

§ 4.1.3 无左递归无回溯的自顶向下分析

- 递归下降分析---高级语言或汇编语言实现
- LL(1)分析法---用一个分析表和分析栈实现

1、递归下降分析(递归子程序, 推导过程)

实现方式:

对每一个非终结符号U,编写一个子程序F(U)

F(U):boolean

true:分析过程正常(得以匹配)

false:分析过程出错(无法匹配)

例：G[E]：

$E \rightarrow T | EAT$ 改写文法： $E \rightarrow T \{AT\}$

$T \rightarrow F | TMF$ $T \rightarrow F \{MF\}$

$T \rightarrow (E) | i$ $F \rightarrow (E) | i$

$A \rightarrow + | -$ $A \rightarrow + | -$

$M \rightarrow * | /$ $M \rightarrow * | /$

(1)消除左递归：有左递归

改写后的文法

G[E]:

$E \rightarrow TE'$

$E' \rightarrow ATE' \mid \varepsilon$

$T \rightarrow FT'$

$T' \rightarrow MFT' \mid \varepsilon$

$F \rightarrow (E)i$

$A \rightarrow +|-$

$M \rightarrow *|/$

(2) 消除回溯:无回溯

产生式	FIRST(α)	FOLLOW(A)
$E \rightarrow TE'$	{ (, i }	{), # }
$E' \rightarrow ATE'$	{ +, - }	{), # }
$E' \rightarrow \varepsilon$	{ ε }	
$T \rightarrow FT'$	{ (, i }	{ +, -,), # }
$T' \rightarrow MFT'$	{ *, / }	{ +, -,), # }
$T' \rightarrow \varepsilon$	{ ε }	
$F \rightarrow (E)$	{ (}	{ +, -, *, / ,), # }
$F \rightarrow i$	{ i }	
$A \rightarrow +$	{ + }	{ (, i }
$A \rightarrow -$	{ - }	
$M \rightarrow *$	{ * }	{ (, i }
$M \rightarrow /$	{ / }	

递归子程序的框图

设:

- current 中放置当前正扫描的输入符号.
- advance 表示输入符号指针后移一位.

假定:

- 当进入某子程序时,要分析的输入符号已经在 current 中.
- 在从某一子程序退出时,下一个要分析的输入符号放入 current 中.

$E \rightarrow TE'$

$F(E)$

$FIRST(TE') = \{i, (\}$

Current=i或(

N

Y

CALL T

T=true

N

Y

CALL E'

E' =true

N

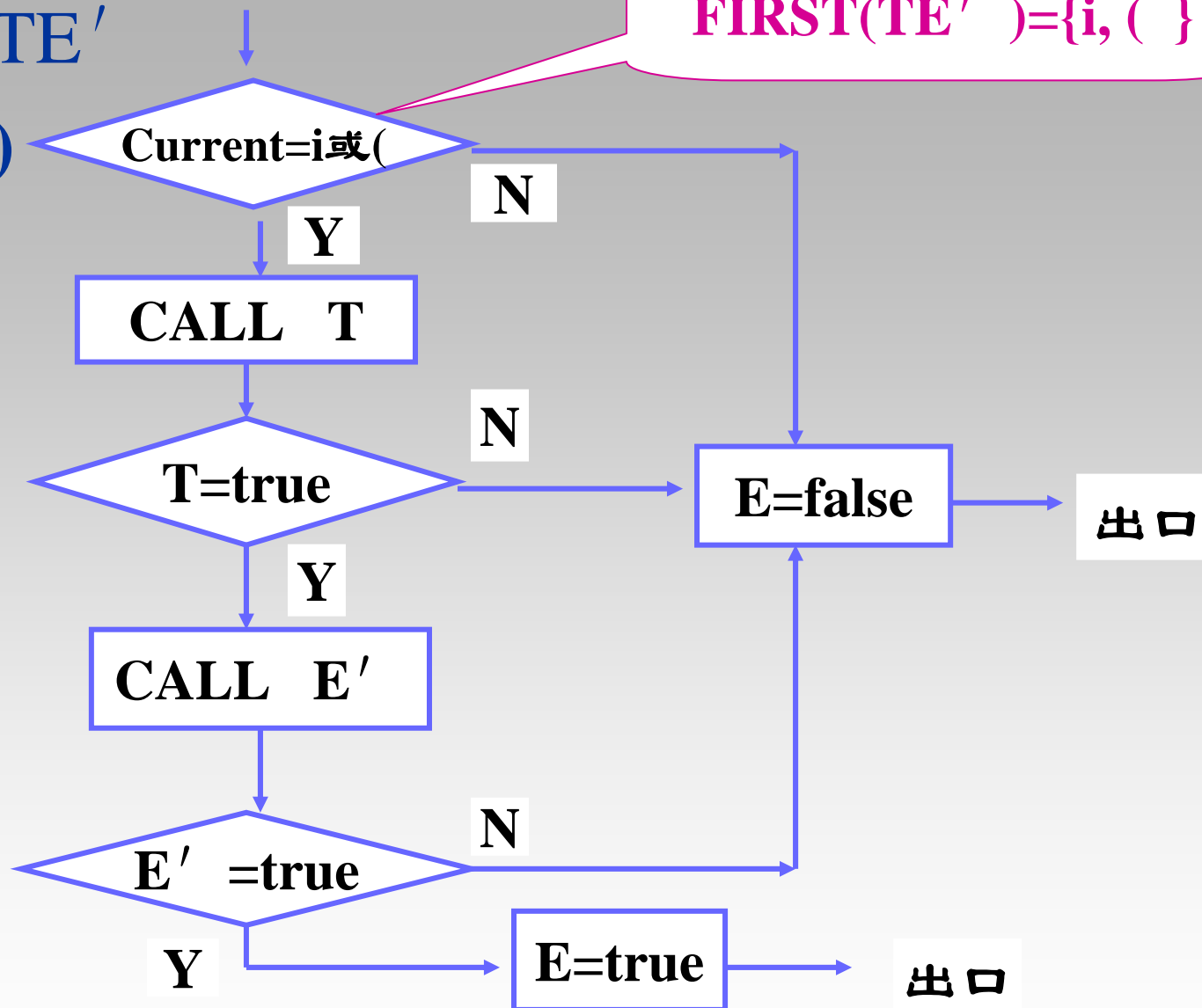
Y

E=false

出口

E=true

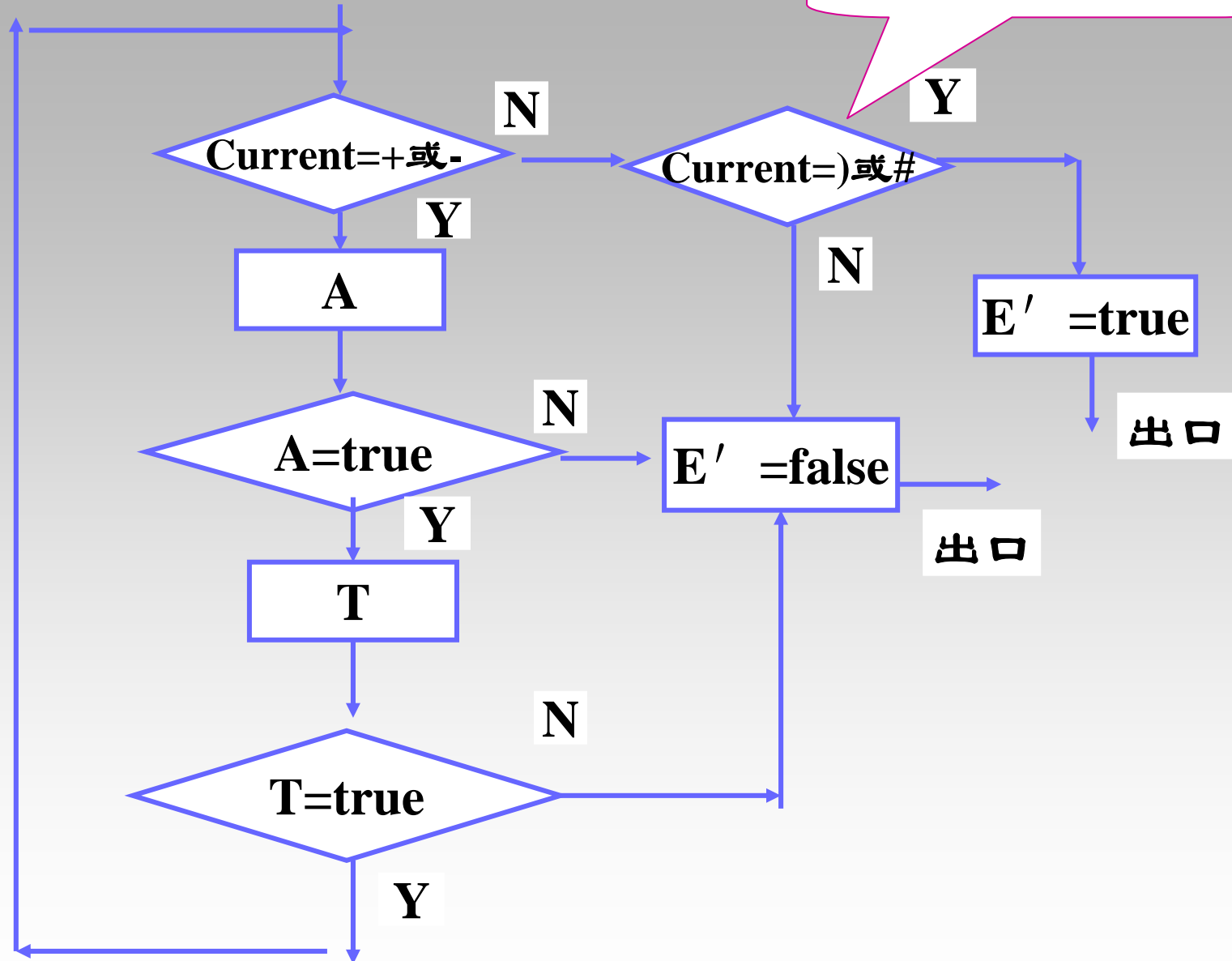
出口



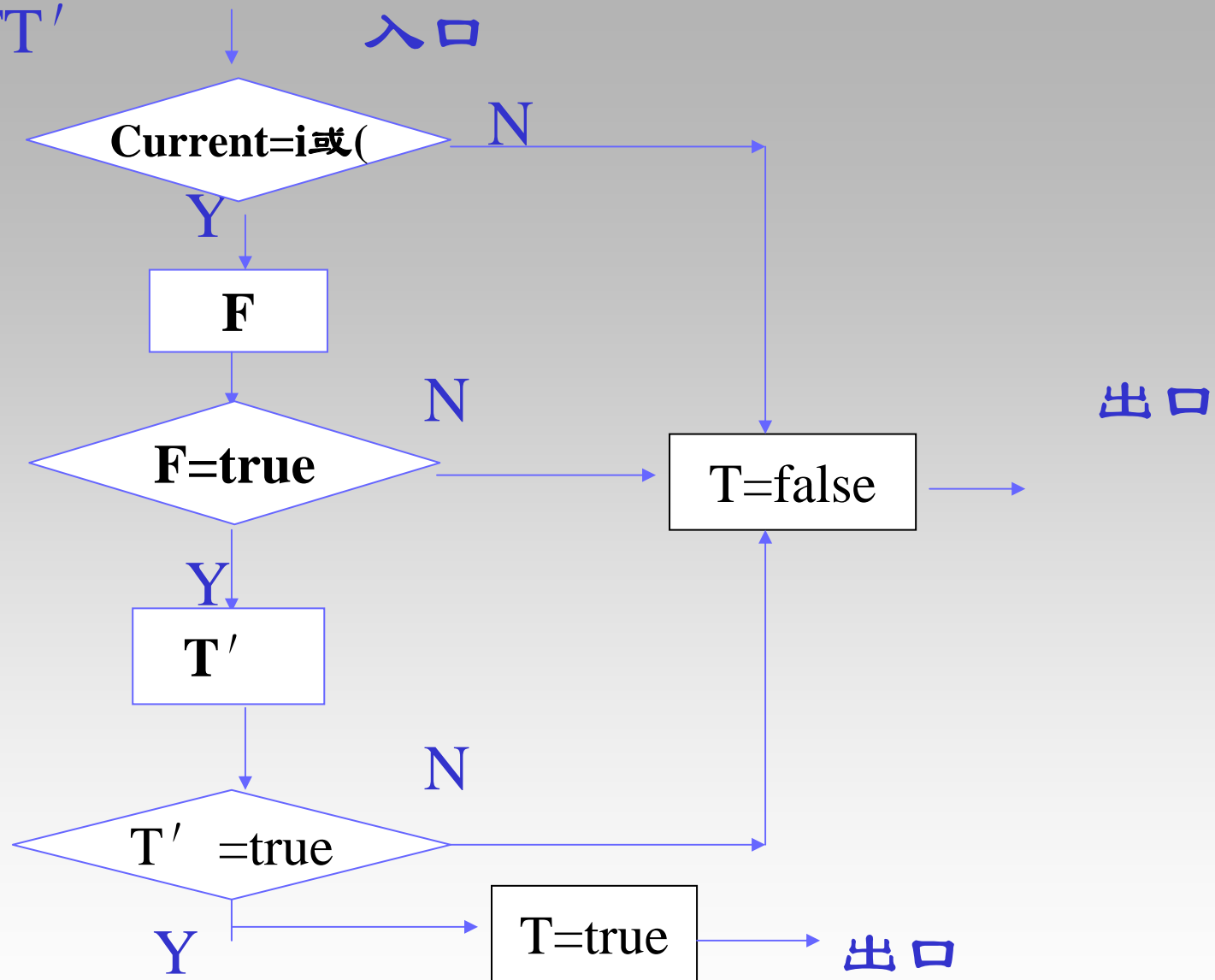
$E' \rightarrow ATE' \mid \varepsilon$

$F(E')$

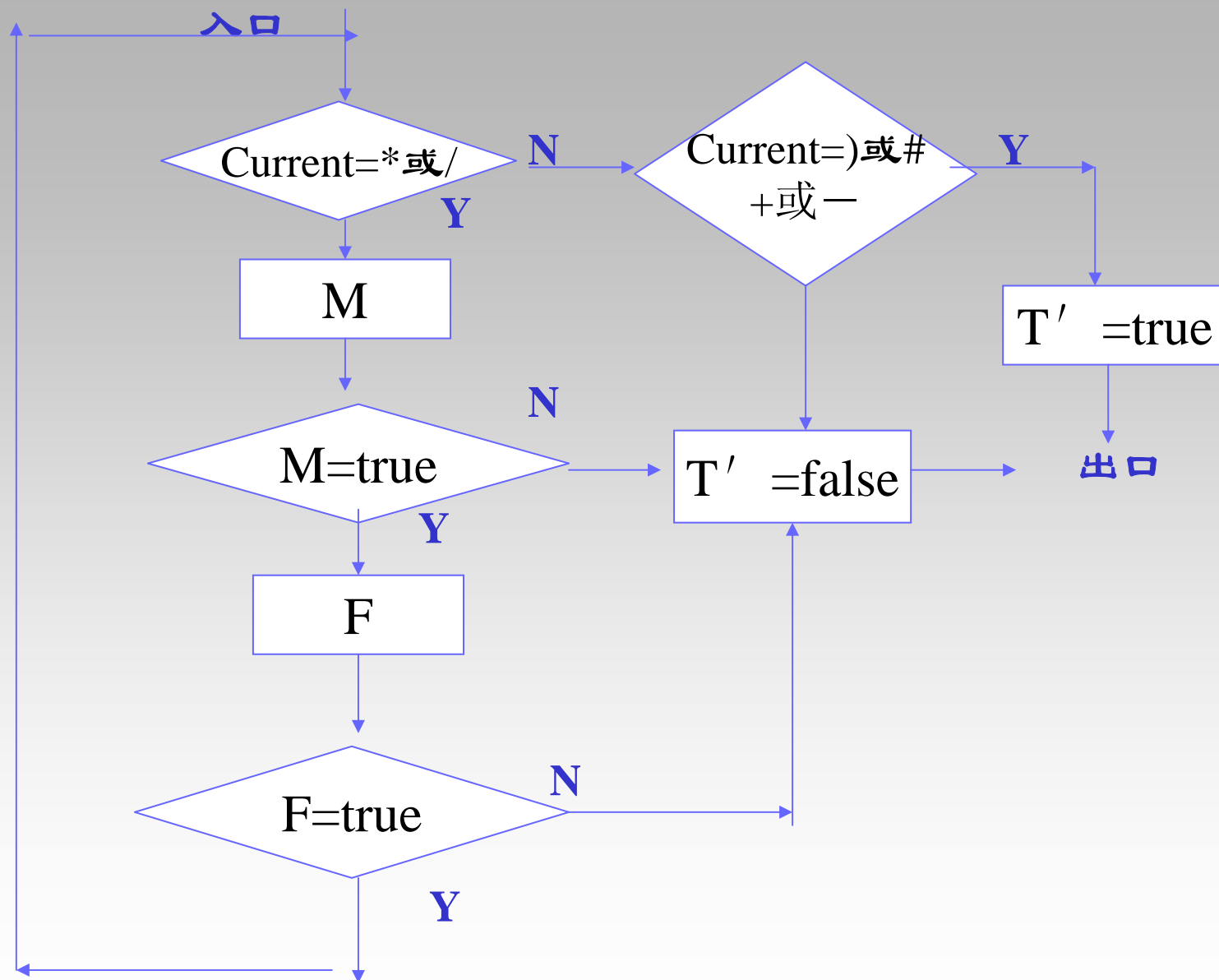
$FOLLOW(E') = \{), \# \}$



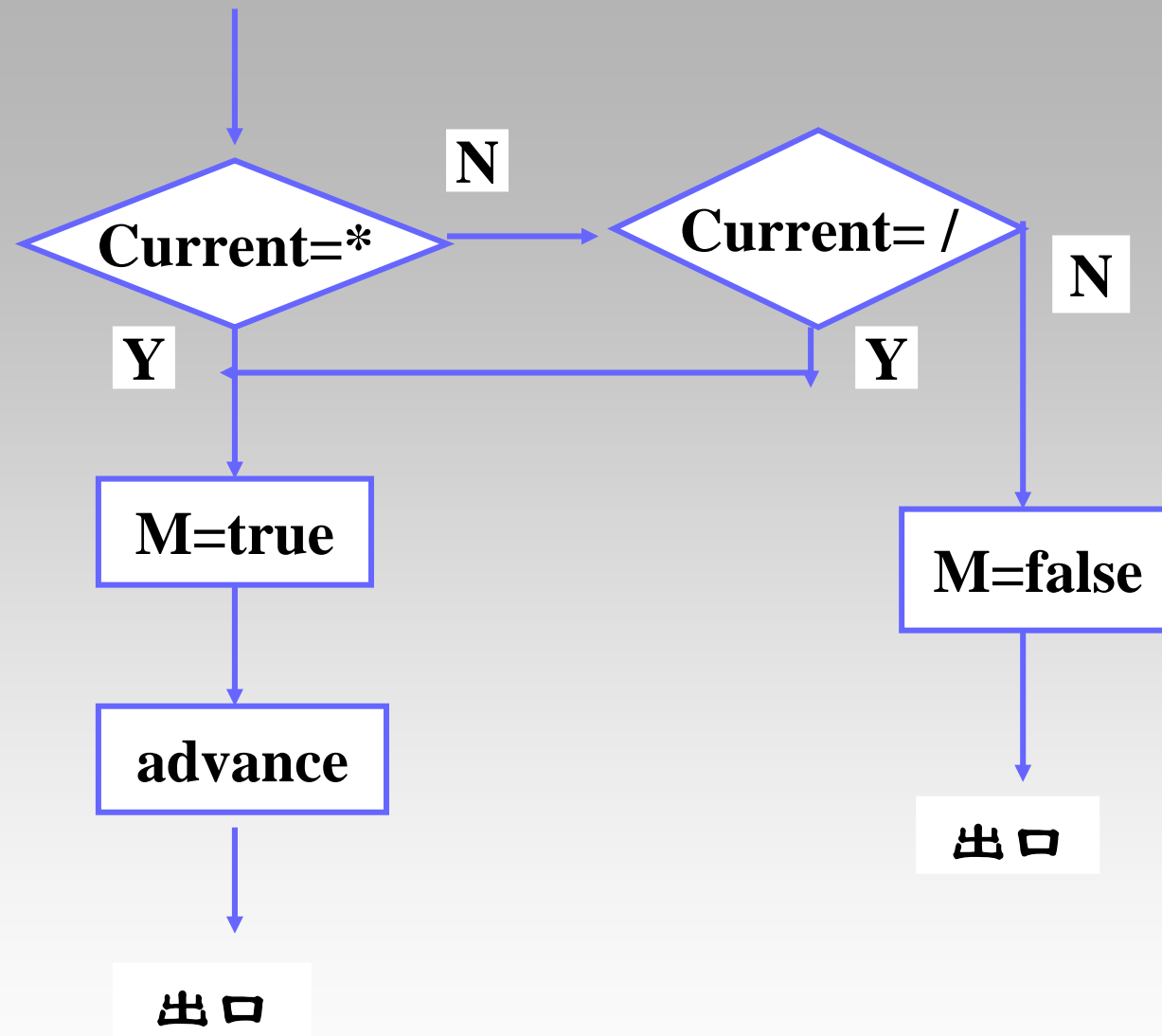
$T \rightarrow FT'$



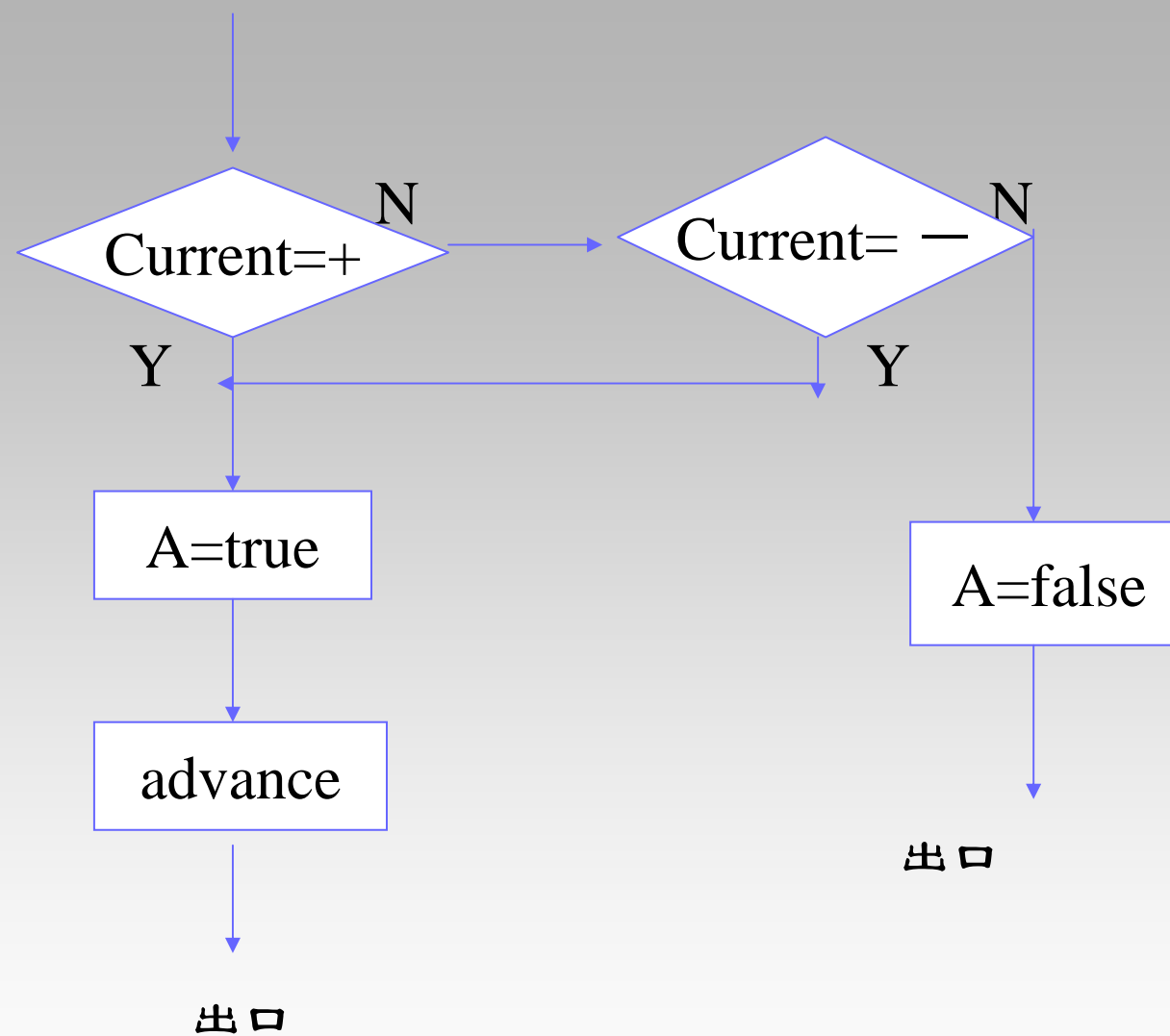
$T' \rightarrow MFT' \mid \varepsilon$



$M \rightarrow * \mid /$
 $F(M)$

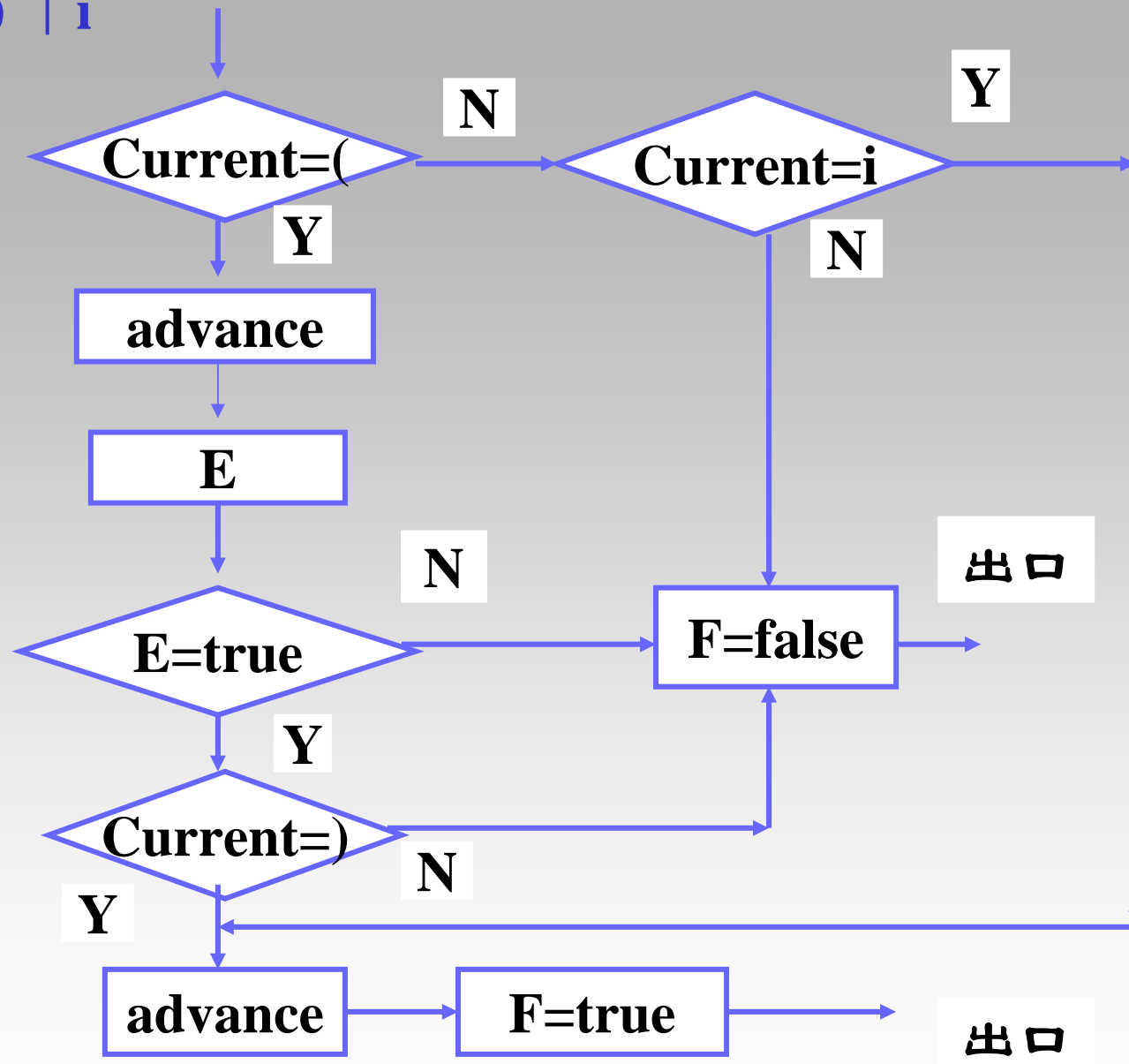


$A \rightarrow + \mid -$



$F \rightarrow (E) \mid i$

$F(F)$



2、LL (1)分析法

1:只向前看一个输入符号便能确定
当前应选择的规则

L:产生一个最左推导 (leftmost)

L:自左 (left) 向右扫描源程序

(1) LL (1) 分析器的描述

① 逻辑结构:

一张分析表M: 包含文法的全部信息

一分析栈: 用于存放分析过程中的文法符号

总控程序: 控制分析过程 (不同的文法可用一个)

M[A,a]

	a
A	M[A,a]
.....

A:处于分析栈中

a: 处于输入串中

M[A,a]:分析栈中面临输入符号a时应采取的动作

a_1	a_2			a_i	...	a_n	#
-------	-------	-------	--	--	-------	-----	-------	---

将#号放在输入串的尾部

②LL(1)分析器



分析栈

#	S	X_{m-1}	X_m
---	---	-------	-----------	-------	------

输入串 $a_i a_{i+1} \dots a_n \#$

$X_m \in V_n$ 且 $M[X_m, a_i]$ 为 $X_m \rightarrow UVW$

则分析栈为

#	S	X_{m-1}	W	V	U
---	---	-------	-----------	---	---	---	------

(2)LL(1)分析过程

①初始格局: #, S依次入栈, #置输入串尾

分析栈

输入: $a_1a_2\dots a_n\#$

#	S		
---	---	-------	--	--

②反复执行,任何时候按栈顶 X_m 和输入 a_i 依据分析表,执行下述三个动作之一

■若 $X_m \in V_n$

若 $M[X_m, a_i]$ 对应一产生式 则 X_m 退栈,

产生式右部符号按反序进栈

(相当于进行一步推导)右部为 ε ,不进栈

若 $M[X_m, a_i]$ 为error:出错处理

■ 若 $X_m \in V_T$

① $X_m = a_i \neq \#$ 一步匹配,

则 X_m 出栈, 输入符号指针指下一位置.

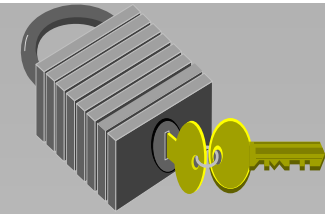
② $X_m \neq a_i$ 调error

■ 若 $X_m = \#$

① $X_m = a_i = \#$ 分析成功, 结束

② $X_m \neq a_i$, 调error

分析算法



BEGIN

把‘#’，文法开始符号依次入栈；把第一个输入符号读进a；

FLAG:=TRUE；

WHILE FLAG DO

BEGIN

把栈顶符号出栈并放在X中；

IF $X \in V_T$ THEN IF $X=a$ THEN

把下一个输入符号读进a

ELSE ERROR

ELSE IF $X=\#$ THEN

IF $X=a$ THEN FLAG:=FALSE

ELSE ERROR

ELSE IF $M[X,a]=\{X \rightarrow X_1X_2..X_K\}$

THEN 把 $X_K, X_{K-1},...,X_1$ 依次入栈

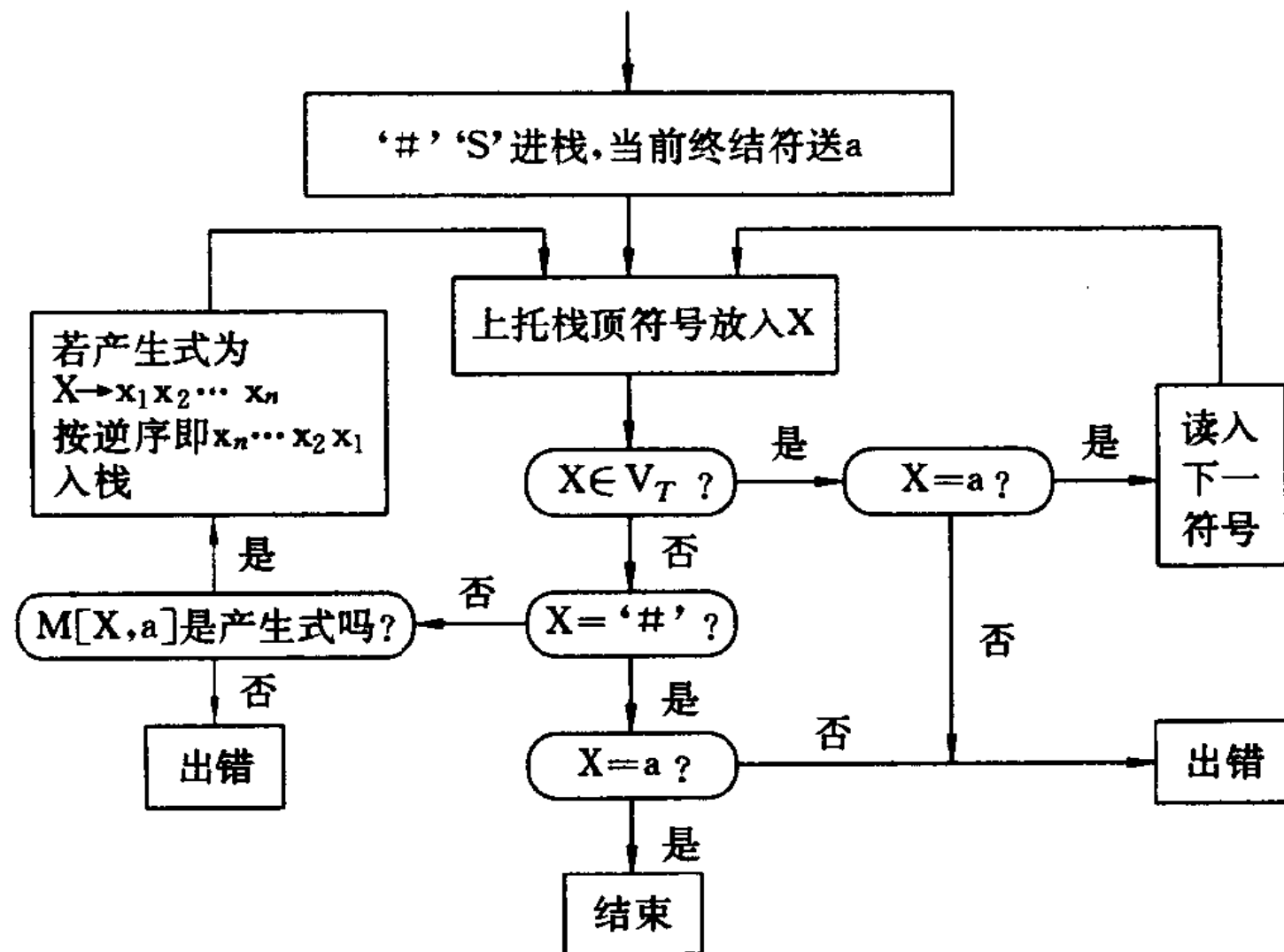
ELSE ERROR

END OF WHILE;

STOP/*分析成功，过程完毕*/

END

LL(1)分析（预测分析）程序框图



G[E]:

$E \rightarrow TE'$

$E' \rightarrow ATE' \mid \varepsilon$

$T \rightarrow FT'$

$T' \rightarrow MFT' \mid \varepsilon$

$F \rightarrow (E) \mid i$

$A \rightarrow + \mid -$

$M \rightarrow * \mid /$

问题：试用LL (1) 分析法分析输入串 $i+i*i$ 是否是文法的句子。

LL(1)分析表

	i	+	—	*	/	()	#
E	$E \rightarrow TE'$					$E \rightarrow TE$,		
E'		$E' \rightarrow ATE'$	$E' \rightarrow ATE'$				$E' \rightarrow \epsilon$	$E' \rightarrow \epsilon$
T	$T \rightarrow FT'$					$T \rightarrow FT$,		
T'		$T' \rightarrow \epsilon$	$T' \rightarrow \epsilon$	$T' \rightarrow MFT'$	$T' \rightarrow MFT'$		$T' \rightarrow \epsilon$	$T' \rightarrow \epsilon$
F	$F \rightarrow i$					$F \rightarrow (E)$		
A		$A \rightarrow +$	$A \rightarrow -$					
M				$M \rightarrow *$	$M \rightarrow /$			

例:输入串 $i+i*i$ 的分析过程 (查LL(1)分析表)

分析栈

#	E	
---	---	--

#	E'	T	
---	----	---	--

#	E'	T'	F	
---	----	----	---	--

#	E'	T'	i	
---	----	----	---	--

#	E'	T'	
---	----	----	--

#	E'	
---	----	--

余留输入串

$i+i*i\#$

$i+i*i\#$

$i+i*i\#$

$i+i*i\#$

$+i*i\#$

$+i*i\#$

分析表中产生式

$M[E,i] \quad E \rightarrow TE'$

$M[T,i] \quad T \rightarrow FT'$

$M[F,i] \quad F \rightarrow i$

Pop , Nextsym

$M[T',+] \quad T' \rightarrow \varepsilon$

$M[E',+] \quad E' \rightarrow ATE'$

分析栈

#	E	T	A		
,					
#	E	T	+		
,					
#	E	T			
,					
#	E	T	F		
,					
#	E	T	i		
,					
#	E	T			
,					
#	E	T	F	M	
,					

余留输入串

+i*i#
+i*i#
i*i#
i*i#
i*i#
*i#
*i#

分析表中产生式

$A \rightarrow +$
Pop , Nextsym
$T \rightarrow FT'$
$F \rightarrow i$
Pop , Nextsym
$T' \rightarrow MFT'$
$M \rightarrow *$

分析栈

#	E	T	F	*	
	,	,			
#	E	T	F		
	,	,			
#	E	T	i		
	,	,			
#	E	T			
	,	,			
#	E				
	,				
#					

余留输入串

*i #
i#
i#
#
#
#

分析表中产生式

Pop , Nextsym
$F \rightarrow i$
Pop , Nextsym
$T' \rightarrow \varepsilon$
$E' \rightarrow \varepsilon$
成功

结论：i+i*i是文法的合法句子

(3)LL(1)分析表的构造

两个集合

$\text{FIRST}(\alpha_i) =$

$\{\mathbf{a}_i \mid \alpha_i \xRightarrow{*} \mathbf{a}_i \delta, \text{且 } \mathbf{a}_i \in V_t, \delta \in V^*\}$

若 $\alpha_i \xRightarrow{*} \varepsilon$, 则 $\varepsilon \in \text{FIRST}(\alpha_i)$

$\text{FOLLOW}(A) =$

$\{\mathbf{a} \mid S \xRightarrow{*} \alpha A \mathbf{a} \delta, \text{且 } \mathbf{a} \in V_t, \alpha, \delta \in V^*\}$

若 $S \xRightarrow{*} \alpha A a$, 则 $\# \in \text{FOLLOW}(A)$

■构造FIRST的算法

(-)对 $G[S]$, $x \in V_n \cup V_t$, 计算 $\text{FIRST}(x)$

① 若 $x \in V_t$

则 $\text{FIRST}(x) = \{x\}$

② 若 $x \in V_n$

有 $x \rightarrow a\alpha$, ($a \in V_t$)或/和 $x \rightarrow \varepsilon$

则 a 或/和 $\varepsilon \in \text{FIRST}(x)$

③对 $x \rightarrow Y_1 Y_2 \dots Y_k$ (且 $Y_1 \in V_n$), 反复使用以下直到每一个 $\text{FIRST}(x)$ 不再增大为止.

i 若 $Y_1 \in V_n$

则把 $\text{FIRST}(Y_1) - \{ \varepsilon \}$ 元素加入 $\text{FIRST}(x)$ 中

ii 若 $Y_1, Y_2, \dots, Y_{i-1} \in V_n$ $(2 \leq i \leq k)$

且 $\varepsilon \in \text{FIRST}(Y_j)$ $(1 \leq j \leq i-1)$

则把 $\text{FIRST}(Y_i) - \{ \varepsilon \}$ 元素加入 $\text{FIRST}(x)$ 中

iii 若 $Y_1, Y_2, \dots, Y_k \in V_n$

且 $\varepsilon \in \text{FIRST}(Y_j)$ $(1 \leq j \leq k)$

则把 ε 元素加入 $\text{FIRST}(x)$ 中

(二) 构造 $\text{FIRST}(\alpha)$ $\alpha = X_1X_2 \dots X_n$
 $X_i \in V, \quad \alpha \in V^*$

① 置 $\text{FIRST}(\alpha) = \{ \quad \}$

② $\text{FIRST}(X_1) - \{ \varepsilon \}$ 加入 $\text{FIRST}(\alpha)$

③ 若 $\varepsilon \in \text{FIRST}(X_1)$,

则 $\text{FIRST}(X_2) - \{ \varepsilon \}$ 加入 $\text{FIRST}(\alpha)$

若 $\varepsilon \in \text{FIRST}(X_1)$ 且 $\varepsilon \in \text{FIRST}(X_2)$

则 $\text{FIRST}(X_3) - \{ \varepsilon \}$ 加入 $\text{FIRST}(\alpha)$

.....以此类推

若 $\varepsilon \in \text{FIRST}(X_i) \quad 1 \leq i \leq n$

则 $\varepsilon \in \text{FIRST}(\alpha)$

■构造FOLLOW(A)的算法 $A, B \in V_n$

①令 $\# \in \text{FOLLOW}(S)$ S 为文法开始符号

②对 $A \rightarrow \alpha B \beta$, 且 $\beta \neq \varepsilon$

则将 $\text{FIRST}(\beta) - \{\varepsilon\}$ 加入 $\text{FOLLOW}(B)$ 中

③反复, 直至每一个 $\text{FOLLOW}(A)$ 不再增大

对 $A \rightarrow \alpha B$ 或 $A \rightarrow \alpha B \beta$ (且 $\varepsilon \in \text{FIRST}(\beta)$)

则 $\text{FOLLOW}(A)$ 中的全部元素加入 $\text{FOLLOW}(B)$

■构造分析表的算法

由每一个产生式 $A \rightarrow \alpha_1 \mid \alpha_2 \mid \dots \mid \alpha_n$

确定 $M[A, a]$ 矩阵 $a \in V_t$

① 任何 $a \in \text{FIRST}(\alpha_i)$, 置 $M[A, a] = \text{"pop, push}(\alpha_i')\text{"}$
 α_i' 为 α_i 倒置 或将 $A \rightarrow \alpha_i$ 规则填入 $M[A, a]$

② 若 $\varepsilon \in \text{FIRST}(\alpha_i)$,

则对于任一个 $b \in \text{FOLLOW}(A)$ $b \in V_t$ 或 #

置 $M[A, b] = \text{"pop"}$ 或将 $A \rightarrow \varepsilon$ 规则填入 $M[A, b]$

➤ 此时 b 不属于 $\text{FIRST}(A)$

③ 其它空白为出错