复习

- **2型文法推导，归约，语法树，二义性**

  对于前缀表达式文法G1:

  E::= – EE

  E::= – E

  E::= a | b | c

  画出文法的句子 – – a – bc 的推导、归约过程，和所有可能语法树，判断该文法是否具有二义性。

§ 4.2 上下文无关文法的变换

- 上下文无关文法(Context-Free Grammar, CFG) 的简化
  - 消无用符号
  - 消 ε 产生式
  - 消单产生式
- 对生成式形式进行标准化

生成式的标准形式

- Chomsky范式 (CNF - Chomsky Normal Form)

生成式形式为A→BC，A→a，A, B, C∈N，a∈T (后面将证明, 每个上下文无关文法都有一个CNF文法)

- Greibach范式 (GNF)

生成式形式为A→aβ，a∈T，β∈N*

意义：对每个2型语言都可找到一个文法使产生式的右端都以终结符开始

中心思想:消除左递归.

变换算法 -- 消去无用符号

- 有用符号（useful symbol）

  对于 CFG $G=(N, T, P, S)$，称符号 $X \in N \cup T$ 是有用的，当且仅当 $S \stackrel{*}{\Rightarrow} \alpha X\beta \stackrel{*}{\Rightarrow} w$，其中 $w \in T^*$，$\alpha, \beta \in (N \cup T)^*$.

  – 称符号 X 是生成符号（generating symbol），当且仅当存在 $w \in T^*$，满足 $X \stackrel{*}{\Rightarrow} w$.

  – 称符号 X 是可达符号（reachable symbol），当且仅当存在 $\alpha, \beta \in (N \cup T)^*$，满足 $S \stackrel{*}{\Rightarrow} \alpha X\beta$.

- 无用符号（useless symbol）

  – 非生成符号

  – 不可达符号

消去无用符号

- 计算生成符号（generating symbol）集
- 计算可达符号（reachable symbol）集
- 消去非生成符号、不可达符号
- 消去相关产生式

计算生成符号集

✧ **思路** 对于 CFG $G=(N, T, P, S)$，可通过下列归纳步骤计算生成符号集合：

基础 任何终结符 $a \in T$ 都是生成符号；

归纳 如果有产生式 $A \rightarrow \alpha$，其中 $\alpha \in (N \cup T)^*$ 的每一个符号都是生成符号，则 A 也是生成符号；

算法1: 找出有用非终结符

✧步骤:

(1) $N_0 = \Phi$ (赋为Φ) N_0为有用的非终结符集

(2) $N' = \{A \mid A \rightarrow \omega$且$\omega \in T^*\}$ N'为非终结符集合

(3) 如果$N_0 \neq N'$ 则转(4),否则转(6)

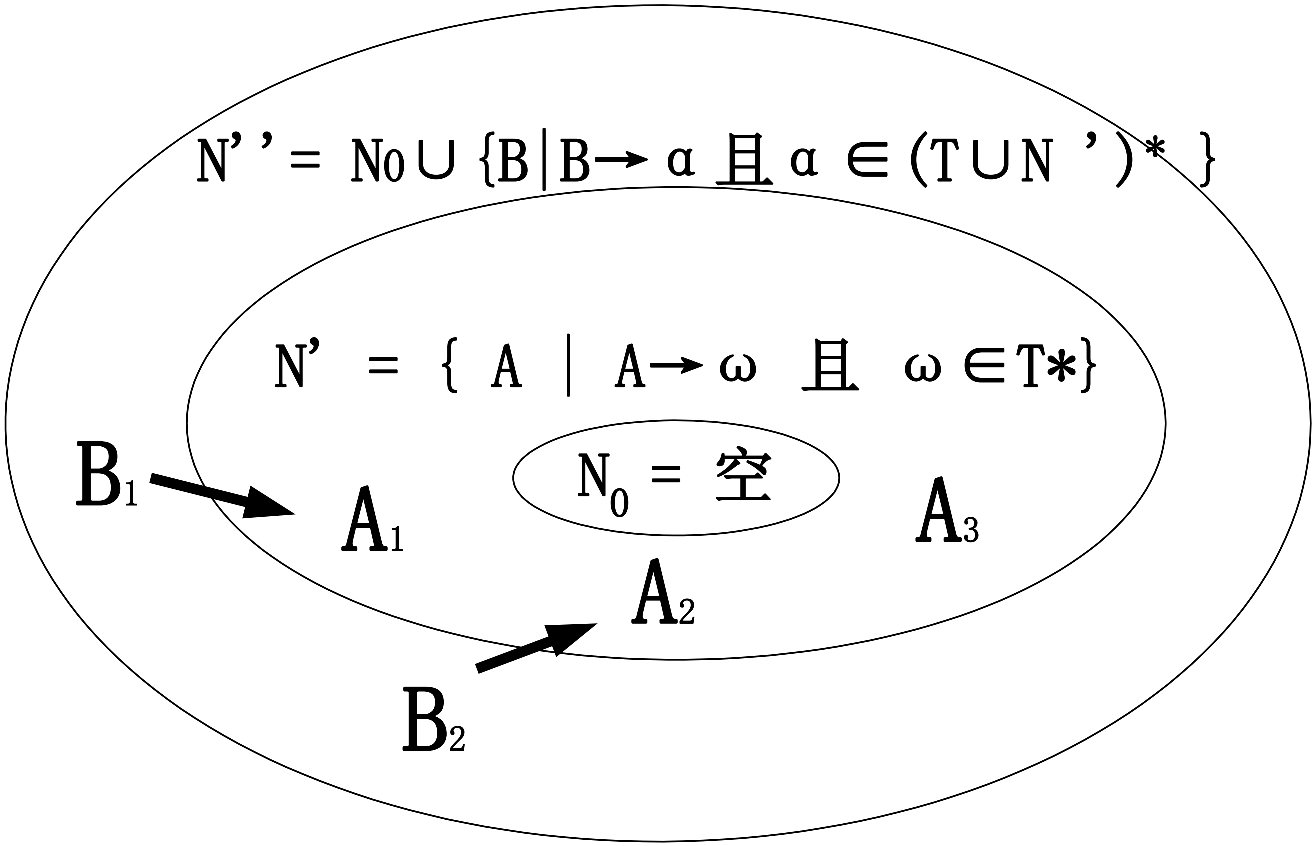
(4) $N_0 = N'$

(5) $N' = N_0 \cup \{ A \mid A \rightarrow \alpha$且$\alpha \in (T \cup N_0)^* \}$, 转(3)

(6) $N_1 = N'$

小结: 算法1找出能推出终结符串的非终结符作为有用符号.

算法1: 找出有用非终结符（图示）



一层层向外扩展，直至最外两层相等为止。所得集合即是算法1的有用符号。

计算可达符号集

✧ **思路** 对于 CFG $G=(N, T, P, S)$，可通过下列归纳步骤计算可达符号集合：

基础 S 是可达符号；

归纳 如果 A 是可达符号，并且有产生式 $A \rightarrow \alpha$，其中 $\alpha \in (N \cup T)^*$，则 α 中的每一个符号都是可达符号；

算法2: 找出有用符号(从S可达的符号)

✧算法步骤:

(1) $N_0 = \{S\}$

(2) $N' = \{x \mid A \in N_0$ 且 $A \rightarrow \alpha x \beta\} \cup N_0$

(N'为有用符号集合)

(3) 如果 $N_0 \neq N'$ 转(4), 否则转(5)

(4) $N_0 = N'$; 转(2)   (继续迭代下去)

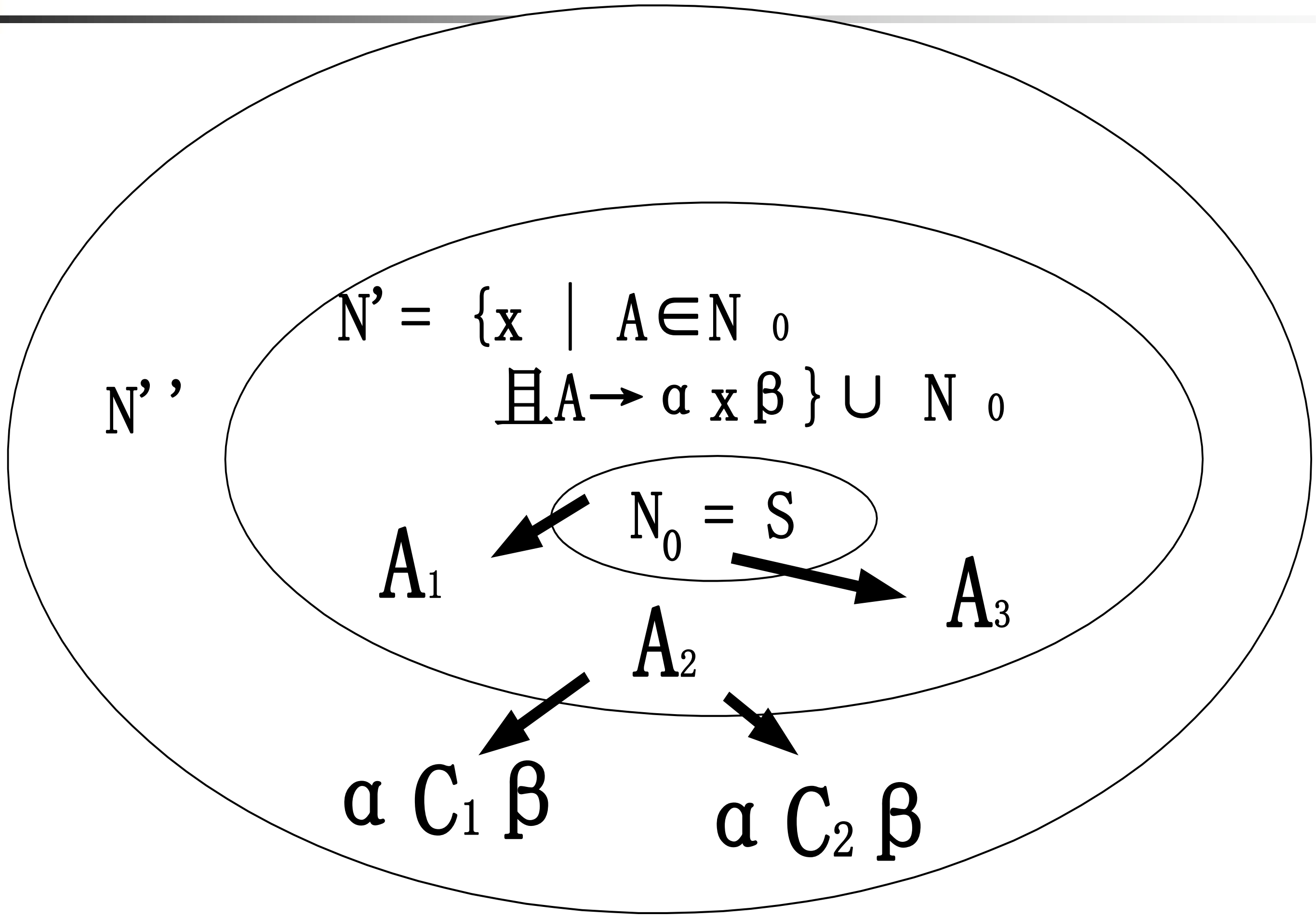
(5) $N_1 = N' \cap N$

$T_1 = N' \cap T$

P_1 由P内只含N'中符号的生成式组成

(即删去了从S起不可达的符号).

算法2: 找出从S可达的符号 (图示)



一层层外扩,找出从S可达的所有符号(含非终结符和终结符)

消去非生成符号及不可达符号

例：(书P102 例1)

已知2型文法G=({S, A, B}, {a}, P, S)

P: $S\to AB$, $S\to a$, $A\to a$

由算法1: B推不出终结符串, 删除之, 并删$S\to AB$.

N_1={S, A}, P_1: $S\to a$, $A\to a$

由算法2: A不出现在S能推导出的句型中, 删除之.

并删$A\to a$

∴ 结果为G1=({S}, {a}, { $S\to a$}, S)

应用算法1和算法2,可删去文法中所有无用符号.

消去非生成符号及不可达符号

注意:

必须先执行算法1,再执行算法2,不能颠倒.

否则,可能导致无用符号不会被完全删除.

例:

上例中,若先用算法2,得

$S \to a$, $S \to AB$, $A \to a$

再用算法1, 得$S \to a$, $A \to a$。而$A \to a$是多余的.

定理:

任何非空的上下文无关语言,可由不存在无用符号的上下文无关语言产生(证明略)。

课堂作业

去除下面生成式的无用符号

G_1：

$S \rightarrow DC \mid ED$

$C \rightarrow CE \mid DC$

$D \rightarrow a$

$E \rightarrow aC \mid b$

消去ε产生式

✧ 目的 方便文法的设计，利于文法规范化.

✧ 影响 消去ε产生式，除了文法不能产生字符串ε外，不会影响到原文法相应的语言中其它字符串的产生.

✧ 可致空符号（nullable symbol）

对于 CFG $G=(N, T, P, S)$，称符号 $A \in N$ 是可致空的，当且仅当 $A \overset{*}{\Rightarrow} \varepsilon$.

消去ε产生式及其影响，需要计算可致空符号的集合.

算法3: 生成无ε文法

✧ **定义:**

若G的生成式中无任何ε产生式,或只有一个生成式 $S\rightarrow\varepsilon$ 且S不出现在任何生成式的右边,则称G为无ε文法.

✧ **思路** 对于 CFG G = (N, T, P, S), 可通过下列归纳步骤计算可致空符号集合:

基础 对于所有产生式 $A\rightarrow\varepsilon$，A 是一个可致空符号

归纳 如果有产生式 $B\rightarrow C_1C_2\ldots C_k$，其中每一个 $C_i \in N$ 是可致空符号，则 B 是一个可致空符号;

算法3：生成无ε文法

算法步骤：

(1) 利用算法1, 找出$N' = \{A \mid A \in N$且$A \Rightarrow^+ \varepsilon\}$

(找出能推导出ε的所有非终结符A)

(2) 用以下两步组成P_1

① 如果生成式$A \rightarrow \beta_0 C_1 \beta_1 C_2 \ldots C_n \beta_n \in P \quad n \geq 0$

且每个C_k $(1 \leq k \leq n)$ 均在N'内

而对于$0 \leq j \leq n$, 没有β_j在N'内 (β_i也可能是终结符)

则P_1应加入 $A \rightarrow \beta_0 Y_1 \beta_1 Y_2 \beta_2 \ldots Y_n \beta_n$

其中Y_k是C_k或ε (即所有的可能组合)

但是$A \rightarrow \varepsilon$不加入$P_1$

算法3：生成无ε文法

算法步骤（续）：

② 如果$S \in N'$,则P_1中应加入以下生成式

$S_1 \to \varepsilon | S$　　S_1是新的起始符

$N_1 = N \cup \{S_1\}$

如果$S \notin N'$, 则$N_1 = N$, $S_1 = S$

由此得出$G_1 = (N_1, T, P_1, S_1)$

消去ε产生式

例：(书P103 例2)

$G = (N, T, P, S)$ 其中$N=\{S\}, T=\{ a, b \}$

$P: S \rightarrow aSbS \mid bSaS \mid \varepsilon$

求其无ε文法 $G_1=(N_1, T, P_1, S_1)$

解： (1) $\because S \Rightarrow^+ \varepsilon \quad \therefore N'=\{S\}$

(2) ① P_1的构造

由$S \rightarrow aSbS$; 加入

$S \rightarrow aSbS \mid a\varepsilon bS \mid aSb\varepsilon \mid a\varepsilon b\varepsilon$

由$S \rightarrow bSaS$; 加入$S \rightarrow bSaS \mid b\varepsilon aS \mid bSa\varepsilon \mid b\varepsilon a\varepsilon$

但$S \rightarrow \varepsilon$不加入$P_1$

② 由$S \in N'$得出

$S_1 \rightarrow \varepsilon \mid S \qquad N_1 = N \cup \{S_1\} = \{S, S_1\}$

消去ε产生式

- 练习

以下产生式表示的文法中，B, D 为可致空符号：

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

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

通过上述步骤，消去ε产生式，得到如下产生式集合：

$S \rightarrow A$; $A \rightarrow 0BD$; $A \rightarrow 0B$; $A \rightarrow 0D$; $A \rightarrow 0$; $B \rightarrow 0BC$;

B->0C; $B \rightarrow 1$; $C \rightarrow 1$.

消去单产生式

✧ 单产生式 形如 $A \rightarrow B$ 的产生式，其中A、B 为非终结符.

✧ 消去单产生式的目的 可简化某些证明，减少推导步数，利于文法规范化.

✧ 单元偶对（unit pairs）

对于 CFG $G = (N, T, P, S)$，$A, B \in N$，称 (A, B) 是单元偶对，当且仅当 $A \stackrel{*}{\Rightarrow} B$，且该推导过程仅使用过单产生式.

消去单产生式时，需要计算所有单元偶对的集合.

消去单产生式

思路 设 CFG $G=(N, T, P, S)$，对每个单元偶对 (A, B)，在 G_1 中加入产生式 $A\to\alpha$，其中 $B\to\alpha$ 为一个非单产生式；从而消去 G 中的单产生式，得到 CFG $G_1=(N, T, P_1, S)$：

算法步骤:

(1) 对每个 $A\in N$,构造非终结符集 $N_A=\{B|A\Rightarrow^* B\}$

(N_A是可由A推出的单生成式中的非终结符集)。

构造方法分三步:

① $N_0=\{A\}$

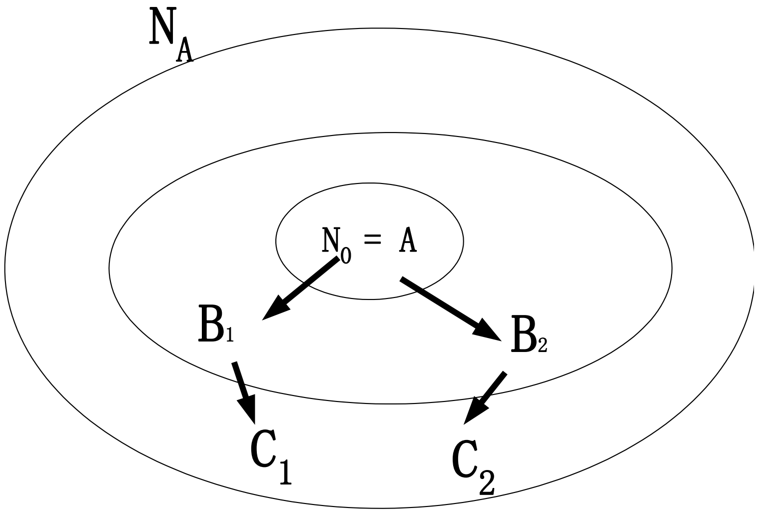
② $N'=\{C \mid$ 如果 $B\to C\in P$ 且 $B\in N_0\}\cup N_0$

③若 $N'\neq N_0$,则 $N_0=N'$,转向② (继续迭代)

否则 $N_A=N'$, 转向(2). (已得到了完整的 N_A)

消去单产生式

算法步骤（续）：



(2) 构造P_1:

如果$B\rightarrow\alpha\in P$且不是单生成式，则对于$B\in N_A$的所有A,把$A\rightarrow\alpha$加入到P_1中

(即对每个$B \in N_A$ (意味着$A \Rightarrow^+ B$)的非单生成式$B\rightarrow\alpha\in P$,直接将α与N_A的A连接, 构成新产生式$A\rightarrow\alpha$加入到P_1中)

(3) 得到$G_1=(N_1, T_1, P_1, S)$

CFG 的简化

✧ **小结**

设 CFG $G = (V, T, P, S)$，可以通过下列步骤对 G 进行简化：

（1）消除 G 中的 ε 产生式；

（2）消除 G 中的单元产生式；

（3）消除 G 中的无用符号.

✧ **注意** 以上简化步骤的次序.

✧ **结论** 设 CFG G 的语言至少包含一个非 ε 的字符串，通过上述步骤从 G 构造 G_1，则有 $L(G_1)= L(G) - \{\varepsilon\}$.

消去单产生式（例）

例: 设2型文法G = ({S,A,B}, { (,) , + , * , a}, P, S)

P: $S \to S+A|A$　$A \to A*B|B$　$B \to (S)|a$

构造无单生成式的文法G_1

解: (1) 构造N_S，N_A，N_B

① $N_0=\{S\}$

② $N' = \{C \mid B \to C \in P 且 B \in N_0\} \cup N_0$

$= \{A\} \cup \{S\} = \{A, S\}$

③ $\because N_0 \neq N'$　$\therefore N_0 = N' = \{A,S\}$

继续转②　$N'=\{B\} \cup \{A,S\}=\{B,A,S\}$

$\because N_0 \neq N'$　$\therefore N_0 = N' =\{B,A,S\}$

继续转②　有$N'=\{B,A,S\}= N_0$

$\therefore N_S= \{B, A, S\}$

同理可得: $N_A=\{A,B\}$, $N_B=\{B\}$

消去单产生式（续）

(2) 构造P_1

对N_S= {S,A,B}

$\because$ S→S+A∈P且不是单生成式, 且S ∈ N_S

于是，将S→S+A加到P_1中.

又$\because$ A→A*B ∈ P, A ∈ N_S

$\therefore$ 将S→A*B加到P_1中.

($\because$S→A, A→A*B $\therefore$ 直接用S→A*B代替这两条产生式)

又$\because$ B→(S) ∈ P, B ∈ N_S $\therefore$ 将S→（S）加到P_1中.

又$\because$ B→a ∈ P, B ∈ N_S $\therefore$ 将S→a加到P_1中.

消去单产生式（续）

同理: 对N_A= {A, B}

∵ $A \rightarrow A*B \in P$且非单生成式, $A \in N_A$

∴ 将$A \rightarrow A*B$加到P_1中.

∵ $B \rightarrow (S)|a \in P$且非单生成式, $B \in N_A$

∴ 将$A \rightarrow (S)|a$加到P_1中.

同理: 对N_B={B}

将$B \rightarrow (S)|a$加到P_1中.

结果: P_1为

$S \rightarrow S+A|A*B|(S)|a$　　$A \rightarrow A*B|(S)|a$　　$B \rightarrow (S)|a$

消除递归

递归的定义：

在2型文法中

若存在 $A \Rightarrow^{+} \alpha A \beta$, $A \in N$，则称G是递归文法。

若存在 $A \Rightarrow^{+} A\beta$　G是左递归文法

若存在 $A \Rightarrow^{+} \alpha A$　G是右递归文法

若存在 $A \Rightarrow^{+} A$　G是循环文法

生成式的代换

定义: 2型文法中所有形如$A\to\alpha$的生成式称为A生成式.

引理1: 对$A\to\alpha$的A生成式可进行变换

设 $G = (N,T,P,S)$

$A\to\alpha B\beta$是P中的生成式, $B\in N,\alpha,\beta\in(N\cup T)^*$

$B\to r_1|\ r_2|\ldots|\ r_k$是P中的B生成式

可生成$G_1 = (N_1,T, P_1,S)$

P_1中的生成式是将P中的$A\to\alpha B\beta$用$A\to\alpha r_1\beta|\ldots|\alpha r_k\beta$取代.

证明思路: P_1和P中生成式产生的语言相等,证明从略。

生成式的代换

例：设文法G有$S \to \underbrace{a}_{\alpha}\ \underbrace{S}_{B}\ \underbrace{S}_{\beta} \mid b$

应用引理1, 设 α =**a** **B=S,** β =S

$B \to \underbrace{a\ S\ S}_{r_1} \mid \underbrace{b}_{r_2}$

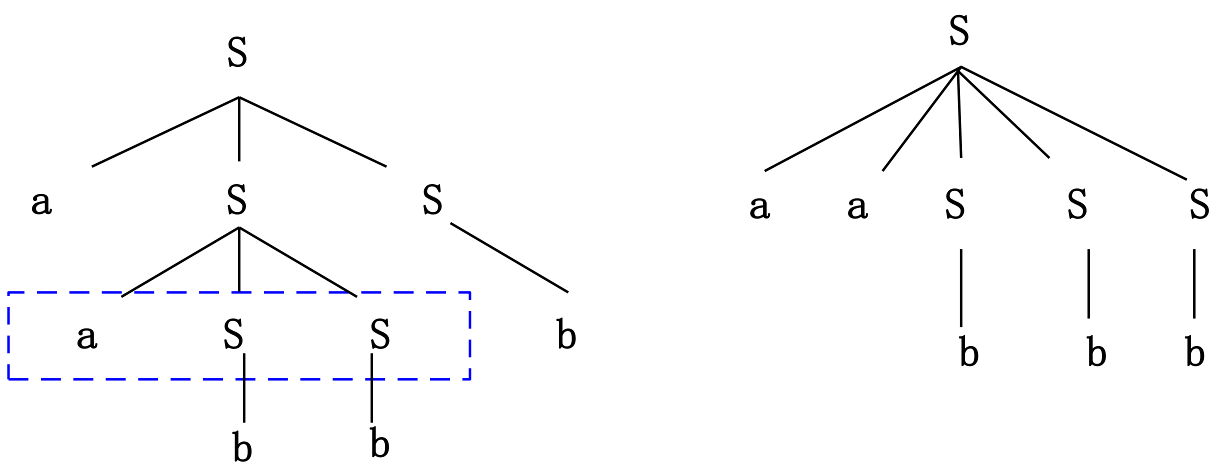
即

替换$S \to aSS \mid b$有$\mathbf{G_1}$的产生式为:

$S \to a\ \underbrace{aSS}_{r_1}\ S \mid a\underbrace{b}_{r_2}S \mid b$

生成式的代换

其句子aabbb的推导树为



即将子树根S用下一层的直接后代代替了.

消除直接左递归

引理2: 消除直接左递归

设$G = (N, T, P, S)$, P中有A生成式

$$A \rightarrow A\alpha_1 \mid A\alpha_2 \mid \ldots \mid A\alpha_m \mid \beta_1 \mid \beta_2 \mid \ldots \mid \beta_n$$

其中β_i的第一个字符不是非终结符A (可以是其它非终结符)

可构成$G_1 = (N \cup \{A'\}, T, P_1, S)$, A'为新引入的非终结符

G_1中的P_1为，将P中的A生成式用以下生成式取代

$$A \rightarrow \beta_1 \mid \beta_2 \mid \ldots \mid \beta_n \mid \beta_1 A' \mid \beta_2 A' \mid \ldots \mid \beta_n A'$$

$$A' \rightarrow \alpha_1 \mid \alpha_2 \mid \ldots \mid \alpha_m \mid \alpha_1 A' \mid \alpha_2 A' \mid \ldots \mid \alpha_m A'$$

证明思路:

G和G_1二者的正则式都是$(\beta_1+\beta_2+\ldots+\beta_n)(\alpha_1+\ldots+\alpha_m)^*$

消除直接左递归（例）

例: (书P106 例4)

$$S\to S\underbrace{+A}_{\alpha_1}\mid\underbrace{A}_{\beta_1},\quad A\to A\underbrace{*B}_{\alpha_1}\mid\underbrace{B}_{\beta_1},\quad B\to(S)\mid a$$

可变换为

$S\to A \mid AS'$

$S'\to +A \mid +A\,S'$

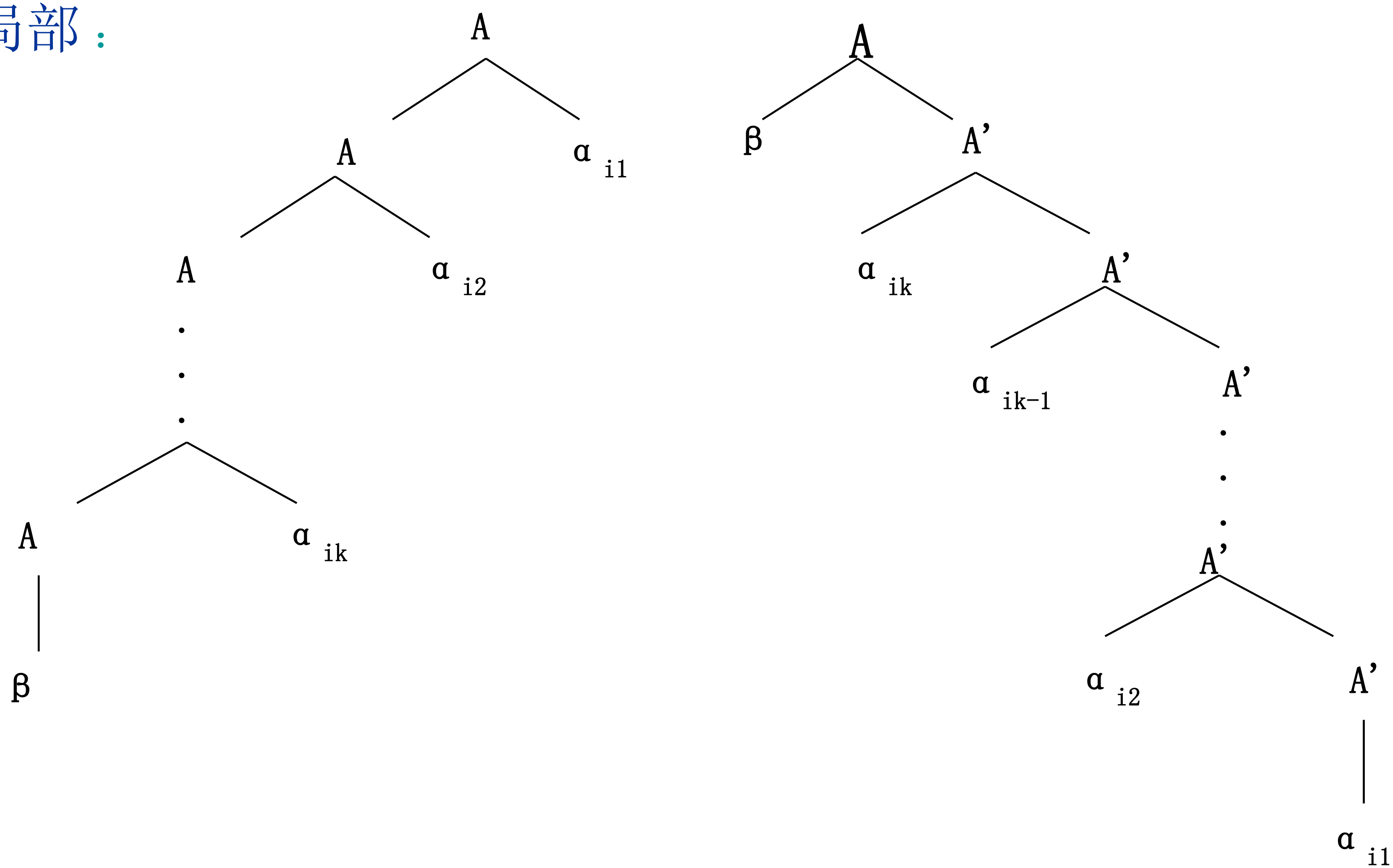
$A\to B \mid BA'$

$A'\to *B \mid *B\,A'$

$B\to (S) \mid a$

消除直接左递归对推导树的影响

G中局部：



消除左递归的算法

Why 消左递归？

✧ 以后的句法分析算法不适用于左递归, 会引起死循环。

✧ 对于给定的2型文法, 该文法不存在无用符号, 无循环且是无 ε 生成式的文法, 为了消除G中可能存在的左递归, 构成一个等效的无左递归的文法G_1, 可用算法5。

✧ *算法5在原理上与求解正规表达式方程组的算法类似.*

算法5: 消除全部左递归

排列非终结符 $N = \{ A_1, A_2, \ldots, A_m \}$

$i := 1$

将 $A_i \rightarrow A_i \alpha_1 | \ldots | A_i \alpha_n | \beta_1 | \ldots | \beta_p$ 变换为

$A_i \rightarrow \beta_1 | \ldots | \beta_p | \beta_1 A_i' | \beta_p A_i'$ $A_i' \rightarrow \alpha_1 | \ldots | \alpha_n | \alpha_1 A_i' | \ldots | \alpha_n A_i'$

i=m? —Y→ 结束

i：=i+1，j：=1

对每个 $A_i \rightarrow A_j \alpha$，$A_j \rightarrow \beta_1 | \ldots | \beta_n$ 用 $A_i \rightarrow \beta_1 \alpha | \beta_2 \alpha | \ldots | \beta_n \alpha$ 代替

j=i-1? —N→ j：=j+1

Y

消除左递归(示例)

$A_1 \to A_2A_3|a$ （1）

$A_2 \to A_3A_1|\ A_1b$ （2）

$A_3 \to A_1A_2\ |\ A_3A_3\ |\ a$ （3）

排序：{ A_1， A_2， A_3}

当 i=1 对（1）变换，不用变。 $A_1 \to A_2A_3|a$

当 i=2 对（2）变换 $A_2 \to A_3A_1\ |\ A_2\ A_3b\ |\ ab$ （4）

β_1 α_1 β_2

消直接左递归 $\begin{cases} A_2 \to A_3A_1|ab|\ A_3A_1\ A_2'|abA_2' & (5) \\ A_2' \to A_3\ b\ |\ A_3b\ A_2' & (6) \end{cases}$

当i=3，j=1 $A_3 \to A_1A_2\ |\ A_3A_3\ |\ a$

$\to A_2\ A_3\ A_2\ |\ a\ A_2\ |A_3A_3\ |\ a$ （7）

消除左递归(示例)

$$j=2 \quad A_3 \rightarrow A_3\overbrace{A_1\, A_3A_2}^{\alpha_1} \mid \overbrace{a\, b\, A_3\, A_2}^{\beta_1} \mid A_3\, \overbrace{A_1\, A_2'A_3\, A_2}^{\alpha_2}$$

$$\mid \underbrace{a\, b\, A_2'A_3\, A_2}_{\beta_2} \mid \underbrace{a\, A_2}_{\beta_3} \mid A_3\underbrace{A_3}_{\alpha_3} \mid \underbrace{a}_{\beta_4} \qquad (8)$$

对（8）消直接左递归

$$\begin{cases} A_3 \rightarrow \beta_1\, A_3' \mid \beta_2 A_3' \mid \beta_3 A_3' \mid \beta_4\, A_3' \mid \beta_1 \mid \beta_2 \mid \beta_3 \mid \beta_4 \\ A_3' \rightarrow \alpha_1 \mid \alpha_2 \mid \alpha_3 \mid \alpha_1\, A_3' \mid \alpha_2\, A_3' \mid \alpha_3\, A_3' \mid \end{cases}$$

作业

Ch4 习题： 8. 9.