZp] \rightarrow 1[qXp][pZp]$	消除无用符号	重命名 (可选)
(2)	$[qXq] \rightarrow 1[qXq][qXq]$ $[qXq] \rightarrow 1[qXp][pXq]$ $[qXp] \rightarrow 1[qXq][qXp]$ $[qXp] \rightarrow 1[qXp][pXp]$	$S \rightarrow [qZq]$ $[qZq] \rightarrow 1[qXp][pZq]$ $[qXp] \rightarrow 1[qXp][pXp]$ $[qXp] \rightarrow 0[pXp]$ $[qZq] \rightarrow \varepsilon$ $[pXp] \rightarrow 1$	$S \rightarrow A$ $A \rightarrow 1BC$ $B \rightarrow 1BD$ $B \rightarrow 0D$ $A \rightarrow \varepsilon$ $D \rightarrow 1$
(3)	$[qXq] \rightarrow 0[pXq]$ $[qXp] \rightarrow 0[pXp]$	$[pZq] \rightarrow 0[qZq]$	$C \rightarrow 0A$
(4)	$[qZq] \rightarrow \varepsilon$		
(5)	$[pXp] \rightarrow 1$		
(6)	$[pZp] \rightarrow 0[qZp]$ $[pZq] \rightarrow 0[qZq]$		

6.4 确定型下推自动机

定义. 如果 $PDA P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ 满足

1. $\forall a \in \Sigma \cup \{\varepsilon\}, \delta(q, a, X)$ 至多有一个动作;
2. $\exists a \in \Sigma$, 如果 $\delta(q, a, X) \neq \emptyset$, 那么 $\delta(q, \varepsilon, X) = \emptyset$.

则称 P 为确定型下推自动机 (DPDA).

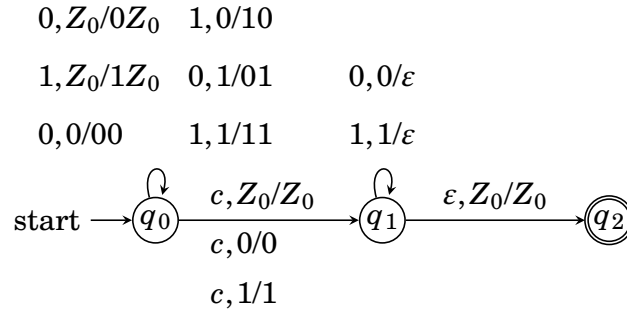
定义. DPDA P 以终态方式接受的语言 $\mathbf{L}(P)$ 称为确定的上下文无关语言 (DCFL).

- DPDA 中 $\forall (q, a, Z) \in Q \times \Sigma \times \Gamma$ 满足 $|\delta(q, a, Z)| + |\delta(q, \varepsilon, Z)| \leq 1$

DPDA 与 PDA 不等价

例 8. 任何 DPDA 都无法接受 L_{w^R} , 但是可以接受

$$L_{wcw^R} = \{wcw^R \mid w \in (\mathbf{0} + \mathbf{1})^*\}.$$



DPDA 无法识别 L_{wwr} . 因为, 如果它想识别输入串 $0^n 110^n$, 需要利用栈保存 n 个 0, 再根据 11 改变状态, 将栈弹出. 而这时如果输入带上还有 $0^n 110^n$, 那么它还应该接受. 但如果接受 $0^n 110^n 0^n 110^n$, 它同样会接受 $0^n 110^n 0^m 110^m$. DPDA 使用栈无法同时记住 0^n 和 $0^n 110^n$.

6.4.1 正则语言与 DPDA

定理 29. 如果 L 是正则语言, 那么存在 DPDA P 以终态方式接受 L , 即 $L = \mathbf{L}(P)$.

证明: 显然, 因为 DPDA P 可以不用栈而模拟任何 DFA. □

- L_{wwr} 显然是 CFL, 所以 DCFL 语言类真包含正则语言
- DPDA 无法识别 L_{wwr} , 所以 DCFL 语言类真包含于 CFL

定义. 如果语言 L 中不存在两个不同的字符串 x 和 y , 使 x 是 y 的前缀, 称语言 L 满足前缀性质.

定理 30. 如果有 DPDA P 且 $L = \mathbf{N}(P)$, 当且仅当 L 有前缀性质且存在 DPDA P' 使 $L = \mathbf{L}(P')$.

证明: $[\Rightarrow]$ $\forall x \in \mathbf{N}(P)$ 会弹空 P 的栈, 所以不会接受以 x 为前缀的其他串; 而转换为终态方式不改变确定性. $[\Leftarrow]$ 到达终态则弹空栈, 即可. □

- DPDA P 的 $\mathbf{N}(P)$ 更有限, 即使正则语言 0^* 也无法接受

DPDA P 若以空栈方式接受, 能够接受的语言更有限, 仅能接受具有前缀性质的语言. 前缀性质是指, 这个语言中不存在不同的串 x 和 y 使 x 是 y 的前缀. 即使正则语言 0^* 也无法接受, 因为其任何两个串中都有一个前缀. 但以空栈方式接受的语言, 却可以被另一个 DPDA 以终态方式接受.

6.4.2 DPDA 与无歧义文法

定理 31. DPDA P , 语言 $L = \mathbf{N}(P)$, 那么 L 有无歧义的 CFG.

证明: 利用定理 28 由 P 构造的文法 G 一定无歧义, 因为:

1. P 是确定的, 那么它接受 w 的 ID 序列也是确定的;
2. 而由 $\delta(q, a, X) = \{(p, Y_1 \cdots Y_n)\}$ 继续弹出 Y_i 后的状态 r_i 也是确定的;
3. 那么由每个动作构造的一组产生式

$$[qXr_n] \rightarrow a[pY_1r_1][r_1Y_2r_2] \cdots [r_{n-1}Y_nr_n]$$

中, 仅会有一个有效的;

4. 那么, G 中最左派生 $S \xrightarrow{*}_{\text{lm}} w$ 就是唯一的, 所以是无歧义的. □

定理 32. DPDA P , 语言 $L = \mathbf{L}(P)$, 那么 L 有无歧义的 CFG.

证明:

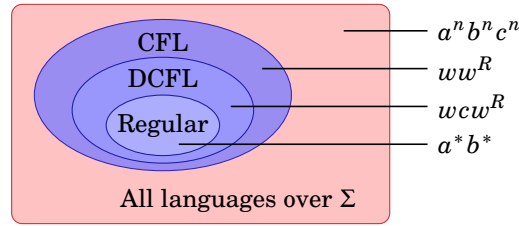
1. 设符号 $\$$ 不在 L 中出现, 令 $L' = \{w\$ \mid w \in L\}$, 则 L' 具有前缀性质;
2. 可修改 P 接受 L' , 则由定理 30, 存在 DPDA P' 使 $\mathbf{N}(P') = L'$;
3. 由定理 31, 存在无歧义文法 G' 使 $\mathbf{L}(G') = L'$;
4. 将 $\$$ 看作变元, 增加产生式 $\$ \rightarrow \epsilon$, 修改 G' 为文法 G ;
5. 则文法 G 和 G' 一样无歧义, 且 $\mathbf{L}(G) = L$. □

DCFL/DPDA 的重要应用

- 程序设计语言的语法分析器
如 $\text{LR}(k)$ 文法, Yacc 的基础, 解析的时间复杂度为 $O(n)$ 的算法
- 非固有歧义语言的真子集
如 L_{wwr} 有无歧义文法 $S \rightarrow 0S0 \mid 1S1 \mid \epsilon$

在任何情况下都不需要去选择可能的移动就是 DPDA, 以终态方式接受的语言也称为 DCFL. 虽然与 PDA 不等价, 但也有意义, 例如语法分析器通常都是 DPDA, DPDA 接受的语言是非固有歧义语言的真子集, Knuth 提出 $\text{LR}(k)$ 文法的语言也恰好是 DPDA 接受语言的一个子集, 解析的时间复杂度为 $O(n)$, $\text{LR}(k)$ 文法也是 YACC 的基础.

语言类之间的关系



6.5 练习题

1. [Exercise 6.2.1] Design a PDA to accept each of the following languages. You may accept either by final state or by empty stack, whichever is more convenient.
 - a) $\{0^n 1^n | n \geq 1\}$
 - b) The set of all strings of 0's and 1's such that no prefix has more 1's than 0's.
 - c) The set of all strings of 0's and 1's with an equal number of 0's and 1's.
2. [Exercise 6.2.2] Design a PDA to accept each of the following languages.
 - a) $\{a^i b^j c^k | i = j \text{ or } j = k\}$. 略
 - b) The set of all strings with twice as many 0's and 1's. (0 的个数是 1 的两倍)
3. [Exercise 6.2.3] Design a PDA to accept each of the following languages.
 - a) $\{a^i b^j c^k | i \neq j \text{ or } j \neq k\}$.
 - b) The set of all strings of a 's and b 's that are *not* of the form ww , that is, not equal to any string repeated.
4. [Exercise 6.3.5] Below are some context-free languages. For each, devise a PDA that accepts the language by empty stack. You may, if you wish, first construct a grammar for the language, and then convert to a PDA.
 - a) $\{a^n b^m c^{2(n+m)} | n \geq 0, m \geq 0\}$.
 - b) $\{a^i b^j c^k | i = 2j \text{ or } j = 2k\}$.
 - c) $\{0^n 1^m | n \leq m \leq 2n\}$ 和 $\{0^n 1^m | n < m < 2n\}$
5. Construct pushdown automata for the following languages. Acceptance either by empty stack or by final state.
 - (a) $\{a^n b^m a^n | m, n \in \mathbf{N}\}$

(b) $\{a^n b^m c^m \mid m, n \in \mathbf{N}\}$

(c) $\{a^i b^j c^k \mid i, j, k \in \mathbf{N}, i > j\}$

(d) $\{a^i b^j c^k \mid i, j, k \in \mathbf{N}, i + j = k\}$

(e) $\{a^i b^j c^k \mid i, j, k \in \mathbf{N}, i + k = j\}$

chunyu@hit.edu.cn

<http://nclab.net/~chunyu>

