第 十 讲

确定下推自动机 CFG化简与规范

确定下推自动机

- 确定下推自动机的概念
- DPDA语言与其它语言的关系
- 终态型DPDA与空栈型DPDA
- DPDA与二义文法

确定下推自动机

- 确定下推自动机的概念
- DPDA语言与其它语言的关系
- 终态型DPDA与空栈型DPDA
- DPDA与二义文法

确定下推自动机的概念

- 定义： PDA $P = (Q, \Sigma, \Gamma, \delta, q_0, z_0, F)$ 称为确定的下推自动机（DPDA），如果满足：
  1. 对于 $a \in \Sigma$ 或 $a = \varepsilon$, $X \in \Gamma - \{\varepsilon\}$, $\delta(q, a, X)$ 最多包含一个元素.
  2. 对于 $a \in \Sigma$, $a \neq \varepsilon$. 若 $\delta(q, a, X) \neq \varnothing$, 则 $\delta(q, \varepsilon, X) = \varnothing$.

$X \neq \varepsilon$

说明：该定义亦称为终态型 DPDA；
类似可定义空栈型 DPDA。

确定下推自动机的概念

容许迁移:

$$q_1 \xrightarrow{a,\ b \to w} q_2$$

或者

$$q_1 \xrightarrow{\varepsilon,\ b \to w} q_2$$

（确定的）

确定下推自动机的概念

容许迁移:

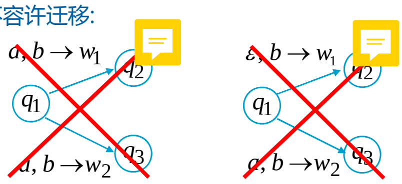
$$q_1 \xrightarrow{a,\ b \to w_1} q_2 \qquad q_1 \xrightarrow{a,\ c \to w_2} q_3$$

$$q_1 \xrightarrow{\varepsilon,\ b \to w_1} q_2 \qquad q_1 \xrightarrow{\varepsilon,\ c \to w_2} q_3$$

（确定的）

确定下推自动机的概念

不容许迁移:

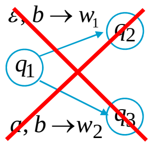


(非确定的)

确定下推自动机的概念

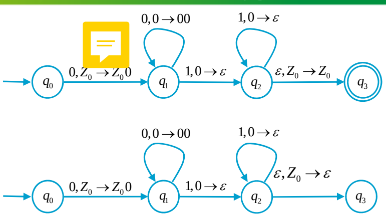
不容许迁移:

δ(q_1,a,b) 和 δ(q_1,ε,b) 不能同时出现。
即可以存在单独的迁移 δ(q_1,a,b) 和 δ(q_1,ε,b)，但是它们互相排斥，不能同时存在。



(非确定的)

DPDA 例



$L = \{0^n 1^n \mid n \geq 1\}$

DPDA 语言

定义:(终态型DPDA语言)

如果语言L 能够被一DPDA

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

接收，则 L 称为确定性下推自动机的语言，或确定性上下文无关语言，或终态型DPDA语言。

语言 $L = \{0^n 1^n \mid n \geq 1\}$ 是一个终态型DPDA语言。

DPDA 语言

定义:(空栈型DPDA语言)

如果语言L 能够被一空栈型DPDA

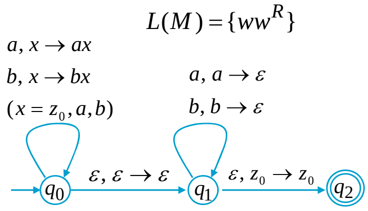
$P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0)$

接收，则 L 称为空栈型DPDA的语言。

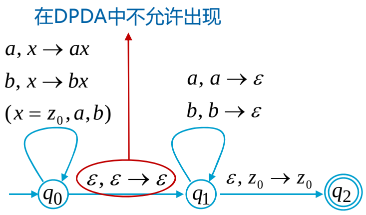
语言 $L = \{0^n 1^n \mid n \geq 1\}$
也是一个空栈型DPDA的语言。

非确定下推自动机

$$L(M) = \{ww^R\}$$



非确定下推自动机



确定下推自动机

- 确定下推自动机的概念
- DPDA语言与其它语言的关系
- 终态型DPDA与空栈型DPDA
- DPDA与二义文法

DPDA与PDA

因为 DPDA 也是NPDA，则：

$$\left\{ \begin{array}{c} \text{DPDA} \\ \text{的语言} \end{array} \right\} \subseteq \left\{ \begin{array}{c} \text{CFL} \\ \text{(NPDA)} \end{array} \right\}$$

DPDA与PDA

因为存在CFL语言：

$$L_{ww^R} = \{ww^R \mid w \in \{0,1\}^*\}$$

没有 DPDA 接受该语言。

DPDA与PDA

因此该包含是真包含：

$$L \notin \left\{ \begin{array}{c} \text{DPDA} \\ \text{的语言} \end{array} \right\} \subset L \in \left\{ \begin{array}{c} \text{CFL} \\ \text{(NPDA)} \end{array} \right\}$$

说明 NPDA 比 DPDA 的表达能力更强

DPDA语言与正则语言

定理：若 L 是正则语言，则存在 DPDA P，使得 $L(P) = L$

证明方法：

设 DFA $A = (Q, \Sigma, \delta_A, q_0, F)$，且 $L(A)=L$

构造一 DPDA

$$P = (Q, \Sigma, \{Z_0\}, \delta_P, q_0, Z_0, F)$$

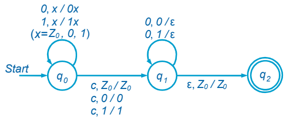
其中 $\delta_P(q, a, Z_0) = \{(p, Z_0)\}$ *iff* $\delta_A(q, a) = p$

则有 $L(P) = L(A)$

DPDA语言与正则语言

DPDA 的表达能力强于有限自动机.

例：语言 L = $\{wcw^R|w$ 为 $0,1$ 字符串$\}$
不是正则语言，但它是如下 DPDA 的语言：



确定下推自动机

- 确定下推自动机的概念
- DPDA语言与其它语言的关系
- 终态型DPDA与空栈型DPDA
- DPDA与二义文法

终态型DPDA与空栈型DPDA

- 前缀性质

一个语言 L 具有前缀性质，当且仅当不存在 $x, y \in L$, $x \neq y$, 且 x 为 y 的前缀

例：语言

$L=L(a^*)$

没有前缀性质

终态型DPDA与空栈型DPDA

- 定理

语言 L 是空栈型 DPDA P 的语言, 当且仅当：
a) L 具有前缀性质;
b) L 是某终态型 DPDA P' 语言, 即 $L=L(P')$

(证明：留作练习)

真包含

定理说明：
{空栈型DPDA的语言} ⊂ {终态型DPDA的语言}

空栈型DPDA语言与正则语言

例：

(1) 语言 $L = \{wcw^R|w$ 为 0,1 字符串$\}$
具有前缀性质，并且 L 是 DPDA 的语言,
所以 L 是某个空栈接受的 DPDA 的语言.

例 (1) 说明空栈型 DPDA的语言不一定是正则语言

(2) 语言$\{0\}^*$不具有前缀性质，所以不存在空栈型 DPDA P, 使得 $N(P)=\{0\}^*$

例 (2) 说明正则语言不一定是空栈型DPDA的语言

因此，空栈型DPDA的语言与正则语言没有关系

确定下推自动机

- 确定下推自动机的概念
- DPDA语言与其它语言的关系
- 终态型DPDA与空栈型DPDA
- DPDA与二义文法

DPDA与二义文法

- 定理:
  若语言 L 是空栈型 DPDA P 的语言, 则存在无二义上下文无关文法G，使得$L=L(G)$。

证明思路

前节中，从空栈型 PDA 构造等价 CFG 的方法，构造与 DPDA 等价的 CFG.

可证对于任何所接受的串w, 此 CFG 有唯一的最左推导，因而是无二义文法.

DPDA与二义文法

- 定理:
  若语言 L 是终态型 DPDA P 的语言, 则存在无二义上下文无关文法G，使得$L=L(G)$。

证明思路

令 \$ 不出现在 L 的任何串中，记 $L' = \{w\$ \mid w \in L\}$,
则 L' 具有前缀性质. 即存在 DPDA P', 使得 $L' = N(P')$，
从而存在一个无二义文法 G', 使得 $L' = L(G')$
从G'构造CFG G, \$ 作为变量, 并增加产生式 $\$ \to \varepsilon$.
显然有 $L = L(G)$.
易证，G是一个无二义文法.

DPDA与二义文法

例： 语言

$$L = \{ww^R \mid w \text{ 为 } 0,1 \text{ 字符串}\}$$

(1) 不是任何 DPDA 的语言.

(2) 非固有二义的. 即存在非二义文法:

$$S \to 0S0 \mid 1S1 \mid \varepsilon.$$

存在非固有二义的CFG语言 L, 不是任何 DPDA 的语言.

{DPDA 语言} $\subset$ {CFG非固有二义语言}

真包含

CFG化简与规范

- 消去无用符号
- 消去ε产生式
- 消去单一产生式
- CFG的化简与Chomsky范式

CFG化简与规范

- 消去无用符号
- 消去ε产生式
- 消去单一产生式
- CFG的化简与Chomsky范式

有用符号

- 有用符号 (*useful symbol*)
  - 对于 CFG G

$$G = (V, T, S, P)$$

称符号 $X \in V \cup T$ 是有用的，当且仅当

$$S \overset{*}{\Rightarrow} \alpha X \beta \overset{*}{\Rightarrow} w,$$

其中 $w \in T^*$，$\alpha, \beta \in (V \cup T)^*$.

- 无用符号 (*useless symbol*)
  - 非有用符号

无用符号与无用产生式

无用产生式：含有无用符号的产生式

例：

$S \to aSb$

$S \to \varepsilon$

$S \to A$

$A \to aA$

无用产生式

产生式推导不能终止：

$S \Rightarrow A \Rightarrow aA \Rightarrow aaA \Rightarrow \ldots \Rightarrow aa\ldots aA \Rightarrow \ldots$

无用符号与无用产生式

另例：

$S \to A$

$A \to aA$

$A \to \varepsilon$

$B \to bA$

无用符号与无用产生式

开始变量 S 不可达

产生符与可达符

- 产生符号

  X 称为产生符，如果存在 $w \in T^*$，满足 $X \stackrel{*}{\Rightarrow} w$

  约定：终结符是产生符

- 可达符号

  X 称为符号可达符，如果存在 $\alpha, \beta \in (V \cup T)^*$，满足

  $S \stackrel{*}{\Rightarrow} \alpha X \beta$

  约定：S是可达符

产生符与可达符

- 与有用符号的关系
  - 有用符号一定是产生符号和可达符号
  - 反之，成立吗？

  不一定。

产生符算法

算法步骤：

对 CFG $G = (V, T, S, P)$，产生符集合由下列步骤计算：

基础：任何终结符 $a \in T$ 都是产生符；

归纳：若有产生式 $A \to \alpha$，且 $\alpha \in (V \cup T)^*$ 中的每个符号都是产生符，则 A 也是产生符；

- 定理：此算法恰好求得 G 的所有产生符。

证明思路：一方面，所得符号的确是产生符；

另一方面，任何产生符都可由上述步骤得到。

可达符算法

算法步骤：

对 CFG $G = (V, T, S, P)$，可达符集合由下列步骤计算：

基础：S 是可达符；

归纳：若 A 是可达符，且$A \to \alpha$，$\alpha \in (V \cup T)^*$，则 α 中的符号都是可达符；

- 定理：此算法恰好求得 G 的所有可达符。

证明思路：一方面，所得符号的确是可达符；

另一方面，任何可达符都可由上述步骤得到。

消去无用符号

设 CFG $G = (V, T, S, P)$, $L(G) \neq \varnothing$

- 消去非产生符号

  从G 中删除所有非产生符以及所有包含这些符号的产生式，得到 CFG

  $G_2 = (V_2, T_2, S, P_2)$

- 消去不可达符号

  从G_2 中删除所有不可达符以及所有包含这些符号的产生式，得到 CFG

  $G_1 = (V_1, T_1, S, P_1)$

注意消去的顺序

消去无用符号

定理：有上述算法得到的文法 G_1不包含无用符号，且 $L(G_1)= L(G)$.

定理说明：

- 剩余的符号都是有用符号;
- 新的文法与原来的文法是等价的;

证明思路:

一方面，G_1中不包含无用符号；
另一方面，对任何 w， $S \overset{*}{\underset{G_1}{\Rightarrow}} w$ *iff* $S \overset{*}{\underset{G}{\Rightarrow}} w$

消去无用符号步骤

✓ 计算产生符号集合

✓ 消去非产生符号

✓ 计算可达符号集合

✓ 消去不可达符号

消去无用符号

例：文法 **CFG G = ({S,A,B,C}, {a,b}, S, P),**

$S \to aS \mid A \mid C$

$A \to a$

$B \to aa$

$C \to aCb$

消去无用符号

例：第一步: 消去非产生符

$S \to aS \mid A \mid C$

$A \to a$

$B \to aa$

$C \to aCb$

第一轮: $\{a, b, A, B\}$

$S \to A$

第二轮: $\{a, b, A, B, S\}$

消去无用符号

例：只保留产生符{A,B,S}，除去非产生符

$S \to aS \mid A \mid \cancel{C}$

$A \to a$

$B \to aa$

$\cancel{C \to aCb}$

删除无用产生式

$S \to aS \mid A$

$A \to a$

$B \to aa$

消去无用符号

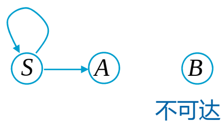
例：第二步: 消去非可达符

$S \to aS \mid A$

$A \to a$

$B \to aa$

使用依赖关系图



消去无用符号

例：只保留可达符{A,S}， 除去非可达符

$S \to aS \mid A$

$A \to a$

$\cancel{B \to aa}$

化简后的文法

$S \to aS \mid A$

$A \to a$

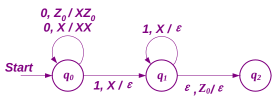
删除无用产生式

消去无用符号

例 对下图的 *PDA*, 构造*CFG* $G = (V,\{0,1\}, S, P)$,

其中 $V = \{S\} \cup \{ [pYq] \mid p,q \in \{q_0,q_1,q_2\}, Y \in \{Z_0,X\} \}$

产生式 *P* 定义如下：



(1) $S \to [q_0Z_0q_0]$;

$S \to [q_0Z_0q_1]$;

$S \to [q_0Z_0q_2]$;

(2) $[q_0Z_0q_j] \to 0[q_0Xq_i]\,[q_iZ_0q_j]$, $i, j = 0,1,2$; $((q_0,XZ_0) \in \delta(q_0, 0, Z_0))$

(3) $[q_0Xq_j] \to 0[q_0Xq_i]\,[q_iXq_j]$, $i , j = 0,1,2$; $((q_0,XX) \in \delta(q_0, 0, X))$

(4) $[q_0Xq_1] \to 1$; $((q_1,\varepsilon) \in \delta(q_0, 1, X))$

(5) $[q_1Xq_1] \to 1$; $((q_1,\varepsilon) \in \delta(q_1, 1, X))$

(6) $[q_1Z_0q_2] \to \varepsilon$; $((q_2,\varepsilon) \in \delta(q_1, \varepsilon, Z_0))$

消去无用符号

$S \to [q_0Z_0q_0]$； $S \to [q_0Z_0q_1]$； $S \to [q_0Z_0q_2]$；

$[q_0Z_0q_j] \to 0[q_0Xq_i]\,[q_iZ_0q_j]$, $i, j = 0,1,2$;

$[q_0Xq_j] \to 0[q_0Xq_i]\,[q_iXq_j]$， $i , j = 0,1,2$;

$[q_0Xq_1] \to 1$; $[q_1Xq_1] \to 1$; $[q_1Z_0q_2] \to \varepsilon$；

上述文法的产生符包括：

终结符号： 0, 1

非终结符号：$[q_0Xq_1]$, $[q_1Xq_1]$, $[q_1Z_0q_2]$, $[q_0Z_0q_2]$

消去无用符号

消去所有非产生符号，得到新文法：

$S \to [q_0Z_0q_2]$;

$[q_0Z_0q_2] \to 0[q_0Xq_1]\,[q_1Z_0q_2]$; $[q_0Xq_1] \to 0[q_0Xq_1]\,[q_1Xq_1]$;

$[q_0Xq_1] \to 1$; $[q_1Xq_1] \to 1$; $[q_1Z_0q_2] \to \varepsilon$;

为使文法更简洁，记：

$[q_0Z_0q_2]$ 为A, $[q_0Xq_1]$ 为B, $[q_1Xq_1]$ 为C, $[q_1Z_0q_2]$ 为D。

上述文法的产生式改写如下：

$S \to A$; $A \to 0BD$; $B \to 0BC$;

$B \to 1$; $C \to 1$; $D \to \varepsilon$.

消去无用符号

以下产生式表示的文法中，

S、A、B、C、D、0、1

都为可达符号，所以消除不可达符号后，结果没有变化：

$S \to A$; $A \to 0BD$; $B \to 0BC$;

$B \to 1$; $C \to 1$; $D \to \varepsilon$;

至此，该文法已经消去了所有无用符号

$S \to A$; $A \to 0BD$; $B \to 0BC$;

$B \to 1$; $C \to 1$; $D \to \varepsilon$.

CFG化简与规范

- 消去无用符号
- 消去ε产生式
- 消去单一产生式
- CFG的化简与Chomsky范式

消去 ε 产生式

目的：方便文法的设计, 利于文法规范化。

消去 ε 产生式, 除文法不能产生串 ε 外，不会影响到原文法相应的语言中其它字符串的产生。

- 可空符号 (*nullable symbol*)

  设 CFG $G = (V, T, S, P)$, 称符号 $A \in V$ 是可空的，如果 $A \stackrel{*}{\Rightarrow} \varepsilon$.

  消去 ε 产生式及其影响，需要计算可空符号的集合。

可空符号集计算

- 计算步骤： 对于 CFG $G = (V, T, S, P)$，可空符号集合通过下列归纳步骤计算：

  基础: 对所有产生式 $A \to \varepsilon$, A 是一个可空符号；

  归纳: 如果有产生式 $B \to C_1C_2\ldots C_k$, 其中每一个 $C_i \in V$ 是可空符号，则 B 也是可空符号；

- 结论: 此算法恰好得到G 的所有可空符号：

  证明思路: 一方面，得到的符号的确是可空符号；
  另一方面，任何可空符号都可由上述步骤得到。

消去ε 产生式算法

设 *CFG* $G = (V, T, S, P)$, 通过下列步骤可以消去 G 中 ε 产生式及其影响：

(1) 计算 G 的可空符号集合；

(2) 对每一产生式 $A \to A_1A_2\ldots A_k$

- 在 G_1 中对应有一组产生式，每一个可空符号都可能出现或不出现；
- 若包含 $m < k$ 个可空符号，则对应G_1 中2^m 个产生式；
- 若包含 k 个可空符号，则对应G_1中 2^k-1 个产生式；

(3) G_1 中不包含 G 的所有 ε 产生式：$A \to \varepsilon$.

消去ε 产生式

定理：通过上述算法 从CFG G 构造CFG G_1，则文法G_1不含ε 产生式，且

$L(G_1)= L(G)$ - {ε}.

证明思路:

证对任何 w, $S \stackrel{*}{\underset{G_1}{\Rightarrow}} w$ iff ($S \stackrel{*}{\underset{G}{\Rightarrow}} w$, $w \neq \varepsilon$).

消去ε 产生式

例：

$S \to aMb$

$M \to aMb$

$\cancel{M \to \varepsilon}$

可空符号

$M \to \varepsilon$

化简后文法

$S \to aMb \mid ab$

$M \to aMb \mid ab$

消去ε 产生式

例：在上节PDA转CFG例中，得到以下产生式表示的文法，D 为可空符号：

$S \to A$; $A \to 0BD$; $B \to 0BC$;

$B \to 1$; $C \to 1$; $D \to \varepsilon$.

消去 ε 产生式，得到如下产生式集合：

$S \to A$; $A \to 0BD$; $A \to 0B$;

$B \to 0BC$; $B \to 1$; $C \to 1$.

CFG化简与规范

- 消去无用符号
- 消去ε产生式
- 消去单一产生式
- CFG的化简与Chomsky范式

单一产生式

- 单一产生式 (*unit productions*)

  形如 $A \to B$ 的产生式，其中A、B 为非终结符。

- 消去单一产生式的目的

  可减少文法的变量，简化文法推导, 利于文法规范化。

单一偶对

- 单一偶对 (*unit pairs*)

  设 CFG $G = (V, T, P, S)$, $A, B \in V$, (A, B) 称为单一偶对，如果 $A \overset{*}{\Rightarrow} B$，且该推导过程仅使用单一产生式。

  消去单一产生式时，需要计算所有单一偶对的集合。

单一偶对集合的计算

- 计算步骤： 设 CFG $G = (V, T, S, P)$,单一偶对的集合通过下列归纳步骤计算：

  基础： 对于任何 $A \in V$, (A, A) 是一个单一偶对；

  归纳： 如果 (A, B) 是一个单一偶对，且 $B \to C$ 是产生式 $(C \in V)$, 则 (A, C) 是一个单一偶对.

- 结论:上述步骤恰好求得G 的所有单一偶对.

  证明思路：一方面，所得到的偶对的确是单一偶对；

  另一方面，任何单一偶对都可由上述步骤得到.

消去单一产生式算法

设 CFG $G = (V, T, S, P)$, 通过下列步骤消去 G 中的单一产生式：

(1) 计算 G 的单一偶对集合；
(2) 对每个单一偶对 (A, B), 在 G_1 中加入产生式$A \to \alpha$, 其中 $B \to \alpha$ 为一非单一产生式；
(3) G_1 中包含 G 的所有非单一产生式。

由此得到CFG: $G_1 = (V, T, S, P_1)$

- 定理: 上述步骤从G 构造G_1, 有$L(G_1)= L(G)$

  证明思路： 欲证对任何 w, $S \overset{*}{\underset{G_1}{\Rightarrow}} w$ *iff* $S \overset{*}{\underset{G}{\Rightarrow}} w$

消去单一产生式

例：以下产生式表示的文法中：

$S \to A$; $A \to 0BD$; $A \to 0B$; $B \to 0BC$;
$B \to 1$; $C \to 1$

单一偶对：$(S, S), (A, A), (B, B), (C, C), (D, D), (S, A)$

通过上述步骤消去单一产生式，得到产生式集合：
$S \to 0BD$; $S \to 0B$; $A \to 0BD$; $A \to 0B$;
$B \to 0BC$; $B \to 1$; $C \to 1$

虽然上述化简步骤能消去一些冗余符号，但有可能产生新的无用符号，如本例中变量 A 和 D

CFG化简与规范

- 消去无用符号
- 消去ε产生式
- 消去单一产生式
- CFG的化简与Chomsky范式

CFG 的简化

- 设 CFG $G = (V, T, S, P)$, 通过下列步骤对 G 简化：
  第一步: 消去 ε-产生式;
  第二步: 消去单一产生式;
  第三步: 消去无用产生式。

注意 以上简化步骤的次序

- 定理: 设 CFG G 的语言包含非 ε 字符串，通过上述步骤从CFG G 构造CFG G_1，则有
  $L(G_1)= L(G) - \{\varepsilon\}$

Chomsky 范式

- Chomsky 范式 (CNF)
  上下文无关文法CFG $G = (V, T, S, P)$ 称为 *Chomsky* 范式，如果:
  (1) G 中不含无用符号;
  (2) 产生式 P 只具有如下两种简单形式之一：
  $A \to BC$ 或 $A \to a$
  其中 A, B, C 是变量，*a* 是终结符。

Chomsky 范式

例：

Chomsky 范式	非 Chomsky 范式
$S \to AS$	$S \to AS$
$S \to a$	$S \to AAS$
$A \to SA$	$A \to SA$
$A \to b$	$A \to aa$

Chomsky 范式

- 转化Chomsky范式算法 步骤 1

任何不含 ε 的非空 CFL, 都存在一个 CFG G.

由下列步骤对 G 进行简化：
(1) 消除 G 中的 ε 产生式；
(2) 消除 G 中的单一产生式；
(3) 消除 G 中的无用符号.

得到不含 ε 产生式、单一产生式和无用符号的CFG G_2.
由前所述，$L(G_2)= L(G)$.

Chomsky 范式

- 转化Chomsky范式算法 步骤 2

将 G_2 做如下变换：

a) 如果某一终结符 a 出现于某些右部长度大于 1 的产生式中，则引入一个新的非终结符，如A，将这些产生式中的 a 替换为 A, 并增加新的产生式：

$$A \to a$$

则右部长度大于 1 的产生式中只含有非终结符；

Chomsky 范式

- 转化Chomsky范式算法 步骤 2

将 G_2 做如下变换：

b) 右部长度 >2 的产生式 $A \to B_1B_2\dots B_k$, k>2，采用级连 (*cascade*) 方法转变为只含两个非终结符；引入k-2个新的非终结符$C_1, C_2, \dots, C_{k-2}$,

$$A \to B_1C_1,\ C_1 \to B_2C_2,\ \dots,\ C_{k-3} \to B_{k-2}C_{k-2},\ C_{k-2} \to B_{k-1}B_k$$

得到$CFG\ G_1$

Chomsky 范式

例: 以下产生式表示的文法中，已经不存在 ε 产生式和单一产生式：

$S \to 0BD$; $S \to 0B$; $A \to 0BD$; $A \to 0B$;
$B \to 0BC$; $B \to 1$; $C \to 1$

消去无用符号 D，A 后，得到如下产生式集合：

$S \to 0B$; $B \to 0BC$; $B \to 1$; $C \to 1$

引入新的非终结符 A，用 A 替换 0，并增加新的产生式 $A \to 0$ ，得到如下产生式集合：

$S \to AB$; $B \to ABC$; $A \to 0$; $B \to 1$; $C \to 1$

Chomsky 范式

进一步引入新的非终结符 D, 将其变换为：

$S \to AB$; $B \to AD$; $D \to BC$; $A \to 0$; $B \to 1$; $C \to 1$

定理：设 CFG G 的语言包含非 ε 的字符串，通过上述步骤从 G 构造 G_1，则 G_1 符合 CNF 的要求，且满足：

$$L(G_1) = L(G) - \{\varepsilon\}$$

$S \to AB$; $B \to ABC$; $A \to 0$; $B \to 1$; $C \to 1$

课后练习

✧ 必做题:
- P-251 *Ex.6.4.2* (*c*) (其中$m \neq 0$任意)
- P-251 *Ex.6.4.3* *(*a*),!(*b*)
- P-271 *Ex.7.1.3*
- P-272 *Ex.7.1.9*(*b*)

✧ 思考题:
- P-251 *Ex.6.4.3* *!(*c*)
- P-272 *!*Ex.7.1.10*

Thank you