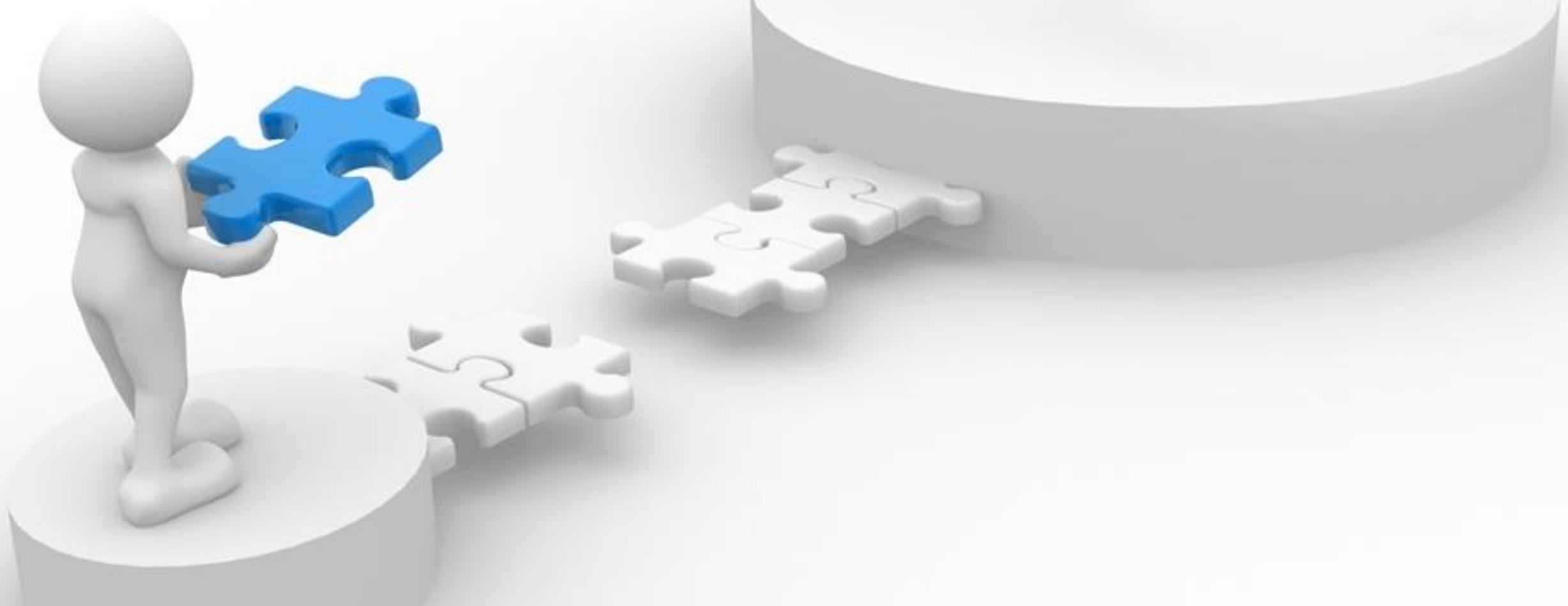


# 编译原理



# 目录

CONTENTS

1 词法分析

---

★ 2 语法分析

---

3 语义分析

---

4 中间代码生成

---

5 代码优化

---

6 目标代码生成

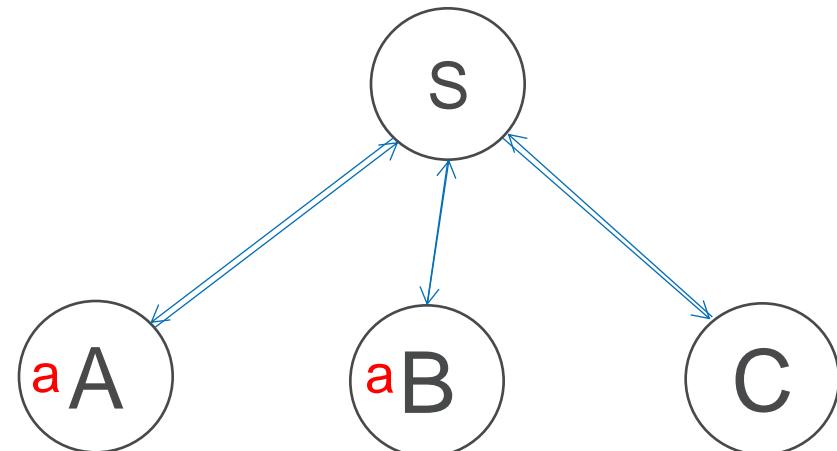
---



# 基本思想

## ✧ 语法分析

- 核心问题：识别 (*recognition*) 与解析 (*parsing*)
- 两种实现途径
  - 自顶向下 (*top-down*) 分析
  - 自底向上 (*bottom-up*) 分析



## 第4章 自顶向下语法分析方法

### ■ 例4.1

设文法G1[S]:

$$S \rightarrow pA \mid qB$$

$$A \rightarrow cAd \mid a$$

$$B \rightarrow dB \mid b$$

考虑对输入串w=pccadd的分析。

- 这个文法有以下两个特点：
  - 每个产生式的右部都由终结符号开始。
  - 如果两个产生式有相同的左部，那么它们的右部由不同的终结符开始。



## ■ 例4.2

设文法G2[S]: 考虑对输入串w=ccap的分析。

$S \rightarrow Ap$

$S \rightarrow Bq$

$A \rightarrow a$

$A \rightarrow cA$

$B \rightarrow b$

$B \rightarrow db$

- 
- 这个文法有以下特点：
    - 产生式的右部不全是由终结符开始
    - 如果两个产生式有相同的左部，它们的右部是由不同的终结符或非终结符开始。
    - 文法中无空产生式

## ■ 例4.3

设文法G[S]:

$S \rightarrow aA$

$S \rightarrow d$

$A \rightarrow bAS$

$A \rightarrow \epsilon$

考虑对输入串 $w= abd$ 的分析。



## ■ 结论

- 当某一非终结符的产生式中含有空产生式时，它的非空产生式右部的首符号集两两不相交，并与在推导过程中紧跟该非终结符右边可能出现的终结符集也不相交则仍可构造确定的自顶向下分析。



# 自顶向下预测分析

◆ 确定的自顶向下分析

◆ LL(1) 分析

- 非终结符选择和产生式选择都是确定的  
无回溯的方法

从左向右扫描，可能向前查看 (lookahead)  
确定数目的单词

分析成功的结果：得到唯一的最左推导

分析条件：对文法需要有一定的限制



# 自顶向下预测分析

## ✧ 举例

– 单词序列  $a^n b^m$  ( $n \geq 0, m > 0$ ) 的预测分析过程

文法 G (S) :

(1)  $S \rightarrow AB$

(2)  $A \rightarrow aA$

(3)  $A \rightarrow \epsilon$

(4)  $B \rightarrow b$

(5)  $B \rightarrow bB$

S (1)

$\Rightarrow AB$  (2)

文法 G (S) :  $S \rightarrow abA \mid abB$  (2)

.....  
 $\Rightarrow a^nAB$  (3)  
 $\quad \quad \quad B \rightarrow b$

$\Rightarrow a^nB$  (5)

$\Rightarrow a^nB$  (5)

文法 G (S) :  $S \rightarrow aAb \mid aAc$

$\Rightarrow a^nA^{n-1}B$  (4)  
 $\quad \quad \quad A \rightarrow a$

$\Rightarrow a^nB$  (成功)

只要向前查看 2 个单词，就可预测分析 L(G) 中所有句子



# LL (1) 分析

从左 (Left)  
向右扫描单  
词

- 向前查看  
一个单词

- 产生的是最左  
(Leftmost)  
推导



- ✧ 对文法的限制
    - 要求文法是LL (1) 的
- 思考题：什么是LL (1) 文法？

例4.4  
文法G1[S]：  
 $S \rightarrow aSb$   
 $S \rightarrow aS$   
 $S \rightarrow \epsilon$

- 定义4.1 (开始符号集或首符号集)  
设 $G=(V_T, V_N, P, S)$ 是上下文无关文法

$\text{FIRST}(\alpha) = \{a | \alpha \xrightarrow{*} a\beta, a \in V_T, \alpha, \beta \in V^*\}$

若 $\alpha \xrightarrow{*} \epsilon$ , 则规定 $\epsilon \in \text{FIRST}(\alpha)$ .

- 设文法 $G_2[S]$ :
  - $S \rightarrow Ap$
  - $S \rightarrow Bq$
  - $A \rightarrow a$
  - $A \rightarrow cA$
  - $B \rightarrow b$
  - $B \rightarrow db$



◦ 定义4.2 (后跟符号集)

设 $G=(V_T, V_N, P, S)$ 是上下文无关文法, 对 $A \in V_N$ 定义

$\text{FOLLOW}(A) = \{a | S \xrightarrow{*} \mu A \beta \text{ 且 } a \in V_T, a \in \text{FIRST}(\beta), \mu \in V^*, \beta \in V^+ \}$

若 $S \xrightarrow{*} uA\beta$ , 且 $\beta \xrightarrow{*} \epsilon$ , 则规定 $\# \in \text{FOLLOW}(A)$

■ 设文法 $G[S]$ :

$S \rightarrow aA \mid d$

$A \rightarrow bAS \mid \epsilon$



◦ 定义4.3

给定上下文无关文法的产生式 $A \rightarrow \alpha, A \in V_N, \alpha \in V^*$ ,

若 $\alpha \stackrel{*}{\not\Rightarrow} \varepsilon$ , 则 $\text{SELECT}(A \rightarrow \alpha) = \text{FIRST}(\alpha)$

若如果 $\alpha \stackrel{*}{\Rightarrow} \varepsilon$ , 则 $\text{SELECT}(A \rightarrow \alpha) = (\text{FIRST}(\alpha) - \{\varepsilon\}) \cup \text{FOLLOW}(A)$

例如对文法 $G3[S]$ :

$$S \rightarrow aA | d$$

$$A \rightarrow bAS | \varepsilon$$

$$\text{SELECT}(S \rightarrow aA) = \{a\}$$

$$\text{SELECT}(S \rightarrow d) = \{d\}$$

$$\text{SELECT}(A \rightarrow bAS) = \{b\}$$

$$\text{SELECT}(A \rightarrow \varepsilon) = \{a, d, \#\}$$



- 定义4.4

一个上下文无关文法是LL(1)文法的充分必要条件是，对每个非终结符A的两个不同产生式， $A \rightarrow \alpha$ ,  $A \rightarrow \beta$ , 满足

$$\text{SELECT}(A \rightarrow \alpha) \cap \text{SELECT}(A \rightarrow \beta) = \emptyset$$

其中 $\alpha$ 、 $\beta$ 不能同时 $\xrightarrow{*} \varepsilon$ 。



## 4.2 LL(1)文法的判别

- 求出能推出 $\epsilon$ 的非终结符
  - 计算FIRST集
  - 计算FOLLOW集
  - 计算SELECT集
  - 判别是否是LL(1)文法



◦ 例：设文法G[S] 为：

$S \rightarrow AB$

$S \rightarrow bC$

$A \rightarrow \epsilon$

$A \rightarrow b$

$B \rightarrow \epsilon$

$B \rightarrow aD$

$C \rightarrow AD$

$C \rightarrow b$

$D \rightarrow aS$

$D \rightarrow c$

判断它是否是LL(1)文法。



# 1.求出能推出 $\epsilon$ 的非终结符

~~S ,~~

~~S → bC~~

~~A → c~~

~~A → b~~

~~B → C~~

~~B → aD~~

~~C , D~~

~~C → b~~

~~D → aS~~

~~D → c~~

非终结符	S	A	B	C	D
初值	未定	未定	未定	未定	未定
第一次扫描		:			
第二次扫描				:	



## 2.计算FIRST集

- 根据定义计算

1.若 $X \in V_T$ ,则 $\text{FIRST}(X)=\{X\}$

2.若 $X \in V_N$ ,且有产生式 $X \rightarrow a\dots$ ,  $a \in V_T$ ,则 $a \in \text{FIRST}(X)$ ;若 $X \rightarrow \varepsilon$ 也是一条产生式,则 $\varepsilon \in \text{FIRST}(X)$ .

3.若 $X \rightarrow Y\dots$ 是一个产生式且 $Y \in V_N$ ,则把 $\text{FIRST}(Y)$ 中的所有非 $\varepsilon$ 元素都加到 $\text{FIRST}(X)$ 中;若 $X \rightarrow Y_1 Y_2 \dots Y_K$ 是一个产生式, $Y_1, Y_2, \dots, Y_{i-1}$ 都是非终结符,而且对于任何 $j$ , $1 \leq j \leq i-1$ , $\text{FIRST}(Y_j)$ 都含有 $\varepsilon$ (即

\*

$Y_1 \dots Y_{i-1} \Rightarrow \varepsilon$ ),则把 $\text{FIRST}(Y_i)$ 中的所有非 $\varepsilon$ 元素都加到 $\text{FIRST}(X)$ 中;特别是,若所有的 $\text{FIRST}(Y_j)$ , $j=1, 2, \dots, K$ 均含有 $\varepsilon$ ,则把 $\varepsilon$ 加到 $\text{FIRST}(X)$ 中.



# 计算FIRST集示例

文法 $G[S]$ 为：

$$S \rightarrow AB | bC$$

$$A \rightarrow b | \epsilon$$

$$B \rightarrow aD | \epsilon$$

$$C \rightarrow AD | b$$

$$D \rightarrow aS | c$$

非终结符	S	A	B	C	D
					$FIRST(AB) = \{a, b, \epsilon\}$
初值 $ST(S)$	未定 $\{a, b, \epsilon\}$	未定 $\{b\}$	未定 $\{b\}$	未定 $\{b\}$	未定 $\{b\}$
第一次扫描		$FIRST(A) = \{b, \epsilon\}$		是 $FIRST(b)$	$\{b\}$
第一次扫描		$FIRST(B) = \{a, \epsilon\}$		$FIRST(\epsilon)$	$\{\epsilon\}$
第一次扫描		$FIRST(C) = \{a, b, c\}$			$FIRST(aD) = \{a\}$
					$FIRST(D) = \{a, c\}$
					$FIRST(AD) = \{a, b, c\}$
					$FIRST(aS) = \{a\}$
					$FIRST(c) = \{c\}$



### 3.计算FOLLOW集

- 根据定义计算

- 1.对于文法的开始符号S,置#于FOLLOW(S) 中;
- 2.若  $A \rightarrow \alpha B \beta$  是一个产生式,则把  $\text{FIRST}(\beta)-\{\varepsilon\}$  加至 FOLLOW(B)中;
- 3.若  $A \rightarrow \alpha B$  是一个产生式,或  $A \rightarrow \alpha B \beta$  是一个产

生式而  $\beta \xrightarrow{*} \varepsilon$ , 则把 FOLLOW(A)加至 FOLLOW(B) 中.



# 计算FOLLOW集示例

文法 $G[S]$ 为：

$$S \rightarrow AB \mid bC$$

$$A \rightarrow b \mid \epsilon$$

$$B \rightarrow aD \mid \epsilon$$

$$C \rightarrow AD \mid b$$

$$D \rightarrow aS \mid c$$

非终结符	S	A	B	C	D
初值	$\text{FOLLOW}(S) = \{\#\}$				
第一次扫描	$\text{FOLLOW}(A) = \{a, C, \#\}$	未定	未定	未定	未定
第二次扫描	$\text{FOLLOW}(B) = \{\#\}$	是	是		否
	$\text{FOLLOW}(C) = \{\#\}$			否	
	$\text{FOLLOW}(D) = \{\#\}$				



## 4.计算SELECT集

文法 $G[S]$ 为：

$$S \rightarrow AB \mid bC$$

$$A \rightarrow b \mid \epsilon$$

$$B \rightarrow aD \mid \epsilon$$

$$C \rightarrow AD \mid b$$

$$D \rightarrow aS \mid c$$

非终结符	S	A	B	C	D
初值	未定	未定	未定	未定	未定
第一次扫描		是	是		否
第二次扫描	是			否	



## 4.计算SELECT集

非终结符名	是否 $\xrightarrow{*} \epsilon$	FIRST 集	FOLLOW 集
S	是	{b,a, $\epsilon$ }	{#}
A	是	{b, $\epsilon$ }	{a,c,#}
B	是	{a, $\epsilon$ }	{#}
C	否	{a,b,c}	{#}
D	否	{a,c}	{#}

$\text{SELECT}(S \rightarrow AB) = \{a, b, \#\}$

$\text{SELECT}(B \rightarrow aD) = \{a\}$

$\text{SELECT}(S \rightarrow bC) = \{b\}$

$\text{SELECT}(C \rightarrow AD) = \{a, b, c\}$

$\text{SELECT}(A \rightarrow \epsilon) = \{a, c, \#\}$

$\text{SELECT}(C \rightarrow b) = \{b\}$

$\text{SELECT}(A \rightarrow b) = \{b\}$

$\text{SELECT}(D \rightarrow aS) = \{a\}$

$\text{SELECT}(B \rightarrow \epsilon) = \{\#\}$

$\text{SELECT}(D \rightarrow c) = \{c\}$



## 5. 判别是否是LL(1)文法

$\text{SELECT}(S \rightarrow AB) \cap \text{SELECT}(S \rightarrow bC) = \{b, a, \# \} \cap \{b\} = \{b\} \neq \emptyset$

$\text{SELECT}(A \rightarrow \epsilon) \cap \text{SELECT}(A \rightarrow b) = \{a, \epsilon, \# \} \cap \{b\} = \emptyset$

$\text{SELECT}_{\tau}(B \rightarrow \epsilon) \cap \text{SELECT}(B \rightarrow aD) = \{\# \} \cap \{a\} = \emptyset$

$\text{SELECT}_{\tau}(C \rightarrow AD) \cap \text{SELECT}(C \rightarrow b) = \{b, a, c\} \cap \{b\} = \{b\} \neq \emptyset$

$\text{SELECT}(D \rightarrow aS) \cap \text{SELECT}(D \rightarrow c) = \{a\} \cap \{c\} = \emptyset$

所以文法G[S] 不是LL(1)文法。



# LL (1) 分析

✧ 课堂练习：验证如下文法 G (S) 不是 LL (1) 文法

文法 G (S) :

- (1)  $S \rightarrow AB$
- (2)  $A \rightarrow Da | \varepsilon$
- (3)  $B \rightarrow cC$
- (4)  $C \rightarrow aADC | \varepsilon$
- (5)  $D \rightarrow b | \varepsilon$



## 4.3 某些非LL(1)文法到LL(1)文法的等价变换

- 提取左公共因子
- 消除左递归



1. 提取左公共因子

$A \rightarrow \alpha\beta_1|\alpha\beta_2| \dots |\alpha\beta_n$  变换为

$A \rightarrow \alpha(\beta_1|\beta_2| \dots |\beta_n)$  , 即

$A \rightarrow \alpha A'$

$A' \rightarrow \beta_1|\beta_2| \dots |\beta_n$

◦ 例4.6

文法G1[S] :

$S \rightarrow aSb$

$S \rightarrow aS$

$S \rightarrow \epsilon$

化为:

$S \rightarrow aS(b | \epsilon)$

$S \rightarrow \epsilon$

进一步化为:

$S \rightarrow aSA$

$A \rightarrow b$

$A \rightarrow \epsilon$

$S \rightarrow \epsilon$

结果仍然不是LL(1)文法。因此，文法中不含左公共因子只是LL(1)文法的必要条件。



例4.7

文法G2[A]:

$A \rightarrow ad$

$A \rightarrow Bc$

$B \rightarrow aA$

$B \rightarrow bB$

1.化为:

$A \rightarrow ad$

$A \rightarrow aAc$

$A \rightarrow bBc$

$B \rightarrow aA$

$B \rightarrow bB$

2.化为:

$A \rightarrow a(d|Ac)$

$A \rightarrow bBc$

$B \rightarrow aA$

$B \rightarrow bB$

3.化为:

$A \rightarrow aA'$

$A \rightarrow bBc$

$A' \rightarrow d$

$A' \rightarrow Ac$

$B \rightarrow aA$

$B \rightarrow bB$



◦ 例4.8

文法G3[S] :

$S \rightarrow aSd$

$S \rightarrow Ac$

$A \rightarrow aS$

$A \rightarrow b$

1.化为:

$S \rightarrow aSd$

$S \rightarrow aSc$

$S \rightarrow bc$

$A \rightarrow aS$

$A \rightarrow b$

2.化为:

$S \rightarrow aS(d|c)$

$S \rightarrow bc$

$A \rightarrow aS$

$A \rightarrow b$

3.化为:

$S \rightarrow aSA'$

$S \rightarrow bc$

$A' \rightarrow d$

$A' \rightarrow c$

结果中A是不可达到的符号。



◦ 例4.9

文法G4[S] :

$S \rightarrow Ap | Bq$

$A \rightarrow aAp | d$

$B \rightarrow aBq | e$

1.化为：

$S \rightarrow aApp | aBqq | dp | eq$

$A \rightarrow aAp | d$

$B \rightarrow aBq | e$

2.化为：

$S \rightarrow a(App | Bqq)$

$S \rightarrow dp | eq$

$A \rightarrow aAp | d$

$B \rightarrow aBq | e$

3.化为：

$S \rightarrow aS'$

$S \rightarrow dq | eq$

$S' \rightarrow App | Bqq$

$A \rightarrow aAp | d$

$B \rightarrow aBq | e$

4.化为：

$S \rightarrow aS'$

$S \rightarrow dp | eq$

$S' \rightarrow$

$aAppp | aBqqq | dpp | eqq$

$A \rightarrow aAp | d$

$B \rightarrow aBq | e$



## ◦ 结论

不一定每个文法的左公共因子都能在有限的步骤内替换成无左公共因子的文法。

一个文法提取了左公共因子后，只解决了相同左部产生式右部的FIRST集不相交问题，当改写后的文法不含空产生式，且无左递归时，则改写后的文法是LL(1)文法，否则还需用LL(1)文法的判别方式进行判断才能确定是否为LL(1)文法。



## 2. 消除左递归

- 直接左递归

$A \rightarrow A\beta \quad A \in VN, \beta \in V^*$

- 间接左递归

$A \rightarrow B\beta$

$B \rightarrow A\alpha \quad A, B \in VN, \alpha, \beta \in V^*$



④ 例4.10

G5[S]:

S→Sa

S→b

考慮对输入串baaaa#的分析

S→bS'

S'→aS'|ε

⑤ 例4.11

G6[A]:

A→aB

A→Bb

B→Ac

B→d

考慮对输入串adbcdbc#的分析

B→aBc

B→Bbc

B→d

B→aBcB' | dB'

B'→bcB'| ε



## 消除直接左递归

◦  $A \rightarrow A\alpha_1 | A\alpha_2 | \dots | A\alpha_m | \beta_1 | \beta_2 | \dots | \beta_n$

其中：  $\alpha_i$  不等于  $\epsilon$ ，  $\beta_j$  不以  $A$  开头。

改为：

$A \rightarrow \beta_1 A' | \beta_2 A' | \dots | \beta_n A'$

$A' \rightarrow \alpha_1 A' | \alpha_2 A' | \dots | \alpha_m A' | \epsilon$



## 消除间接左递归

- 将间接左递归变为直接左递归，然后消除直接左递归。



# 消除文法中一切左递归的算法

1. 以某种顺序将文法非终结符排列 $A_1, A_2 \dots A_n$

2. for  $i:=1$  to  $n$  do

begin

for  $j:=1$  to  $i-1$  do

用 $A_j \rightarrow \alpha_1 | \alpha_2 | \dots | \alpha_k$ 关于 $A_j$ 的全部产生式替代

形如 $A_i \rightarrow A_j r$ 的产生式为:

$A_i \rightarrow \alpha_1 r | \alpha_2 r | \dots | \alpha_k r;$

end;

消除 $A_i$ 的直接左递归;

3. 化简由2得到的文法



◦ 例

文法**G[S]**:

**S**→**Qc|c**

**Q**→**Rb|b**

**R**→**Sa|a**

将非终结符排序为**S、 Q、 R**

**R**→ **Qca|ca|a**

**R**→ **Rbca|bca|ca|a**

**R**→ **(bca|ca|a)R'**

**R'** → **bcaR'|ε**



## ✧ 课堂练习：左递归消除

原文法  $G[E]$ :  $E \rightarrow E + T \mid T$   
 $T \rightarrow T * F \mid F$   
 $F \rightarrow (E) \mid a$

消除左递归后的文法  $G'[E]$ :

- (1)  $E \rightarrow TE'$
- (2)  $E' \rightarrow + TE'$
- (3)  $E' \rightarrow \varepsilon$
- (4)  $T \rightarrow FT'$
- (5)  $T' \rightarrow * FT'$
- (6)  $T' \rightarrow \varepsilon$
- (7)  $F \rightarrow (E)$
- (8)  $F \rightarrow a$



## 4. 4不确定的自顶向下分析思想（自学）

### ✧ 举例

– 单词序列 aaab 的一个自顶向下分析过程

	S	(1)
文法 G (S) :	$\Rightarrow AB$	(2)
(1) $S \rightarrow AB$	$\Rightarrow aAB$	(3)
(2) $A \rightarrow aA$	$\Rightarrow aB$	(4)
(3) $A \rightarrow \epsilon$	$\Rightarrow ab$	(回溯)
(4) $B \rightarrow b$	.....	
(5) $B \rightarrow bB$		

复杂度很高

失败条件较复杂



## 4.5 确定的自顶向下分析方法

- 递归子程序法
- 预测分析法

借助于预测分析表  
和符号栈。

- 实现思想
  - 对文法中每个非终结符编写一个递归过程，每个过程的功能是识别由该非终结符推出的串，当某非终结符的产生式有多个候选时能够按LL(1)形式可唯一地确定选择某个候选进行推导。
- 限制
  - 对文法要求高，必须满足LL(1)文法；
  - 速度慢，占用空间多。



# 递归下降 LL(1) 分析程序

## ✧ 非终结符对应的递归下降子程序

### — 一般结构

设  $A$  的产生式:

$$A \rightarrow u_1 | u_2 | \dots | u_n,$$

相对于非终结符  $A$   
的递归下降子程序  
ParseA 的一般结  
构

```
void ParseA()
{
    switch (lookahead) {
        case S( $A \rightarrow u_1$ ):
            /* code to recognize  $u_1$  */
            break;
        case S( $A \rightarrow u_2$ ):
            /* code to recognize  $u_2$  */
            break;
        ...
    }
}
```



# 递归下降 LL(1) 分析程序

$G(S): S \rightarrow AaS \mid BbS \mid d$

$A \rightarrow a$

$B \rightarrow \epsilon \mid c$

$Select(S \rightarrow AaS) = \{a\}$

void ParseS()

$Select(S \rightarrow BbS) = \{c, b\}$

{

$Select(S \rightarrow d) = \{d\}$

switch (lookahead) {

case a:

ParseA();

MatchToken(a);

ParseS();

break;

case b,c:

ParseB();

MatchToken(b);

ParseS();

break;

case d:

MatchToken(d);

break;

default:

printf("syntax error \n")

exit(0);

}

}



# 递归下降 LL(1) 分析程序

$G(S): S \rightarrow AaS \mid BbS \mid d$

$A \rightarrow a$

$B \rightarrow \epsilon \mid c$

$Select(A \rightarrow a) = \{a\}$

$Select(B \rightarrow \epsilon) = \{b\}$

$Select(B \rightarrow c) = \{c\}$

void ParseB( )

{

    if (lookahead==c) {

        MatchToken(c);

}

    else if (lookahead==b) {

}

    else {

        printf("syntax error \n");

        exit(0);

}

}

void ParseA( )

{

    if (lookahead==a) {

        MatchToken(a);

}

    else {

        printf("syntax error \n");

        exit(0);

}

}



# PL/O 编译程序的语法分析

## 递归子程序的设计实例

- <表达式> ::= [ + | - ] <项> { ( + | - ) <项> }

```
int expression(...)  
{  
    if (sym==plus || sym==minus) /* 此时表达式被看作正的或负的项 */  
        getsym();  
        term (...); /* 处理<项> */  
    }  
    else /* 此时表达式被看作项的加减 */  
    {  
        term (...); /* 处理<项> */  
    }  
    while (sym==plus || sym==minus)  
    {  
        getsym;  
        term (...); /* 处理<项> */  
    }  
    return 0;  
}
```



# PL/O 编译程序的语法分析

## 递归子程序的设计实例

- <项> ::= <因子> { (\* | /) <因子> }

```
int term(...)  
{  
    factor (...);      /*处理<因子>*/  
    while (sym==times || sym==slash)  
    {  
        getsym();  
        factor (...);    /* 处理<因子> */  
    }  
    return 0;  
}
```



# PL/0 编译程序的语法分析

## 递归子程序的设计实例

- <因子> ::= <标识符> | <无符号整数> | ‘(’ <表达式> ‘)’

```
int factor (...)  
{  
    if (sym==ident)      /* <因子>为常量或变量 */  
        getsym();  
    else if (sym==number) /* <因子>为立即数 */  
        getsym();  
    else if (sym==lparen); /* <因子>为立即数 */  
    {  
        expression(...);  
        if (sym==rparen)  
            getsym();  
        else  
            error(22); /*提示22号出错信息：缺少右括  
号*/  
    }  
    return 0;  
}
```



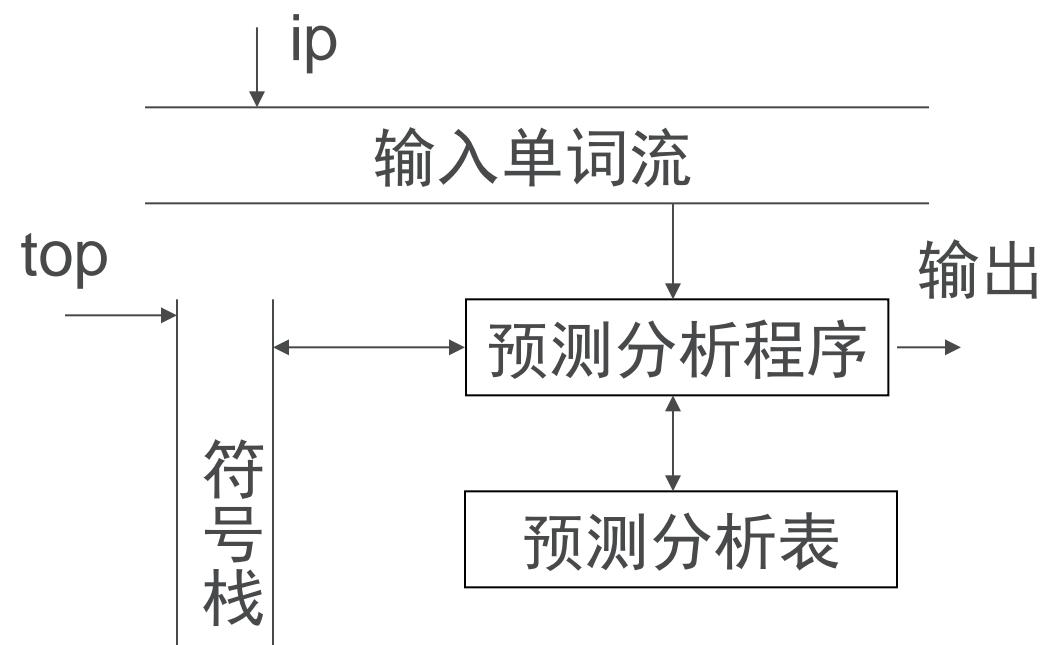
# 预测分析方法

- 一个预测分析器是由三个部分组成

预测分析程序

符号栈

预测分析表



# 预测分析表

- 预测分析表可用一个矩阵M表示。矩阵的元素 $M[A, a]$ 中的内容为一条关于A的产生式，表明当用非终结符A向下推导且面临输入符a时，所应采取的候选产生式，当元素内容无产生式时，则出错。
- 一个文法的预测分析表不含有多重入口，当且仅当该文法是LL(1)的。
- 预测分析表的构造算法

若 $a \in \text{SELECT}(A \rightarrow \alpha)$ ，则把 $A \rightarrow \alpha$ 放入 $M[A, a]$ 中。

把所有无定义的 $M[A, a]$ 标上出错标记。



- 例文法  $G[E]$ :
 
$$E \rightarrow TE'$$

$$E' \rightarrow +TE' \mid \epsilon$$

$$T \rightarrow FT'$$

$$T' \rightarrow *FT' \mid \epsilon$$

$$F \rightarrow i \mid ( E )$$

	i	+	*	(	)	#
E	$\rightarrow TE'$			$\rightarrow TE'$		
E'		$\rightarrow +TE'$			$\rightarrow \epsilon$	$\rightarrow \epsilon$
T	$\rightarrow FT'$			$\rightarrow FT'$		
T'		$\rightarrow \epsilon$	$\rightarrow *FT'$		$\rightarrow \epsilon$	$\rightarrow \epsilon$
F	$\rightarrow i$			$\rightarrow (E)$		

构造过程：

- 判断文法是否为LL(1)文法
- 构造预测分析表



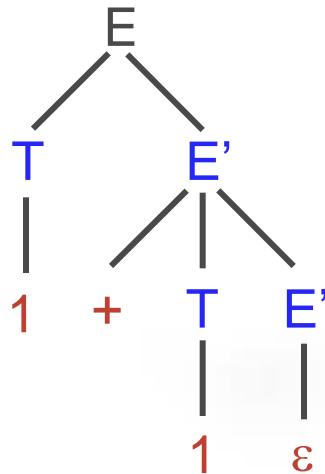
## 加法表达式文法:

$$E \rightarrow TE'$$

$$E' \rightarrow +TE' \mid \epsilon$$

$$T \rightarrow id$$

成功?  
失败?



输入

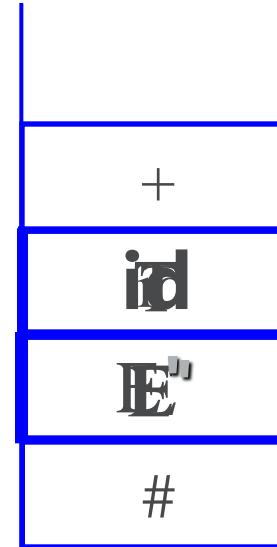
1 + 1 #

输出

预测分析程序

初始化

查找 找到



分析栈

栈顶  
栈顶

分析  
成功!

	id	+	#
E	$\rightarrow TE'$	不应为 +	无效句子
E'	不应为 终结符	$\rightarrow +TE'$	$\rightarrow \epsilon$
T	$\rightarrow id$	不应为 +	无效 句子



## 例 对输入串 $i+i^*i$ 的分析

	i	+	*	(	)	#
E	$\rightarrow TE'$			$\rightarrow TE'$		
E'		$\rightarrow +TE'$			$\rightarrow \epsilon$	$\rightarrow \epsilon$
T	$\rightarrow FT'$			$\rightarrow FT'$		
T'		$\rightarrow \epsilon$	$\rightarrow *FT'$		$\rightarrow \epsilon$	$\rightarrow \epsilon$
F	$\rightarrow i$			$\rightarrow (E)$		

分析栈	剩余串	产生式
#E	i+i*i#	$E \rightarrow TE'$
#E'T	i+i*i#	$T \rightarrow FT'$
#E'T'F	i+i*i#	$F \rightarrow i$
#E'T'i	i+i*i#	i 匹配
#E'T'	+i*i#	$T' \rightarrow \epsilon$
#E'	+i*i#	$E' \rightarrow +TE'$
#E'T+*	+i*i#	+ 匹配
#E'T	i*i#	$T \rightarrow FT'$
#E'T'F	i*i#	$F \rightarrow i$
#E'T'i	i*i#	i 匹配
#E'T'	*i#	$T \rightarrow *FT'$
#E'T'F*	*i#	* 匹配
#E'T'F	i#	$F \rightarrow i$
#E'T'i	i#	i 匹配
#E'T'	#	$T' \rightarrow \epsilon$
#E'	#	$E' \rightarrow \epsilon$
#	#	接受



# 表驱动 LL(1) 分析程序

## ✧ 表驱动预测分析过程举例

- 对于下列文法  $G(S)$ :

$S \rightarrow AaS \mid BbS \mid d$       剩余的输入串

$A \rightarrow a$

$aabd\#$

$B \rightarrow \epsilon \mid c$

分析输入串  $aabd$  的过程:

**S**  
#



# 表驱动 LL(1) 分析程序

## ✧ 表驱动预测分析过程举例

- 对于下列文法  $G(S)$ :

$S \rightarrow AaS \mid BbS \mid d$  剩余的输入串

$A \rightarrow a$

$aabd\#$

$B \rightarrow \epsilon \mid c$

A  
a  
S  
#

分析输入串  $aabd$  的过程:

	a	b	c	d	#
S	$S \rightarrow AaS$	$S \rightarrow BbS$	$S \rightarrow BbS$	$S \rightarrow d$	
A	$A \rightarrow a$				
B		$B \rightarrow \epsilon$	$B \rightarrow c$		



# 表驱动 LL(1) 分析程序

## ✧ 表驱动预测分析过程举例

- 对于下列文法  $G(S)$ :

$S \rightarrow AaS \mid BbS \mid d$       剩余的输入串

$A \rightarrow a$

$B \rightarrow \epsilon \mid c$

*aabd#*

**a  
a  
S  
#**

分析输入串 *aabd* 的过程:

	a	b	c	d	#
S	$S \rightarrow AaS$	$S \rightarrow BbS$	$S \rightarrow BbS$	$S \rightarrow d$	
A	$A \rightarrow a$				
B		$B \rightarrow \epsilon$	$B \rightarrow c$		



# 表驱动 LL(1) 分析程序

## ✧ 表驱动预测分析过程举例

- 对于下列文法  $G(S)$ :

$S \rightarrow AaS \mid BbS \mid d$  剩余的输入串

$A \rightarrow a$

**abd#**

$B \rightarrow \epsilon \mid c$

**a S #**

分析输入串 **aabd** 的过程:

	a	b	c	d	#
S	$S \rightarrow AaS$	$S \rightarrow BbS$	$S \rightarrow BbS$	$S \rightarrow d$	
A	$A \rightarrow a$				
B		$B \rightarrow \epsilon$	$B \rightarrow c$		



# 表驱动 LL(1) 分析程序

## ✧ 表驱动预测分析过程举例

- 对于下列文法  $G(S)$ :

$S \rightarrow AaS \mid BbS \mid d$  剩余的输入串

$A \rightarrow a$

$bd\#$

$B \rightarrow \epsilon \mid c$

**S**  
#

分析输入串 **aabd** 的过程:

	a	b	c	d	#
S	$S \rightarrow AaS$	$S \rightarrow BbS$	$S \rightarrow BbS$	$S \rightarrow d$	
A	$A \rightarrow a$				
B		$B \rightarrow \epsilon$	$B \rightarrow c$		



# 表驱动 LL(1) 分析程序

## ✧ 表驱动预测分析过程举例

- 对于下列文法  $G(S)$ :

$S \rightarrow AaS \mid BbS \mid d$  剩余的输入串

$A \rightarrow a$

$bd\#$

$B \rightarrow \epsilon \mid c$

**B**  
**b**  
**S**  
#

分析输入串 **aabd** 的过程:

	a	b	c	d	#
S	$S \rightarrow AaS$	$S \rightarrow BbS$	$S \rightarrow BbS$	$S \rightarrow d$	
A	$A \rightarrow a$				
B		$B \rightarrow \epsilon$	$B \rightarrow c$		



# 表驱动 LL(1) 分析程序

## ✧ 表驱动预测分析过程举例

- 对于下列文法  $G(S)$ :

$S \rightarrow AaS \mid BbS \mid d$  剩余的输入串

$A \rightarrow a$

$bd\#$

$B \rightarrow \epsilon \mid c$

**b**  
**S**  
#

分析输入串 **aabd** 的过程:

	a	b	c	d	#
S	$S \rightarrow AaS$	$S \rightarrow BbS$	$S \rightarrow BbS$	$S \rightarrow d$	
A	$A \rightarrow a$				
B		$B \rightarrow \epsilon$	$B \rightarrow c$		



# 表驱动 LL(1) 分析程序

## ✧ 表驱动预测分析过程举例

- 对于下列文法  $G(S)$ :

$S \rightarrow AaS \mid BbS \mid d$       剩余的输入串

$A \rightarrow a$

$d\#$

$B \rightarrow \epsilon \mid c$

**S**  
#

分析输入串 **aabd** 的过程:

	a	b	c	d	#
S	$S \rightarrow AaS$	$S \rightarrow BbS$	$S \rightarrow BbS$	$S \rightarrow d$	
A	$A \rightarrow a$				
B		$B \rightarrow \epsilon$	$B \rightarrow c$		



# 表驱动 LL(1) 分析程序

## ✧ 表驱动预测分析过程举例

- 对于下列文法  $G(S)$ :

$S \rightarrow AaS \mid BbS \mid d$  剩余的输入串

$A \rightarrow a$

$d\#$

$B \rightarrow \epsilon \mid c$

**$d$**   
**#**

分析输入串  **$aabd$**  的过程:

	a	b	c	d	#
S	$S \rightarrow AaS$	$S \rightarrow BbS$	$S \rightarrow BbS$	$S \rightarrow d$	
A	$A \rightarrow a$				
B		$B \rightarrow \epsilon$	$B \rightarrow c$		



# 表驱动 LL(1) 分析程序

## ✧ 表驱动预测分析过程举例

- 对于下列文法  $G(S)$ :

$S \rightarrow AaS \mid BbS \mid d$  剩余的输入串

$A \rightarrow a$

#

$B \rightarrow \epsilon \mid c$



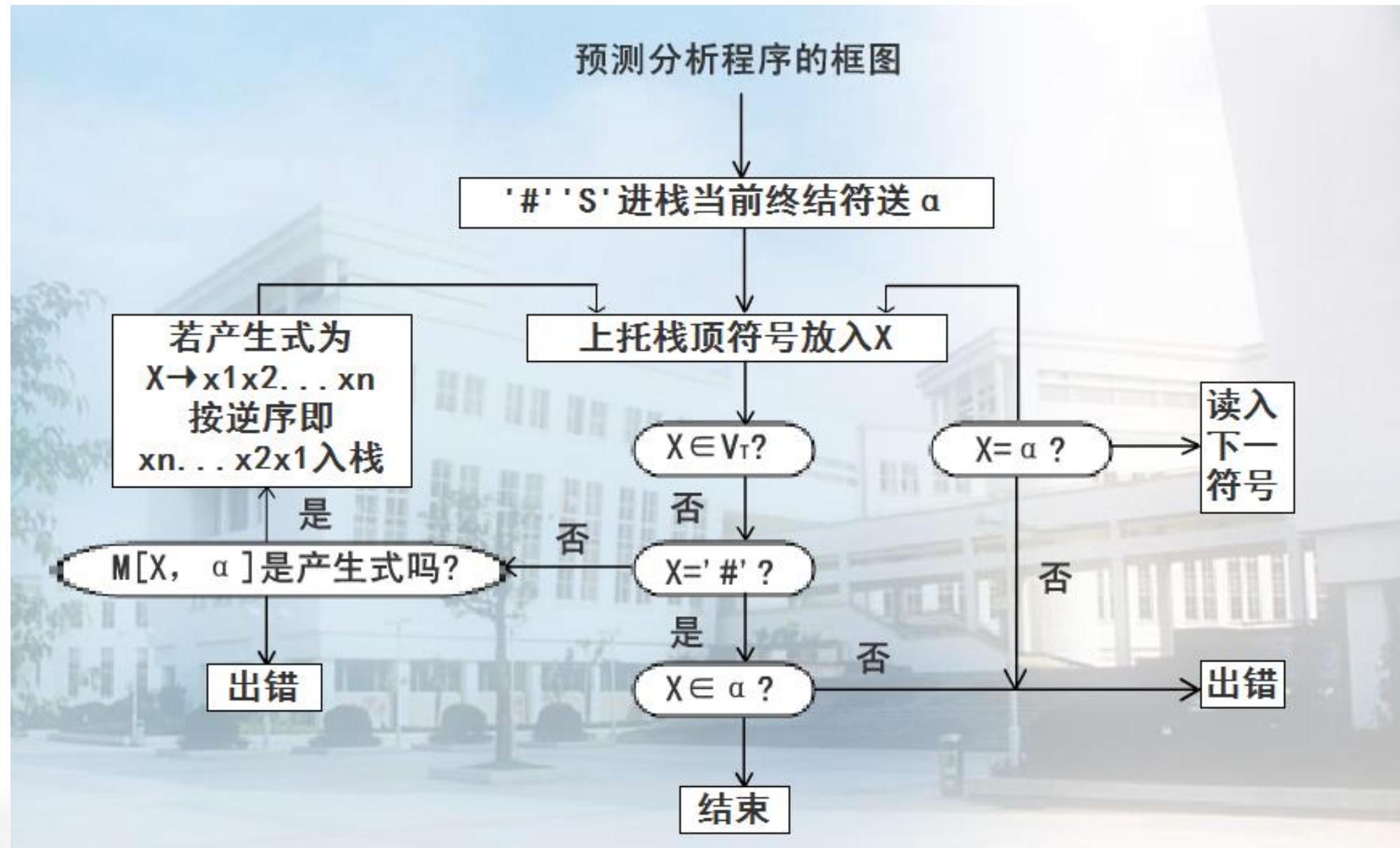
#

分析输入串 **aabd** 的过程:

	a	b	c	d	#
S	$S \rightarrow AaS$	$S \rightarrow BbS$	$S \rightarrow BbS$	$S \rightarrow d$	
A	$A \rightarrow a$				
B		$B \rightarrow \epsilon$	$B \rightarrow c$		



# 预测分析程序算法



✧思考题：

- 1 如何对一个给定的文法判断是否是LL(1)文法；
- 2 如何构造预测分析表；
- 3 如何用预测分析方法判断给定的输入符号串是否 是该文法的句子；
- 4 是不是对所有的非LL(1)文法做等价变换后可能变成LL(1)文法，如果是怎么变换？如果不是，那需要什么条件？

## LL( $K$ )文法- 推广LL(1)文法



## 课堂练习

已知文法G【S】：  $S \rightarrow aH$

$H \rightarrow aMd | d$

$M \rightarrow Ab | \epsilon$

$A \rightarrow aM | e$

1. 判断文法是否为LL(1)文法,若是, 请构造预测分析表。
2. 请给出输入串aaabd#的预测分析过程, 并说明该输入串是否是G[S]的句子。

## 正规式考题：

- 1. 设有语言  $L = \{ a \mid a \in \{0,1\}^+ \text{, 且 } a \text{ 不以 } 0 \text{ 开头, 但以 } 00 \text{ 结尾} \}$ 。
  - (1) 试写出描述  $L$  的正规表达式。 (2分)
  - (2) 构造识别  $L$  的 最小的DFA (要求先画出构造过程中的NFA, 再转化为DFA。如果转化得到的DFA不是最小的DFA, 则将其最小化)。 (8分)
  - (3) 根据 (2) 构造的DFA, 写出相应的正规文法。 (4分)

