第十一讲

✧ 上下文无关语言的性质与运算

上下文无关语言的性质与运算

- ✧ 针对上下文无关语言的 *Pumping* 引理
- ✧ 有关上下文无关语言的几个判定性质
- ✧ 关于上下文无关语言的封闭运算

针对上下文无关语言的Pumping引理

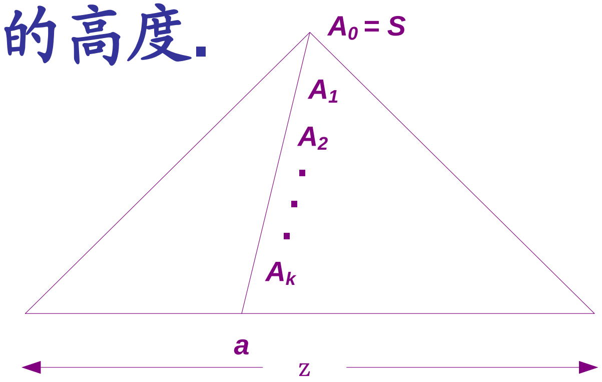
- 上下文无关语言应满足的一个必要条件
- 可用于判定某些语言不是上下文无关语言

✧ 上下文无关语言 的 "Pumping"特性

– "pumping" 特性：先讨论不包含 ε 的非空上下文无关语言 L，并设 CFG $G = (V, T, P, S)$ 为满足 CNF 的文法.

设 $|V|=m$，以及 $n=2^m$. 对于任一长度不小于 n 的字符串 z，即 $|z|\geq n$，考察关于 z 的分析树 S. 由于文法满足 CNF，该分析树为二叉树，其叶结点的个数为 $|z|$. 如右下图所示.

容易证明，$|z|\leq 2^{h-1}$，这里 h 为树 S 的高度.



设从根结点 S 开始的一条最长路径标记为 $A_0A_1A_2\ldots A_ka$.

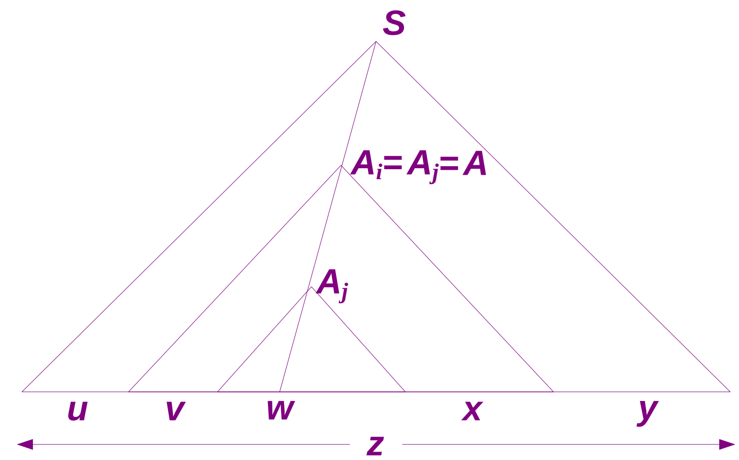
由于 $|z| \geq n = 2^m$，所以该分析树的高度至少为 $m+1$. 因而，$k\geq m$.

但$|V|=m$，因此 A_{k-m} ，A_{k-m+1} ，… ，A_{k-1} ，A_k 中必有重复的非终结符. 假设 $A_i=A_j$ ，其中 $k-m \leq i < j \leq k$.

针对上下文无关语言的Pumping引理

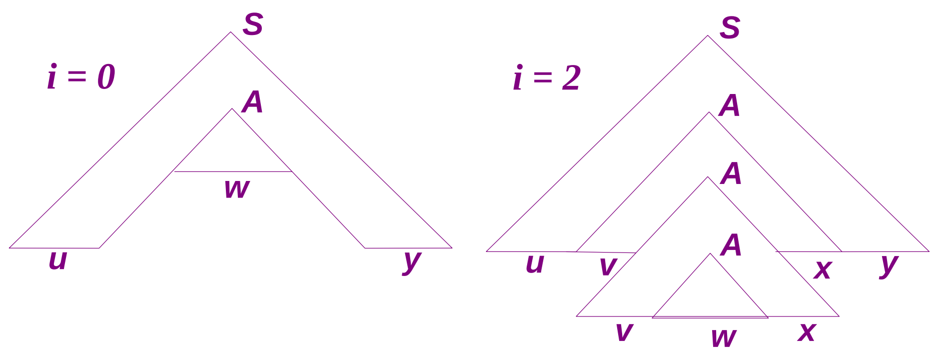
✧ 上下文无关语言 的 “Pumping”特性

– **“pumping” 特性（续前页）**



这样，*z* 的分析树可示意如右图. 可以将 *z* 划分为 *z=uvwxy* . *w* 由根为 A_j 的子树产生，*vwx* 由根为 A_i 的子树产生. 由于没有 unit 产生式，所以 $vx \neq \varepsilon$. 又因为根为 A_i 的子树高度不超过 *m*+1，所以 *vwx* 的长度不超过 $2^m = n$ ，即 $|vwx| \leq n$.

现在，可以对 *v* 和 *x* 进行 *pumped*，左下图是 *i*=0,2 的情形.



“pumping” 特性
对任意的 $i \geq 0$，
$uv^iwx^iy \in L$.

针对上下文无关语言的Pumping引理

✧ ***Pumping Lemma for Context-free Languages***

设 L 是上下文无关语言, 则存在正常数 n, 使得任一长度不小于n 的字符串 $z \in L$, $|z| \geq n$, 都可以分成 5 部分, 即 $z=uvwxy$, 满足下列条件:

1. $vx \neq \varepsilon$.
2. $|vwx| \leq n$.
3. 对任何 $k \geq 0$, 都有 $uv^k wx^k y \in L$.

✧ 证明　若 L-{ε} 为 Φ, 结论自然成立.
否则, 设 CFG $G = (V, T, P, S)$ 为 L-{ε} 的一个满足CNF 的文法, 只要取 $n = 2^{|V|}$ 即可.

针对上下文无关语言的Pumping引理

- ✧ **Pumping 引理的一个应用**
  - – 用于证明某个语言 *L* 不是上下文无关语言

Pumping 引理的条件可形式表示为:

$$\exists n\,\forall z\exists u\exists v\exists w\exists x\exists y\,\forall k(z\in L\wedge|z|\geq n>0\rightarrow z=uvwxy$$
$$\wedge vx\neq\varepsilon\wedge|vwx|\leq n\wedge(k\geq 0\rightarrow uv^kwx^ky\in L))$$

该命题的否定形式为:

$$\forall n\exists z\,\forall u\,\forall v\,\forall w\,\forall x\,\forall y\exists k(z\in L\wedge|z|\geq n>0\wedge(z=uvwxy$$
$$\wedge vx\neq\varepsilon\wedge|vwx|\leq n\rightarrow k\geq 0\wedge uv^kwx^ky\notin L))$$

✧ Pumping 引理的一个应用

– 用于证明某个语言 *L* 不是上下文无关语言（接上页）

– 证明步骤

1. 考虑任意的$n>0$.
2. 找到一个满足以下条件的串$z\in L$ (长度至少为n).
3. 任选满足$z=uvwxy \wedge vx\neq\varepsilon \wedge |vwx| \leq n$ 的u,v,w,x,y.
4. 找到一个 $k \geq 0$，使 $uv^kwx^ky \notin L$.

针对上下文无关语言的Pumping引理

✧ **Pumping 引理的一个应用**

– 用于证明某个语言 L 不是上下文无关语言（接上页）

✧ **举例** 证明语言 $L_{012} = \{ 0^k1^k2^k \mid k \geq 1 \}$ 不是上下文无关语言

证明 对任意$n>0$，取 $z = 0^n1^n2^n$. 任选满足条件 $z=uvwxy \wedge vx \neq \varepsilon \wedge |vwx| \leq n$ 的u,v,w,x,y
若取 $k=0$, 则有 $uv^kwx^ky = uwy \notin L_{012}$.

针对上下文无关语言的Pumping引理

- **Pumping 引理不是上下文无关语言的充分条件**

  – 反例 ***a*, *b*, *c*, *d*** 串构成的语言

  $$L = \{ a^i b^j c^k d^l \mid i, j, k, l \geq 0, \text{若 } i \neq 0 \text{ 则 } j=k=l \}$$

有关上下文无关语言的判定性质

- 有关几个转换问题的复杂度（选讲）
- 判定上下文无关语言是否为空
- 判定上下文无关语言中是否包含特定的字符串
- 有关上下文无关语言的几个不可判定问题（选讲）

有关上下文无关语言的判定性质

✧ 有关 CFG 和 PDA 的几个转换问题的复杂度

– CFG 变换为符合 Chomsky 范式

- 消去无用符号 计算可达符号和生成符号集合为 $O(n^2)$ 复杂度，但若采用适当的数据结构（见 7.4.3 节），复杂度可降为 $O(n)$. 消去无用符号不增加文法的长度.
- 消去 ε 产生式 复杂度为 $O(2^n)$，结果文法的长度为 $O(2^n)$. 如果用级连方法改造产生式，则复杂度可降为 $O(n)$.
- 消去单元产生式 计算单元偶对和消去单元产生式，复杂度为 $O(n^2)$，结果文法的长度为 $O(n^2)$.
- 用非终结符替换终结符以及打破长度大于 2 的右部 复杂度 $O(n)$，结果文法的长度为 $O(n)$.

有关上下文无关语言的判定性质

- ✧ 有关 CFG 和 PDA 的几个转换问题的复杂度
  - 两种接受方式的 PDA 之间相互转换
    - 终态接受方式转化为空栈接受方式 线性复杂度
    - 空栈接受方式转化为终态接受方式 线性复杂度
  - CFG 与 PDA 之间相互转换
    - CFG转化为空栈接受方式的PDA 线性复杂度
    - 空栈接受方式的 PDA 转化为CFG 指数复杂度，但可以对转移函数做适当的变换，得到 $O(n^3)$ 的复杂度

有关上下文无关语言的判定性质

✧ 判定上下文无关语言是否为空

– 以上下文无关文法表示上下文无关语言

- **判定算法** 可由如下步骤判定上下文无关文法表示的语言是否为空:

  1. 计算所有生成符号的集合:
  2. 判定文法的开始符号是否生成符号；若是，则该文法表示的上下文无关语言非空；否则，该语言为空.

- **算法复杂度** 计算生成符号集合为 $O(n^2)$ 复杂度，但若采用适当的数据结构，复杂度可降为$O(n)$.

有关上下文无关语言的判定性质

✧判定上下文无关语言中是否包含特定的字符串

– 以上下文无关文法表示上下文无关语言

- 判定算法 可由如下步骤判定上下文无关文法表示的语言是否包含某一字符串 *w*：
  1. 将该文法变换为符合 Chomsky 范式:
  2. 采用 CYK 算法判定该文法所产生的语言是否包含字符串 *w*.
- 算法复杂度 CYK 算法由 J.Cocke, D.Younger 和T.Kasami 分别独立提出，基于动态规划（*dynamic programming*）的思想. 设 $|w| = n$ ，则该算法复杂度为 $O(n^3)$.

✧ CYK 算法

– 基本思想 设 $G = (V,T,P,S)$ 为满足 CNF 的 CFG，$w=a_1a_2\ldots a_n \in T^*$; 采用动态规划的思想迭代计算满足下列条件的 X_{ij} $(1 \le i \le j \le n)$:

（1）$X_{ij} \subseteq V$;

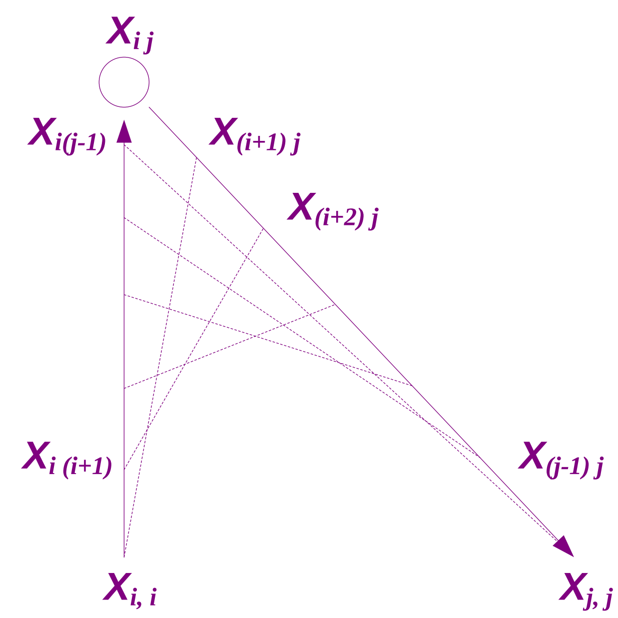
（2）$A \in X_{ij}$ iff $A \overset{*}{\underset{G}{\Rightarrow}} a_i a_{i+1} \ldots a_j$;

这样， $w \in L(G)$ iff $S \in X_{1n}$.

– 迭代计算X_{ij}

（1）$j=i$. 如果 “$A \to a_i$” $\in P$ ， 则$A \in X_{ii}$;

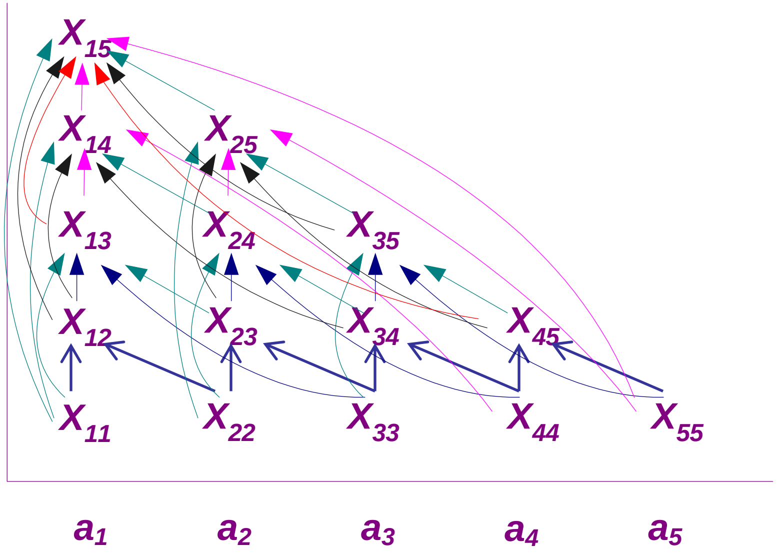
（2）$j>i$. $A \in X_{ij}$ 当且仅当存在 k: $i \le k < j$, 可以找到 $B \in X_{ik}$ 和 $C \in X_{(k+1)j}$,使得 “$A \to BC$” $\in P$. 见右边示意图.



– 复杂度 设 $|w| = n$ ， 则该迭代过程的复杂度为 $O(n^3)$

✧ CYK 算法

– 填表迭代过程 上述计算X_{ij}的迭代过程，可采用填表的方法来实施，如下图所示.



有关上下文无关语言的判定性质

✧ 有关上下文无关语言的几个不可判定问题

1. 给定上下文无关文法是否无二义的?
（定理9.20）
2. 给定上下文无关语言是否固有二义的?
3. 两个上下文无关语言相交是否为空?
4. 两个上下文无关语言是否相等?
5. 给定上下文无关语言是否等于 Σ*? 其中，Σ 为该语言的字母表.

关于上下文无关语言的封闭运算

✧ 关于上下文无关语言的几个主要的封闭运算

- 替换（*substitution*）
- 并（*union*）
- 反向（*reversal*）
- 闭包(星闭包和正闭包)（*closure(*), and closure ($^+$)*）
- 连接（*concatenation*）
- 同态（*homomorphism*）
- 反同态（*inverse homomorphism*）
- 与正规语言的交（*intersection with a regular language*）

关于上下文无关语言的封闭运算

✧ 上下文无关语言的替换

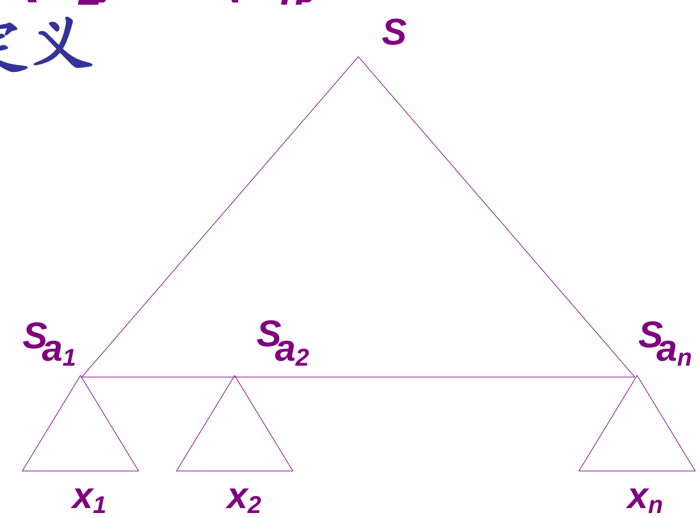
– **记号** 设 Σ 为字母表，L 为语言的集合. 映射 $s: \Sigma \to L$ 称为 Σ 上的一个替换，对 $a \in \Sigma$，$s(a)$ 为某一语言 $L_a \in L$； 替换的概念可以扩充，设 $w = a_1a_2\ldots a_n \in \Sigma^*$，定义

$$s(w) = s(a_1a_2\ldots a_n) = s(a_1)s(a_2)\ldots s(a_n);$$

进一步，设 L 为 Σ 上的语言，定义

$$s(L) = \cup_{w \in L}\, s(w).$$

– **结论** 若 L 为 Σ上的上下文无关语言， s 为 Σ 上的一个替换，并且对任何 $a \in \Sigma$，$s(a)$ 均为上下文无关语言，则 $s(L)$ 也为上下文无关语言.



– **证明思路** 参见右图所示的分析树，$w = a_1a_2\ldots a_n$ 对应的分析树中每个叶结点 a 可替换为语言 $s(a)$ 中任何串的分析树.

✧上下文无关语言的替换

– 举例 设 $\Sigma=\{0,1\}$，替换 s 为

$$s(0) = \{a^n b^n \mid n\geq 1\},\ s(1) = \{aa, bb\}$$

设 $w=01$，则 $s(w) = s(0)s(1) =$

$$\{a^n b^n aa \mid n\geq 1\}\cup\{a^n b^{n+2} \mid n\geq 1\}$$

设 $L= L(0^*)$，则

$$s(L) = (s(0))^* = \{a^n b^n \mid n\geq 1\}^*$$

$$= \{a^{n1} b^{n1} a^{n2} b^{n2} \ldots a^{nk} b^{nk} \mid k\geq 0 \wedge n_i\geq 1\ (1\leq i\leq k)\}.$$

关于上下文无关语言的封闭运算

✧ 上下文无关语言的并

– 结论 若 L 和 M 为CFL，则 $L \cup M$ 也是CFL 。

– 证明 设替换 s 为：$s(0) = L$ ，$s(1) = M$ ，则

$$s(\{0,1\}) = L \cup M.$$

由于$\{0,1\}$ ，L 和 M 皆为CFL，所以$L \cup M$ 为CFL.

关于上下文无关语言的封闭运算

✧ 上下文无关语言的闭包(星闭包和正闭包)

– 结论 若 L 为CFL，则 L^* 和 L^+ 也是CFL .

– 证明 设替换 s 为：$s(1) = L$，则

$$s(\{1\}^*) = L^*, \quad s(\{1\}^+) = L^+.$$

由于L，$\{1\}^*$ 和 $\{1\}^+$ 皆为CFL，所以，L^*和 L^+为CFL.

关于上下文无关语言的封闭运算

✧上下文无关语言的连接

– 结论 若 L 和 M 为CFL，则 LM 也是CFL .

– 证明 设替换 s 为：$s(0) = L$，$s(1) = M$，则

$$s(\{01\}) = LM.$$

由于$\{01\}$，L 和 M 皆为CFL，所以，LM 为CFL.

关于上下文无关语言的封闭运算

✧ 上下文无关语言的同态

– **记号** 设映射 $h: \Sigma \rightarrow T^*$, 则对 $w=a_1a_2\ldots a_n \in \Sigma^*$, 定义

$h(w) = h(a_1)\, h(a_2) \ldots h(a_n)$, 称为串 w 的一个同态;

对语言 $L \subseteq \Sigma^*$, 定义 L 的同态 $h(L) = \{ h(w) \mid w \in L \}$

– **结论** 若 L 为CFL, $h: \Sigma \rightarrow T^*$, 则 $h(L)$ 也是CFL .

– **证明** 设替换 s 为: 对任何$a \in \Sigma$, $s(a) = \{h(a)\}$,

则 $s(L) = h(L)$.

由于$\{h(a)\}$ 和 L 皆为CFL, 所以, $h(L)$ 为CFL.

关于上下文无关语言的封闭运算

✧上下文无关语言的反向

– 记号 设字符串 $w=a_1a_2\ldots a_n$, 则 w 的反向（ reversal ） $w^R = a_na_{n-1}\ldots a_1$; 语言 L 的反向 $L^R = \{ w^R \mid w \in L \}$.

– 结论 若 L 为CFL， 则 L^R 也是CFL：

证明思路 设 $L=L(G)$，其中CFG $G=(V,T,P,S)$.

构造 $G^R = (V,T,P^R,S)$, 其中

$$P^R = \{A \to \alpha^R \mid \text{“}A \to \alpha\text{”} \in P\},$$

可以证明, $L(G^R)= L^R$. 即证，对任何 w,

$$S \overset{*}{\underset{G}{\Rightarrow}} w \quad iff \quad S \overset{*}{\underset{G^R}{\Rightarrow}} w^R$$

(归纳于 G 和 G^R 中推导的长度，留做练习)

□

关于上下文无关语言的封闭运算

✧上下文无关语言的交，补，差

– **结论** 若 L 和 M 为CFL，但 $L \cap M$ 不一定是CFL.

– **举反例** $L = \{0^n 1^n 2^i \mid n,i > 0\}$ 为CFL，它的一个CFG为

$S \to AB$，$A \to 0A1 \mid 01$，$B \to 2B \mid 2$；

$M = \{0^i 1^n 2^n \mid n,i > 0\}$ 为CFL，它的一个CFG为

$S \to AB$，$A \to 0A \mid 0$，$B \to 1B2 \mid 12$；

但 $L \cap M = \{0^n 1^n 2^n \mid n > 0\}$ 不是CFL.

– **推论** 若 L 和 M 为CFL，但 $\overline{L}$ 和 $L-M$ 不一定是CFL.

证明 由于 $L \cap M = \overline{\overline{L} \cup \overline{M}}$，所以，CFL 的补运算不是封闭的.
同样，由于 $\overline{L} = \Sigma^* - L$，所以，CFL 之间的差运算不是封闭的.

关于上下文无关语言的封闭运算

✧ 上下文无关语言与正规语言的交

– 结论 若 L 为CFL，R 为正规语言，则 $L \cap R$ 为CFL .

– 证明思路 设 $R = L(A)$，其中 DFA $A = (Q_A, \Sigma, \delta_A, q_A, F_A)$;
设 $L = L(P)$，其中 PDA $P = (Q_P, \Sigma, \Gamma, \delta_P, q_P, Z_0, F_P)$.
构造 PDA $P' = (Q_P \times Q_A, \Sigma, \Gamma, \delta, (q_P, q_A), Z_0, F_P \times F_A)$,

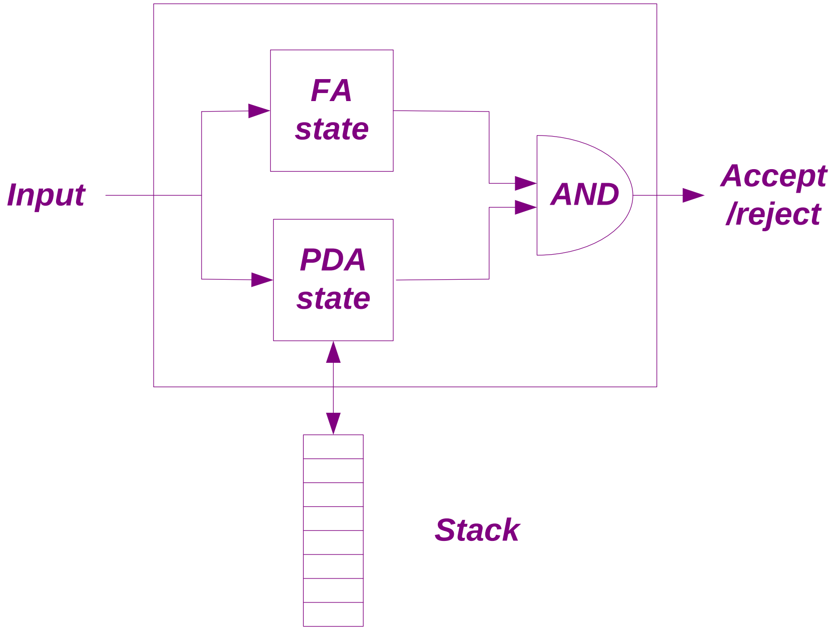
其中 $\delta((q,p), a, X)$ 包含所有满足如下条件的 $((r,s),\gamma)$:

（1）$s = \delta'(p, a)$,
（2）$(r, \gamma) \in \delta(q, a, X)$.

其中 $a \in \Sigma$ 或 $a = \varepsilon$.

可证 $L \cap R = L(P')$.

□



✧ 上下文无关语言的反同态

– **记号** 设映射 $h: \Sigma \to T^*$, 对语言 $L \subseteq T^*$, 定义 L 的反同态

$$h^{-1}(L) = \{ w \mid w \in \Sigma^* \wedge h(w) \in L \}.$$

– **结论** 若 $L \subseteq T^*$ 为CFL， $h: \Sigma \to T^*$, 则 $h^{-1}(L)$ 也是CFL .

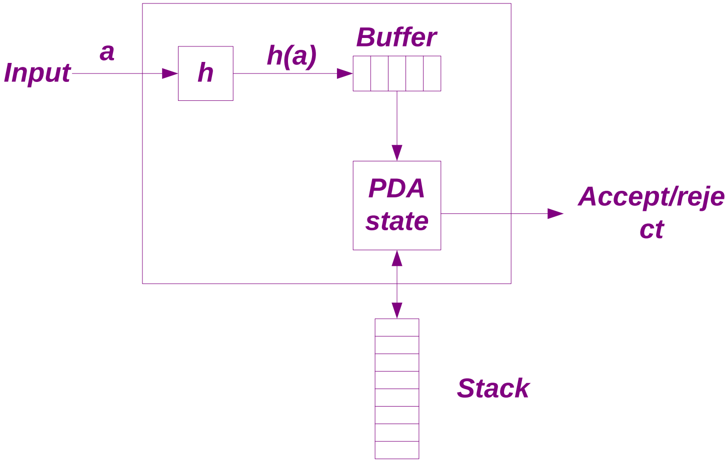
证明思路 设 $L = L(P)$，其中 PDA $P = (Q, T, \Gamma, \delta, q_0, Z_0, F)$.

构造 PDA $P' = (Q \times \{x \mid x$是$h(a)$的后缀$, a \in \Sigma\}, \Sigma, \Gamma, \delta', (q_0, \varepsilon), Z_0, F \times \{\varepsilon\})$.

对$a \in \Sigma$，$\delta'((q,\varepsilon), a, X) = \{((q,h(a)),X)\}$.

若对$b \in T$或$b=\varepsilon$，$(p, \gamma) \in \delta(q, b, X)$，

则有 $((p,x), \gamma) \in \delta'((q,bx), \varepsilon, X)$.



可证 $L(P') = h^{-1}(L)$.

□

课后练习

✧ 必做题:

- Ex.7.2.1(b)
- *!Ex.7.2.1(d)
- *!Ex.7.3.1(b)
- Ex.7.3.2
- Ex.7.3.6
- Ex.7.4.3(c)

✧ 思考题:

- !Ex.7.2.1(f)

That's all for today.

Thank You