

**华中科技大学**

---

# **编译原理**

## **第4章 自顶向下语法分析方法**

**2019年11月27日星期三**

---

**编译原理课程组**

---

## 内容摘要

**本章研究自顶向下语法分析方法(即推导法)，它分为不确定的语法分析方法和确定的语法分析方法两类。**

**主要介绍确定的自顶向下语法分析方法，重点讨论这类分析方法应满足“文法是LL(1)文法”的这个适用条件、LL(1)文法判别、构造方法以及这类分析方法的两种实现技术：递归子程序法和LL(1)预测法。**

---

## 重点讲解

4.1 确定的自顶向下语法分析思想

4.2 LL(1)文法的判别

4.3 某些非LL(1)文法到LL(1)文法的等价变换

4.4 确定的自顶向下语法分析方法

4.5 典型例题及解答

4.6 LL(1)分析中的出错处理

---

## 4.1 确定的自顶向下语法分析思想

---

自顶向下语法分析方法(即推导法)是从文法开始符 $S$ 出发,逐步进行推导,以证实 $S \Rightarrow \alpha$ 的推导过程是否存在的方法。

问题是每步推导会面临两次多种可能选择:

- (1) 选择句型中哪一个非终结符进行推导
- (2) 选择非终结符的哪一个规则进行推导

问题(1)可以采用最左推导解决。问题(2)通常需要穷举每一个规则的可能推导,即**不确定的自顶向下语法分析**。具体思想是:

一旦寻找到一个符号串 $\alpha$ 之推导过程,便结束穷举过程,断定符号串 $\alpha$ 是句子。

只有当穷举全部可能的推导,而没有一个符号串 $\alpha$ 之推导过程的时候,才可以断定符号串 $\alpha$ 不是句子。

---

## 确定的自顶向下语法分析思想

---

对于问题(2) “**选择非终结符U的哪一个规则进行推导**”，  
选择**唯一**的可能推导出输入串  $\alpha$  的规则进行推导。

如果没有一个可能推导出输入串  $\alpha$  的规则，则结束推导，  
宣告输入串  $\alpha$  不是句子。

**说明：**

- ① “**唯一**”意味着非终结符U的其它任意规则，不可能推导出输入串  $\alpha$ 。
- ② “**唯一**”意味着每次选择非终结符U哪一个规则时，选择是“确定的”。

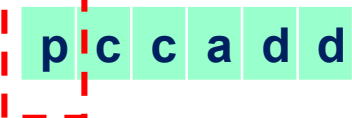
什么类型的文法，才能做到这样的“**唯一**”呢？下面讨论文法应该满足的条件。

## 最左推导举例1

例4.1 设文法 $G_1[S]$ 定义如下，考察输入串pccadd的最左推导过程。

$$\begin{aligned}G_1[S]: S &\rightarrow pA \mid qB \\ A &\rightarrow cAd \mid a \\ B &\rightarrow dB \mid b\end{aligned}$$

推 导:  $S \Rightarrow pA \Rightarrow pcAd \Rightarrow pccAdd \Rightarrow pccadd$



成功使用最左推导推导出符号串

## 最左推导举例2

例4.2 设文法 $G_2[S]$ 定义如下，考察输入串ccap的最左推导过程。

$G_2[S]: S \rightarrow Ap \mid Bq$

$A \rightarrow cA \mid a$

$B \rightarrow dB \mid b$

$Ap \xRightarrow{*} c \dots$

$Bq \not\xRightarrow{*} c \dots$

推 导:  $S \Rightarrow A p \Rightarrow c A p \Rightarrow c c A p \Rightarrow c c a p$

c c a p

成功使用最左推导推导出符号串

## FIRST集的定义

定义 4.1 设文法 $G = (V_N, V_T, P, S)$ ，则

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

特别地， $\alpha \xRightarrow{*} \epsilon$ ，约定  $\epsilon \in \text{FIRST}(\alpha)$ 。

例 设文法 $G[S]: S \rightarrow Ap \mid Bq, A \rightarrow cA \mid a, B \rightarrow dB \mid b$ ，则

$$\text{FIRST}(cA) = \{c\}$$

$$\text{FIRST}(A) = \{c, a\}$$

$$\text{FIRST}(Ap) = \{c, a\}$$

$$\text{FIRST}(Bq) = \{d, b\}$$

**FIRST( $\alpha$ )是由 $\alpha$ 可以推导以终结符号开头符号串的头符号集合。如果所有非终结符右部的FIRST集合两两相交为空，可以使用确定的最左推导。**



## 最左推导举例3

例4.3 设文法G3[S]定义如下，考察输入串abd的最左推导过程。

G3[S]:  $S \rightarrow aA \mid d$

$A \rightarrow bAS \mid \epsilon$

$bAS \xRightarrow{*} b\dots$   
 $\epsilon \not\xRightarrow{*} b\dots$

推 导:  $S \Rightarrow a A \Rightarrow a b A S \Rightarrow a b S \Rightarrow a b d$

a b d

此时  $S \xRightarrow{*} abAS$ , A除空规则外, 其它所有规则都不可能推导出d开头的符号序列, 则此刻采用  $A \rightarrow \epsilon$  才是唯一有可能推导abd的选择!

## FOLLOW集的定义

在最左推导中，一旦句型的最左非终结符A，除空规则外，其它所有规则都不可能推导出由输入符（假定为 $a_1$ ）开头的符号序列，这时使用空规则，意味着将匹配d的工作交给了句型A之后的部分，也就是后面这部分要能推导出以 $a_1$ 开头的符号串才有可能匹配成功。

如果输入符号d能在某个句型中能出现在A的后面，可以使用了A的空规则再继续分析，否则即使使用了A的空规则，后面的推导也不可能匹配。所以需要文法进行分析，文法的所有句型中，任意一个非终结符后可能出现的终结符的集合。

一般形式：

输入串： $a_1 a_2 \cdots a_{i-1} a_i \cdots a_n$

句型推导： $S \xRightarrow{*} a_1 a_2 \cdots a_{i-1} A \beta$

如果使用空规则，意味着需要： $\beta \xRightarrow{*} a_i \cdots a_n$

则有句型： $S \xRightarrow{*} a_1 a_2 \cdots a_{i-1} A a_i \cdots a_n$

## FOLLOW集的定义

---

定义 4.2 设文法 $G=(V_N, V_T, P, S)$ , 则

$$\text{FOLLOW}(A) = \{a \mid S \xRightarrow{*} \alpha A \beta, A \in V_N, a \in \text{FIRST}(\beta), \alpha, \beta \in V^*\}$$

(或者:  $\text{FOLLOW}(A) = \{a \mid S \xRightarrow{*} \cdots Aa \cdots, A \in V_N, a \in V_T\}$ )

$$G3[S]: S \rightarrow aA \mid d$$

$$A \rightarrow bAS \mid \epsilon$$

**FOLLOW(A)**是由任意句型中紧邻非终结符号A之后出现的终结符号a组成的集合。

如果对非终结符A, 有一条空规则, 则A的FOLLOW集合和A的非空右部的FIRST集合两两相交为空, 可以使用确定的最左推导。

## 最左推导举例4

例4.4  $G_4[S]: S \rightarrow aA \mid d$   
 $A \rightarrow bAS \mid B$   
 $B \rightarrow c \mid \varepsilon$

$bAS \xrightarrow{*} c \dots$   
 $B \xrightarrow{*} \varepsilon$   
 $B \xrightarrow{*} c$

(1) 输入串: **a c**

推 导:  $S \Rightarrow a A \Rightarrow a B \Rightarrow a c$

只能选择规则:  $A \rightarrow B$   
 由B的FIRST集确定选

$bAS \xrightarrow{*} b \dots$   
 $d \in \text{FOLLOW}(A)$   
 $B \xrightarrow{*} \varepsilon$   
 $B \xrightarrow{*} c$

(2) 输入串: **abd**

推 导:  $S \Rightarrow a A \Rightarrow a b A S \Rightarrow a b S \Rightarrow a b d$

只能选择规则:  $A \rightarrow B$   
 由A的FOLLOW集确定选规则

## SELECT集的定义

---

使用统一的方法来选择使用规则，即当某规则右部能推导出空时，将其FIRST和FOLLOW这2个集合合并考虑，以确定在什么情况下选择该规则。

**定义 4.3** 设文法 $G=(V_N, V_T, P, S)$ ， $A \in V_N$ ， $A \rightarrow \alpha \in P$ ，则

$$\text{SELECT}(A \rightarrow \alpha) = \begin{cases} \text{FIRST}(\alpha) & (\alpha \xRightarrow{*} \epsilon) \\ (\text{FIRST}(\alpha) - \{\epsilon\}) \cup \text{FOLLOW}(A) & (\alpha \not\xRightarrow{*} \epsilon) \end{cases}$$

$\text{SELECT}(A \rightarrow \alpha)$ 称为规则 $A \rightarrow \alpha$ 的选择集。它是 $\text{FIRST}(\alpha)$ 和 $\text{FOLLOW}(A)$ 组成，是终结符号集 $V$ 的子集。

## LL(1)文法的定义

**定义 4.4** 文法G是**LL(1)文法**的充分必要条件是文法G每个

$U \rightarrow \alpha_1 \mid \alpha_2 \mid \cdots \mid \alpha_n$  规则，满足下列条件：

$$\text{SELECT}(U \rightarrow \alpha_i) \cap \text{SELECT}(U \rightarrow \alpha_j) = \Phi$$

$$(i \neq j, 1 \leq i \leq n, 1 \leq j \leq n)$$

**确定的自顶向下语法分析思想：**

假定文法G是LL(1)文法，在 $S \xrightarrow{*} \alpha$  最左推导过程中，遇到选用U规则时，如果输入串  $\alpha$  的当前符号a，属于某个U规则的

确定的自顶向下语法分析不必穷举所有的推导过程，避免了回溯现象，极大地提高了语法分析的效率。这里“确定的”意指选择规则的确定性。这类分析法，也称为不带回溯的自顶向下语法分析。

## LL(1)文法的句型分析举例

**例4.1** 设文法 $G_1[S]$ 定义如下, 考察输入串pccadd的确定的自顶向下语法分析过程。

$G_1[S]: S \rightarrow pA \mid qB, A \rightarrow cAd \mid a, B \rightarrow dB \mid b$

$\therefore \text{SELECT}(S \rightarrow pA) = \{p\}, \text{SELECT}(S \rightarrow qB) = \{q\}$

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

$\text{SELECT}(B \rightarrow dB) = \{d\}, \text{SELECT}(B \rightarrow b) = \{b\}$

$\therefore \text{SELECT}(S \rightarrow pA) \cap \text{SELECT}(S \rightarrow qB) = \Phi$

$\text{SELECT}(A \rightarrow cAd) \cap \text{SELECT}(A \rightarrow a) = \Phi$

$\text{SELECT}(B \rightarrow dB) \cap \text{SELECT}(B \rightarrow b) = \Phi$

即文法 $G_1[S]$ 是LL(1)文法。

**语法分析过程:**

输入串: pccadd

推导:  $S \Rightarrow pA \Rightarrow pcAd \Rightarrow pccAdd \Rightarrow pccadd$

$\therefore a \in \text{SELECT}(A \rightarrow a)$

$\therefore$  选择  $A \rightarrow a$

## 4.2 LL(1)文法的判别

---

采用确定的自顶向下语法分析方法，必须判别文法是否是LL(1)。判别的实质是SELECT集的计算，SELECT集的计算又归结于可推导 $\epsilon$ 的非终结符的计算、FIRST集的计算和FOLLOW集的计算。

### 4.2.1 计算可推导 $\epsilon$ 的非终结符

对于一个文法，很容易证明以下事实：如果非终结符规则的每个右部至少含有一个终结符，则该非终结符不可以推导出 $\epsilon$ ；如果非终结符规则的某个右部，或者是 $\epsilon$ ，或者是均能推导出 $\epsilon$ 的非终结符串组成，则该非终结符能推导出 $\epsilon$ 。据此，可推导 $\epsilon$ 的非终结符计算方法如下。

---



## 4.2.1 计算可推导 $\epsilon$ 的非终结符

设文法  $G = (V_N, V_T, P, S)$ ，计算步骤是：

- (1)  $X[]$  的每个数组元素置初值为“未定”；
- (2) 删除规则所有右部至少含有一个终结符的规则。若删除后某非终结符  $A$  的所有规则都被删除，置  $X[A]$  为“否”；
- (3) 对所有空规则 ( $A \rightarrow \epsilon$ )，置  $X[A]$  为“是”，并删除左部为  $A$  所有规则；
- (4) 如果非终结符  $A$  的某规则右部均由  $X[]$  标记为“是”的非终结符组成，置  $X[A]$  为“是”，并删除左部为  $A$  所有规则；
- (5) 删除右部含有  $X[]$  标记为“否”的非终结符的规则，若删除后某非终结符  $A$  的所有规则被删除，置  $X[A]$  为“否”；
- (6) 重复(4)、(5)，直到  $X[]$  不再变化为止。

$X[]$  是一个非终结符为下标、元素个数为  $|V_N|$  的一维数组。数组元素  $X[A]$  取值为“未定”、“否”和“是”，分别表示非终结符  $A$  还不确定能否可推导  $\epsilon$ 、确定不可推导  $\epsilon$  和确定可推导  $\epsilon$ 。

## 4.2.2 计算FIRST(X)集

---

设文法 $G = (V_N, V_T, P, S)$ ,  $X \in V_N \cup V_T$ , FIRST(X)初值为空集。计算FIRST(X)集步骤是:

- (1) 对于所有终结符号 $x$ ,  $\text{FIRST}(x) = \{x\}$ ;
- (2) 对于所有空规则 $X \rightarrow \epsilon$ ,  $\text{FIRST}(X) \cup = \{\epsilon\}$ ;
- (3) 对于所有形如 $X \rightarrow a \dots$ 规则, 且 $a \in V_T$ ,  $\text{FIRST}(X) \cup = \{a\}$ ;
- (4) 对于所有形如 $X \rightarrow Y_1 Y_2 \dots Y_n$ 规则,

如果 $Y_1 \xRightarrow{*} \epsilon$ 、 $Y_2 \xRightarrow{*} \epsilon$ 、 $\dots$ 、 $Y_{i-1} \xRightarrow{*} \epsilon$  ( $i \leq n$ ), 则

$\text{FIRST}(X) \cup = (\text{FIRST}(Y_1) \cup \text{FIRST}(Y_2) \dots \cup \text{FIRST}(Y_i)) - \{\epsilon\}$ ;

如果 $Y_1 \xRightarrow{*} \epsilon$ 、 $Y_2 \xRightarrow{*} \epsilon$ 、 $\dots$ 、 $Y_n \xRightarrow{*} \epsilon$  则

$\text{FIRST}(X) \cup = (\text{FIRST}(Y_1) \cup \text{FIRST}(Y_2) \dots \cup \text{FIRST}(Y_i)) \cup \{\epsilon\}$ ;

- (5) 重复(4), 直到FIRST()不再扩大为止。
-

### 4.2.3 计算FIRST( $\alpha$ )集

---

设文法 $G = (V_N, V_T, P, S)$ ，已知 $\text{FIRST}(X)$  ( $X \in V_N \cup V_T$ )。  
 $\alpha = Y_1 Y_2 \cdots Y_n \in (V_N \cup V_T)^*$ ， $\text{FIRST}(\alpha)$ 初值为空集。计算 $\text{FIRST}(\alpha)$ 集的方法是：

如果  $Y_1$  为终结符, 则  $\text{FIRST}(\alpha) = \{ Y_1 \}$ ;

如果  $Y_1 \xRightarrow{*} \epsilon$ 、 $Y_2 \xRightarrow{*} \epsilon$ 、 $\cdots$ 、 $Y_{i-1} \xRightarrow{*} \epsilon$  ( $1 < i \leq n$ )，则

$\text{FIRST}(\alpha) = (\text{FIRST}(Y_1) \cup \text{FIRST}(Y_2) \cdots \cup \text{FIRST}(Y_i)) - \{ \epsilon \}$ ;

如果  $Y_1 \xRightarrow{*} \epsilon$ 、 $Y_2 \xRightarrow{*} \epsilon$ 、 $\cdots$ 、 $Y_n \xRightarrow{*} \epsilon$  则

$\text{FIRST}(\alpha) = (\text{FIRST}(Y_1) \cup \text{FIRST}(Y_2) \cdots \cup \text{FIRST}(Y_n)) \cup \{ \epsilon \}$ 。

---

## 4.2.4 计算FOLLOW集

---

设文法 $G = (V_N, V_T, P, S)$ 。  $X \in V_N$ ， $FOLLOW(X)$ 初值为空集。计算 $FOLLOW(X)$ 步骤是：

(1) 置 $FOLLOW(S) = \{\#\}$

(2) 对所有规则，按下列情况分别计算：

如果 $A \rightarrow \alpha B \beta$  规则，且 $B \in V_N$ ，

则： $FOLLOW(B) \cup = (FIRST(\beta) - \{\epsilon\})$ ；

如果 $\beta \xRightarrow{*} \epsilon$ ，则 $FOLLOW(B) \cup = FOLLOW(A)$ ；

(3) 重复(2)，直到 $FOLLOW()$ 不再扩大为止。

## LL (1) 文法的判定

例4.5 设文法 $G[S]$ 定义如下，判别 $G[S]$ 是否是LL(1)文法。

$$\begin{aligned} G[S]: S &\rightarrow AB \mid bC \\ A &\rightarrow b \mid \epsilon \\ B &\rightarrow aD \mid \epsilon \\ C &\rightarrow AD \mid b \\ D &\rightarrow aS \mid c \end{aligned}$$

### 1. 计算可推出 $\epsilon$ 非终结符号

S	A	B	C	D
是	是	是	否	否

### 2. 计算非终结符号的FIRST(X)集

FIRST()							
S	A	B	C	D	a	b	c
b, a, $\epsilon$	$\epsilon$ , b	$\epsilon$ , a	b, a, c	a, c	a	b	c

# LL (1) 文法的判定

## 3. 计算规则右部的FIRST( $\alpha$ )集

FIRST()									
$S \rightarrow AB$	$S \rightarrow bC$	$A \rightarrow b$	$A \rightarrow \epsilon$	$B \rightarrow aD$	$B \rightarrow \epsilon$	$C \rightarrow AD$	$C \rightarrow b$	$D \rightarrow aS$	$D \rightarrow c$
b,a, $\epsilon$	b	b	$\epsilon$	a	$\epsilon$	b,a,c	b	a	c

## 4. 计算非终结符号的FOLLOW(A)集

FOLLOW ( )				
S	A	B	C	D
#	a c #	#	#	#

重复无变化

$FOLLOW(S) \leftarrow FOLLOW(S) \cup FOLLOW(D)$

$FOLLOW(A) \leftarrow FOLLOW(A) \cup FOLLOW(S) \cup$

$(FIRST(B) \cup FIRST(D) - \{\epsilon\})$

$FOLLOW(B) \leftarrow FOLLOW(B) \cup FOLLOW(S)$	a	b	c
$FOLLOW(C) \leftarrow FOLLOW(C) \cup FOLLOW(S)$	a	b	c

## LL (1) 文法的判定

### 5. 计算规则的SELECT集

$$\text{SELECT}(S \rightarrow AB) = (\text{FIRST}(AB) - \{ \epsilon \}) \cup \text{FOLLOW}(S) = \{ \#, a, b \}$$

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

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

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

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

$$\text{SELECT}(B \rightarrow \epsilon) = (\text{FIRST}(\epsilon) - \{ \epsilon \}) \cup \text{FOLLOW}(B) = \{ \# \}$$

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

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

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

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

$$\text{SELECT}(S \rightarrow AB) \cap \text{SELECT}(S \rightarrow bC) = \{ b \} \neq \Phi$$

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

$$\text{SELECT}(B \rightarrow aD) \cap \text{SELECT}(B \rightarrow \epsilon) = \{ b \} \neq \Phi$$

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

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

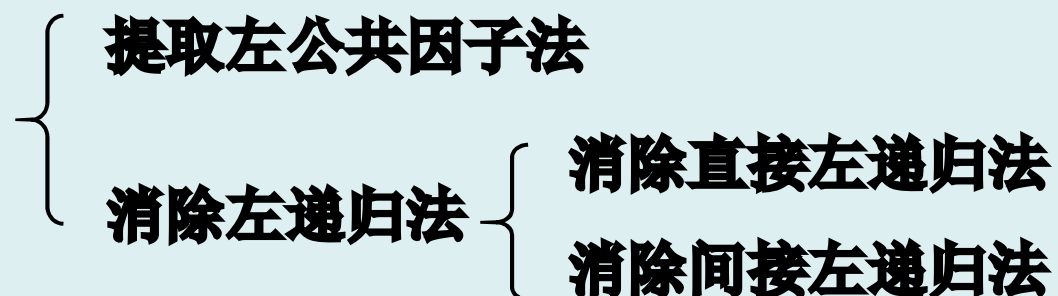
$\therefore$  文法G[S]不是LL(1)文法。

## 4.3 某些非LL(1)文法到LL(1)文法的等价变换

---

某些非LL(1)文法G，可以通过等价变换成LL(1)文法G'。但下面讨论的等价变换方法，仅仅确保变换的等价性（即 $L(G) = L(G')$ ），不能保证变换后的文法G'一定是LL(1)文法。因此，对于变换后的文法G'，必须判别它是LL(1)文法后，方可使用确定的自顶向下语法分析方法。

两种等价变换：





### 4.3.1 提取左公共因子法

$$A \rightarrow \alpha \beta_1 \mid \alpha \beta_2 \mid \cdots \mid \alpha \beta_n \mid \gamma_1 \mid \gamma_2 \mid \cdots \mid \gamma_m$$



$$\begin{aligned} A &\rightarrow \alpha B \mid \gamma_1 \mid \gamma_2 \mid \cdots \mid \gamma_m \\ B &\rightarrow \beta_1 \mid \beta_2 \mid \cdots \mid \beta_n \end{aligned}$$

**注解：**上下组的A红色标记的规则，均推导出如下结果。

$$\alpha \{ \beta_1, \beta_2, \cdots, \beta_n \}$$

**例4.6** 设文法  $G[S]: S \rightarrow aSb \mid aS \mid \epsilon$ 。试采用提取左公共因子法，得到与之等价的文法  $G' [S]$ 。

$\because$  文法仅有一个非终结符S，其规则有左公共因子

$$\alpha = aS, \beta_1 = b, \beta_2 = \epsilon, \gamma_1 = \epsilon。$$

$$\therefore \text{文法 } G' [S]: S \rightarrow aSA \mid \epsilon, A \rightarrow b \mid \epsilon。$$

## 4.3.2 消除左递归法

定义 4.5 设文法 $G = (V_N, V_T, P, S)$ ，形如 $A \rightarrow \alpha A \beta$ 的规则称为文法 $G$ 的直接递归规则。特别地，如果 $\alpha = \epsilon$ 时，则称为文法 $G$ 的**直接左递归规则**。如果 $\beta = \epsilon$ 时，则称为文法 $G$ 的**直接右递归规则**。

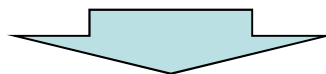
定义 4.6 设文法 $G = (V_N, V_T, P, S)$ ，如果存在推导 $A \Rightarrow \alpha \Rightarrow \lambda A \mu$ ，则规则 $A \rightarrow \alpha$ 称为文法 $G$ 的**间接递归规则**。特别地，如果 $\lambda = \epsilon$ 时，则称为文法 $G$ 的**间接左递归规则**。如果 $\mu = \epsilon$ 时，则称为文法 $G$ 的**间接右递归规则**。

通常，直接递归规则和间接递归规则统称**递归规则**，直接左递归规则和间接左递归规则统称**左递归规则**，直接右递归规则和间接右递归规则统称**右递归规则**。从定义可得到，直接递归规则可以认为是特殊的间接递归规则。

定义 4.7 含有递归规则（直接递归规则或间接递归规则）的文法。称为递归文法。特别地，如果含有左递归规则的文法，则称为**左递归文法**。如果含有右递归规则的文法，则称为**右递归文法**。

## (1) 消除直接左递归法

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



$$\begin{aligned} A &\rightarrow \beta_1 A' \mid \beta_2 A' \mid \cdots \mid \beta_n A' \\ A' &\rightarrow \alpha_1 A' \mid \alpha_2 A' \mid \cdots \mid \alpha_m A' \mid \epsilon \end{aligned}$$

**注解：**上下组的A规则，均推导出如下结果。

$$\{\beta_1, \beta_2, \cdots, \beta_n\} \{\alpha_1, \alpha_2, \cdots, \alpha_m\}^*$$

**例4.7** 设文法 $G[S]$   $S \rightarrow Sb \mid a$ ，试求不含直接左递归的等价文法 $G'[S]$ 。

令 $\alpha_1 = b$ ， $\beta_1 = a$ ，采用消除直接左递归法，得

$$\text{文法 } G'[S]: S \rightarrow a S', S' \rightarrow b S' \mid \epsilon$$

很容易验证，文法 $G'[S]$ 是LL(1)文法。

## (2) 消除左递归法

---

消除左递归法基本思想是利用代入法，将间接左递归规则变换为等价的直接左递归规则，之后利用上述消除直接左递归法，再将直接左递归规则消除。

具体步骤如下：

(1) 将文法所有非终结符按任意顺序线性排列：

$A_1, A_2, \dots, A_i, A_{i+1}, \dots, A_n$

(2) 消除 $A_1$ 的直接左递归；

(3) 对从1到 $n-1$ 的每个 $i$ ，做

(3.1) 消除 $A_{i+1}$ 的间接左递归：

依次将 $A_1, A_2, \dots, A_i$  的规则，代入 $A_{i+1}$  的规则，并替代 $A_{i+1}$  原规则；

(3.2) 消除 $A_{i+1}$ 的直接左递归；

(4) 删除无用规则。

---

## (2) 消除左递归法

---

**例4.8** 设文法 $G[S]$ 定义如下，试消除其左递归，得到与之等价的文法 $G' [S]$ 。

$$G[S]: S \rightarrow Qc \mid c$$
$$Q \rightarrow Rb \mid b$$
$$R \rightarrow Sa \mid a$$

采用消除左递归法，得到文法 $G' [S]$ 如下。

$$G' [S]: S \rightarrow Qc \mid c$$
$$Q \rightarrow Rb \mid b$$
$$R \rightarrow bcaR' \mid caR' \mid aR'$$
$$R' \rightarrow bcaR' \mid \epsilon$$

**注解：**不同非终结符排列顺序，得到不同的等价文法 $G'$ 。

---

## 4.4 确定的自顶向下语法分析方法

---

### 4.4.1 递归子程序法

递归子程序法是将每个非终结符编写成一个递归子程序，即语法分析程序的每个递归子程序完成选择规则、推导和匹配的功能。

在递归子程序中，选择规则的实现步骤是将输入串“下一个符号”逐个与A规则的选择集进行判定，“下一个符号”属于哪个选择集，便选择相应规则推导。只有当“下一个符号”不属于任何选择集时，报告语法错误。

按照递归子程序法构造的语法分析程序是由一个总控子程序和一组非终结符对应的递归子程序组成的。

---

## 4.4.1 递归子程序法

例4.9 设文法G[E]定义如下，试设计其语法分析递归子程序

$$\begin{aligned} G[E] : E &\rightarrow eBa\dot{A} \\ A &\rightarrow a \mid bAcB \\ B &\rightarrow dEd \mid aC \\ C &\rightarrow e \mid dC \end{aligned}$$

(1) 计算SELECT结果如下，文法G[E] 显然是LL(1)文法

$$SELECT(eBaA) = \{e\}$$

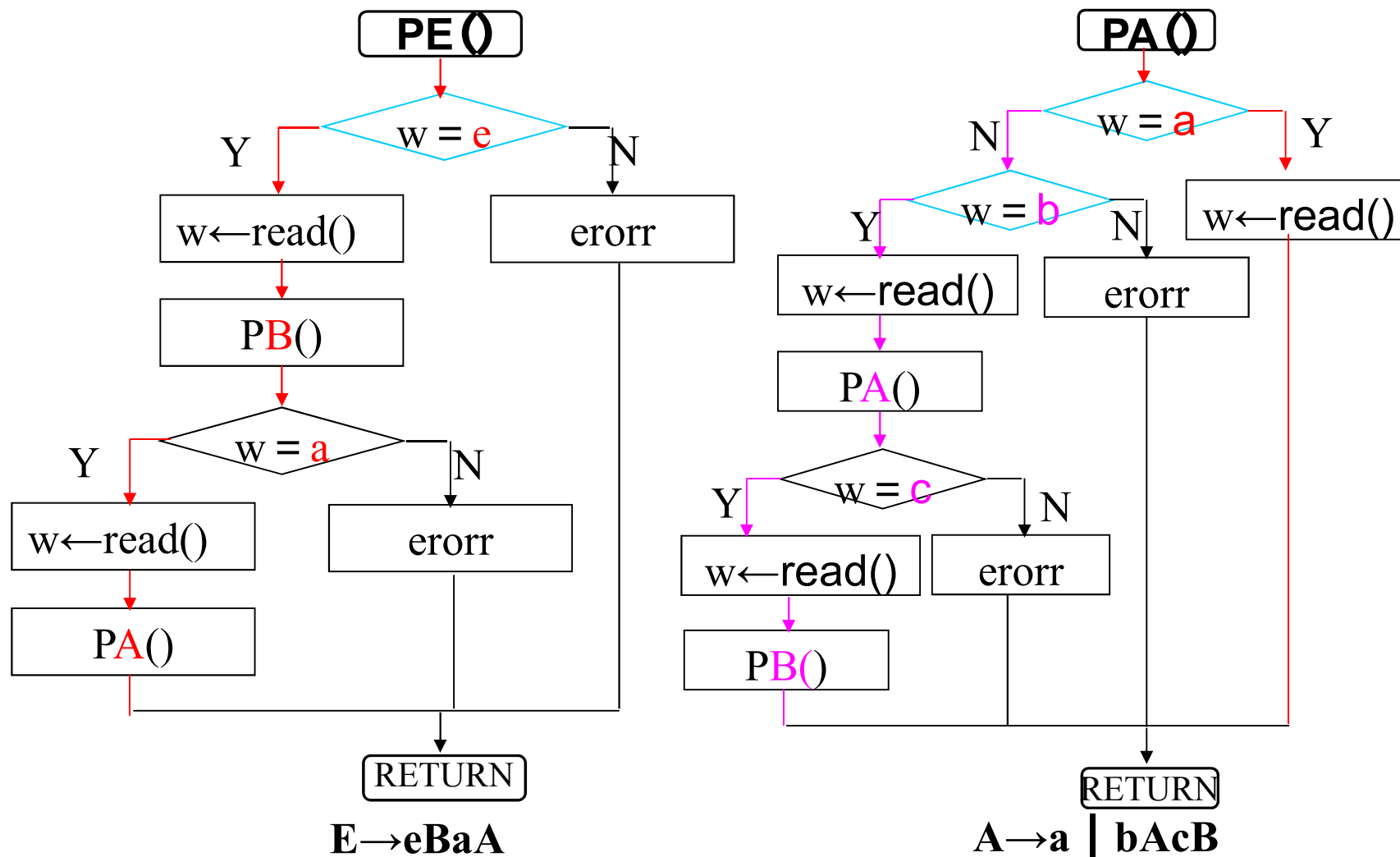
$$SELECT(A \rightarrow a) = \{a\}, \quad SELECT(A \rightarrow bAcB) = \{b\}$$

$$SELECT(B \rightarrow dEd) = \{d\}, \quad SELECT(B \rightarrow aC) = \{a\}$$

$$SELECT(C \rightarrow dC) = \{d\}$$

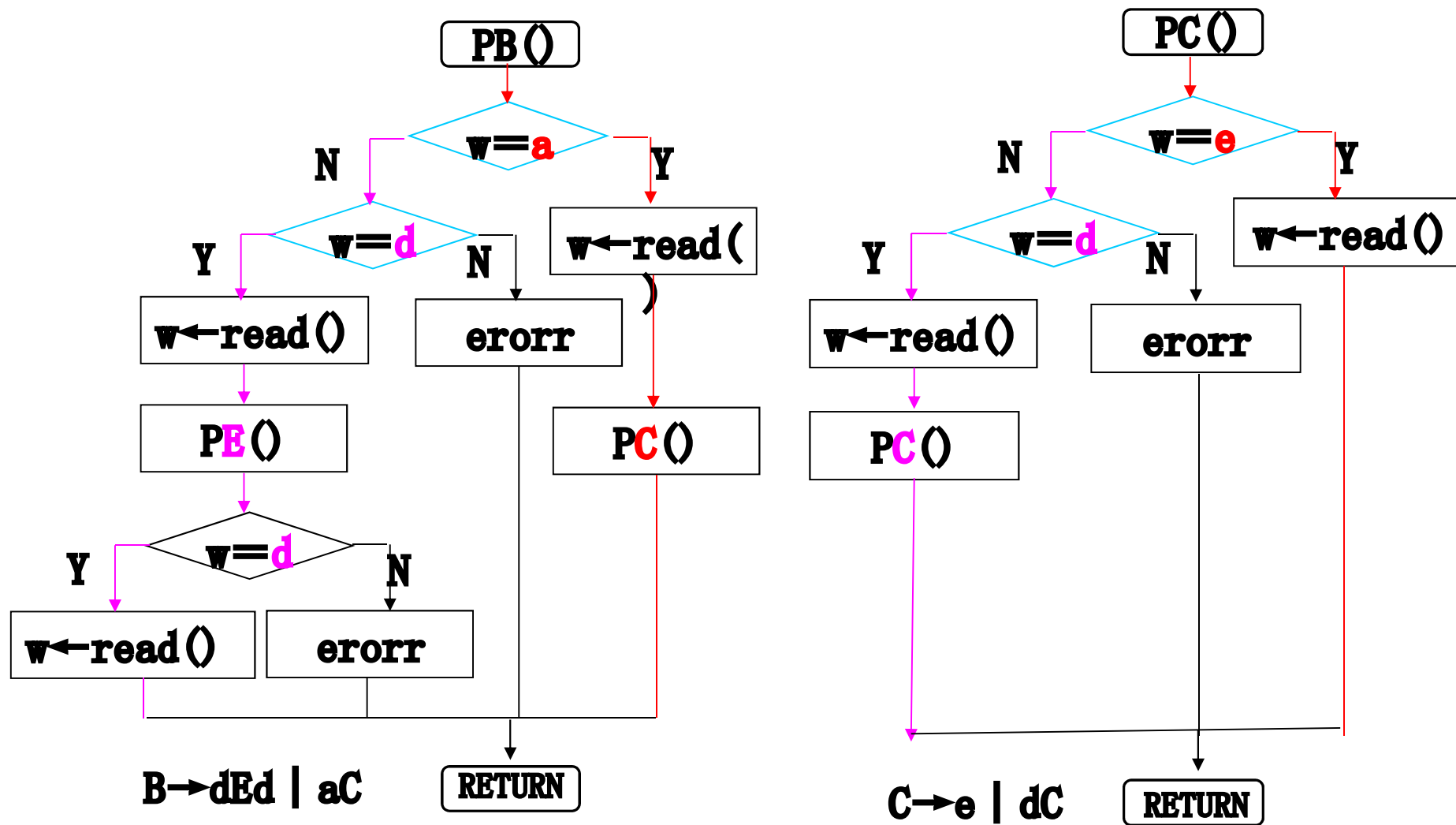
(2) 设计递归子程序如下。

# 递归子程序法举例



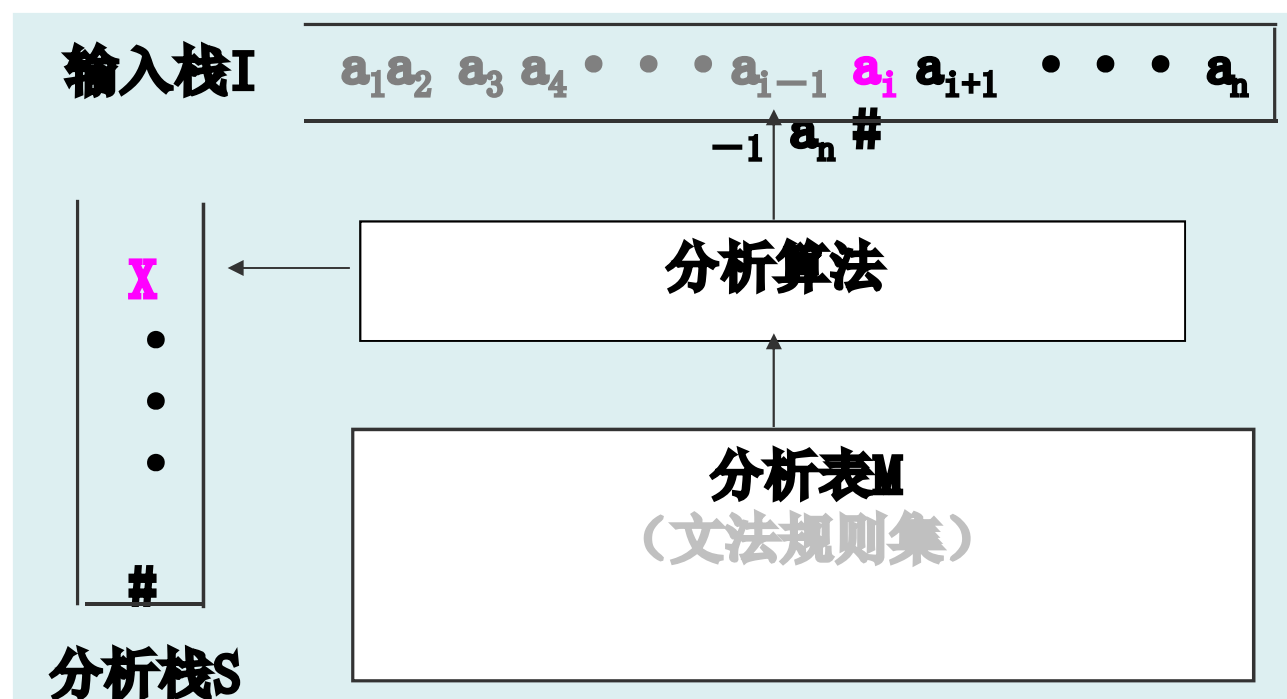


# 递归子程序法举例



## 4.4.2 预测分析法

预测分析法构造语法分析程序的总体框架如下图所示。将输入串视为以串末端为底的栈I，输入串未匹配部分为栈的内容，这个栈称为“输入栈”；推导过程产生的句型未匹配部分，依自右向左顺序，也存放在另一个称为栈S中，这个栈称为“分析栈”；再将**规则选择集**，存放在一个非终结符为行、终结符为列和元素为规则的二维表M中，这个表称为“分析表”。



## 预测分析法举例

---

例4.10 设文法G[E]定义如下，试构造预测分析表，并给出输入串*i+i\*i*的分析过程。

$$\begin{aligned} E &\rightarrow E+T \mid T \\ T &\rightarrow T*F \mid F \\ F &\rightarrow i \mid (E) \end{aligned}$$

左递归文法，消除左递归后：

$$\begin{aligned} E &\rightarrow TE' \\ E' &\rightarrow +TE' \mid \epsilon \\ T &\rightarrow FT' \\ T' &\rightarrow *FT' \mid \epsilon \\ F &\rightarrow i \mid (E) \end{aligned}$$

## 预测分析法举例

(1) 计算SELECT结果如下，文法G[E] 显然是LL(1)文法。

$\text{SELECT}(E \rightarrow TE') = \{i, (\}$

$\text{SELECT}(E' \rightarrow +TE') = \{+\}$ ,  $\text{SELECT}(E' \rightarrow \epsilon) = \{), \#\}$

$\text{SELECT}(T \rightarrow FT') = \{i, (\}$

$\text{SELECT}(T' \rightarrow *FT') = \{*\}$ ,  $\text{SELECT}(T' \rightarrow \epsilon) = \{+, ), \#\}$

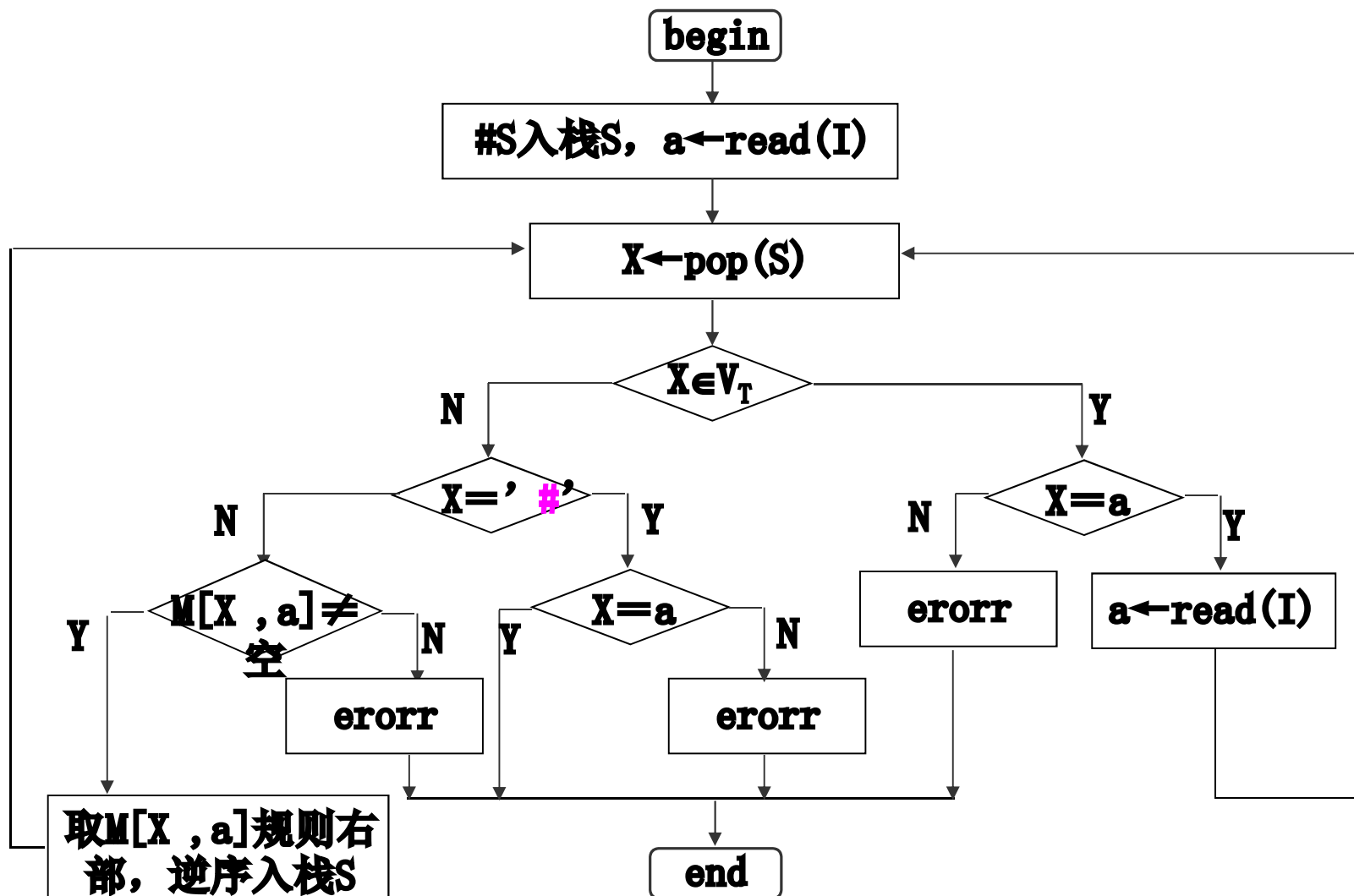
$\text{SELECT}(F \rightarrow (E)) = \{(\}$ ,  $\text{SELECT}(F \rightarrow i) = \{i\}$

(2) 构造预测分析表结果如下。

$\begin{matrix} P \\ V_N \end{matrix} \backslash V_T$	i	+	*	(	)	#
E	$E \rightarrow TE'$			$E \rightarrow TE'$		
E'		$E' \rightarrow +TE'$			$E' \rightarrow \epsilon$	$E' \rightarrow \epsilon$
T	$T \rightarrow FT'$			$T \rightarrow FT'$		
T'		$T' \rightarrow \epsilon$	$T' \rightarrow *FT'$		$T' \rightarrow \epsilon$	$T' \rightarrow \epsilon$
F	$F \rightarrow i$			$F \rightarrow (E)$		

(3) 输入串i+i\*i的分析过程

# 预测分析法算法流程



## 4.5 典型例题及解答

---

例4.11: 已知文法 $G[S]$

$$\begin{array}{l} S \rightarrow Sa \mid A \\ A \rightarrow bA \mid b \end{array}$$

(1) 该文法有左递归, 消除后:

$$\begin{array}{l} S \rightarrow AS' \\ S' \rightarrow aS' \mid \epsilon \\ A \rightarrow bA \mid b \end{array}$$

(2) 该文法有公共左因子, 提取后:

$$\begin{array}{l} S \rightarrow AS' \\ S' \rightarrow aS' \mid \epsilon \\ A \rightarrow bA' \\ A' \rightarrow A \mid \epsilon \end{array}$$

(3)  $S, A$ 不能推导出空,  $S', A'$ 可以推导出空

## 4.5 典型例题及解答

### (3) 分析FIRST集和FOLLOW集

$S \rightarrow AS'$   
 $S' \rightarrow aS' \mid \varepsilon$   
 $A \rightarrow bA'$   
 $A' \rightarrow A \mid \varepsilon$

非终结符	FIRST	FOLLOW
S	b	#
S'	a, $\varepsilon$	#
A	b	a, #
A'	b, $\varepsilon$	a, #

### (4) 分析规则的SELECT集

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

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

$\text{SELECT}(S' \rightarrow \varepsilon) = \{\#\}$

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

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

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

改写后成为LL(1)文法。

## 4.5 典型例题及解答

(5) 构造LL(1)  
分析表

非终结符	a	b	#
S		$\rightarrow AS'$	
S'	$S' \rightarrow aS'$		$\rightarrow \varepsilon$
A		$\rightarrow bA'$	
A'	$\rightarrow \varepsilon$	$\rightarrow A$	$\rightarrow \varepsilon$

步骤	符号栈	剩余输入串	动作
1	#S	ba#	$\rightarrow AS'$
2	#S' A	ba#	$\rightarrow bA'$
3	#S' A' b	ba#	b匹配
4	#S' A'	a#	$\rightarrow \varepsilon$
5	#S'	a#	$S' \rightarrow aS'$
6	#S' a	a#	a匹配
7	#S'	#	$\rightarrow \varepsilon$
8	#	#	分析成功



## 4.6 LL(1)分析中的出错处理

---

### 错误处理的两个任务

- (1) 报错，发现错误时应尽可能准确指出错误位置与错误属性；
- (2) 错误回复，尽可能进行校正，使编译工作能继续下去，提高程序调试效率。

### 4.6.1 应急恢复

在预测分析法中，有两类需要报错：

- (1) 栈顶的终结符与输入符号不匹配；
  - (2) 非终结符A位于栈顶，面临的输入符号位a，但分析表的表项M[A, a]为空。
-

## 4.6.1 应急恢复

---

**处理方法：在空的 $M[A, a]$ 指定同步符号，一旦遇到这种错误，就跳过输入符号，直到遇到同步符号为止。**

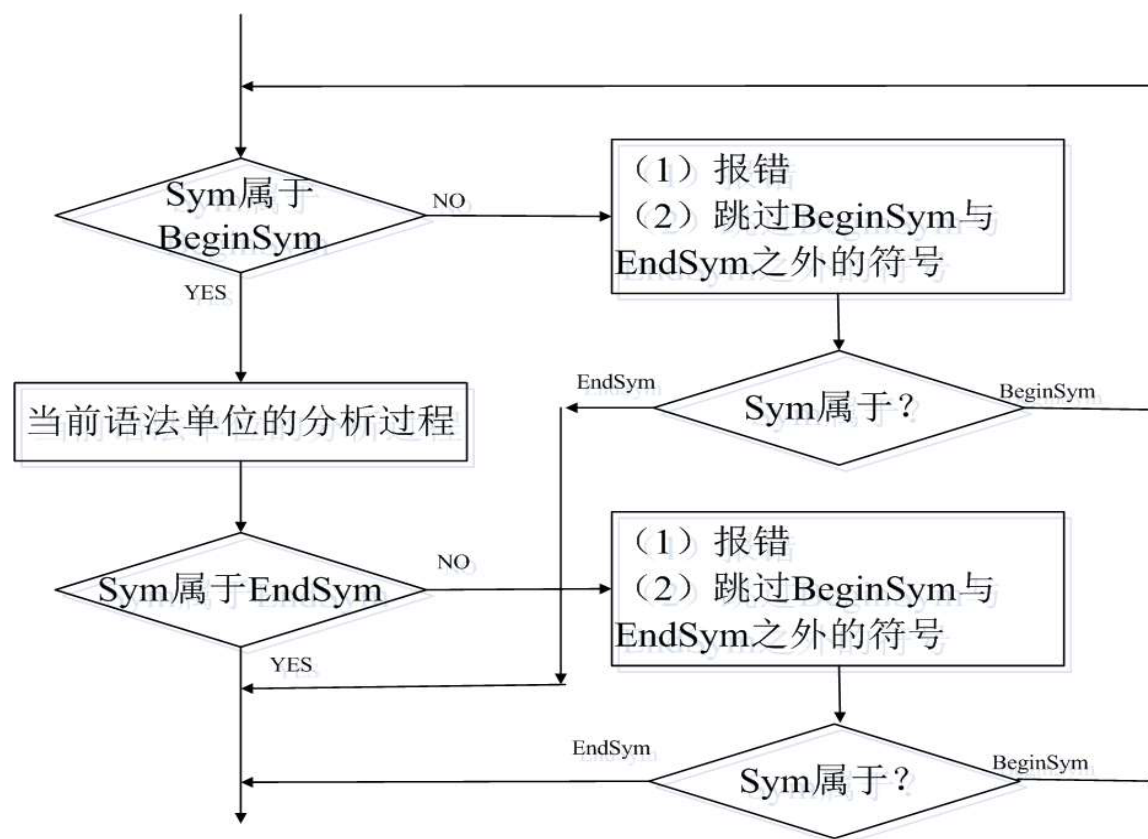
**当分析栈顶为A，一旦错误发生：**

**(1) 跳过输入串中的一些符号，直到遇到 $FOLLOW(A)$ 中的符号，然后A退栈。相当于跳过A能推导出的所有符号，A的推导完成。**

**(2) 跳过输入串中的一些符号，直到遇到 $FIRST(A)$ 中的符号。相当于从A开始推导。**

## 4.6.2 短语恢复

每个非终结符，开始分析时，使用一个符号集合BeginSym；  
分析结束时，使用一个符号集合EndSym。



## 本章小结

---

本章主要介绍确定的自顶向下语法分析方法，重点讨论这类分析方法应满足“文法是LL(1)文法”的这个适用条件、LL(1)文法判别和构造方法以及这类分析方法的两种实现技术：递归子程序法和LL(1)预测法。

采用LL(1)预测法构造语法分析程序时，其语法分析算法是通用的。其技术线路是：

- ① 依据给定的源语言，设计其上下文无关文法，并计算选择集SELECT()判定文法是否是LL(1)文法；
- ② 如果不是LL(1)文法，则可以提取左公共因子法和消除左递归法进行等价转换，或重新设计文法，直到是LL(1)文法；
- ③ 之后，根据选择集SELECT()，构造LL(1)分析表。

## 本章小结

---

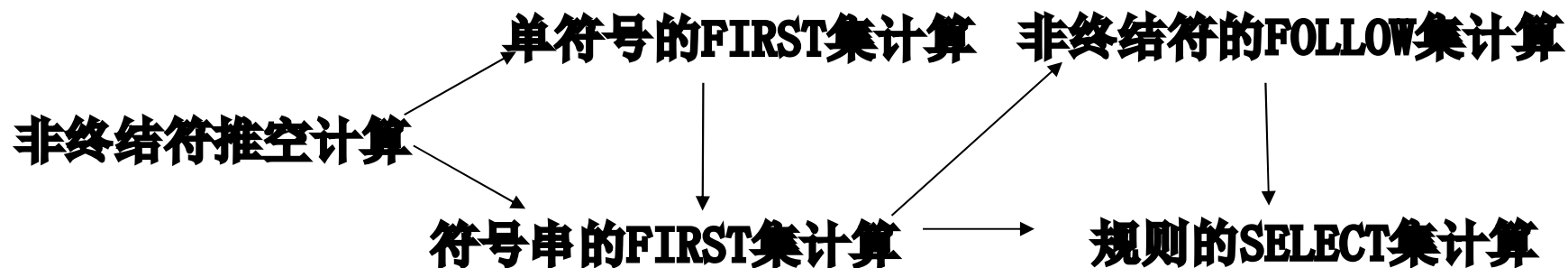
采用递归子程序法构造语法分析程序的技术线路是：

- ① 依据给定的源语言，设计其上下文无关文法，并计算选择集SELECT()判定文法是否是LL(1)文法；
- ② 如果不是LL(1)文法，则可以提取左公共因子法和消除左递归法进行等价转换，或重新设计文法，直到是LL(1)文法；
- ③ 之后，根据选择集SELECT()，对于每个非终结符，设计一个相应的语法分析递归子程序。

## 本章小结

### 重点掌握的内容:

- ① 计算选择集SELECT()
- ② LL(1)文法判别
- ③ 采用提取左公共因子法和消除左递归法等价转换文法
- ④ 构造LL(1)分析表
- ⑤ LL(1)分析算法
- ⑥ 设计非终结符相应的语法分析递归子程序



各种计算之间依赖关系图