

第6章 自下而上分析和优先 分析方法

编译原理 陈炬桦

isscjh@mail.sys.edu.cn

- 自下而上分析是**最左推导的逆过程**，从构造语法树的观点来讲，是指从树的末端结点开始，逐步向上归约，直至树的根结点，也就是说，从输入串本身开始，朝着开始符号进行归约。直至到达开始符号为止。
- 确定的自下而上分析器通常分为**两大类**：优先分析器 (Precedence Parser) 和 LR(k) 分析器。
- 本章将主要讨论优先分析器方面的内容。在具体讨论优先分析方法之前，我们先介绍句柄和归约的概念。因为它们在自下而上分析过程中占有重要地位。

6.1 短语与句柄

- 定义6.1 文法 $G[S]$: $S \xRightarrow{*} xUy, U \Rightarrow^+ \alpha$
称 α 是句型 $x \alpha y$ 相对于 U 的短语。

$$S \xRightarrow{*} xUy, U \Rightarrow \alpha$$

称 α 是句型 $x \alpha y$ 相对于 U 的直接短语或简单短语。
最左直接短语称句柄(Handle)。

- 例6.1 设文法 $G_{37}[\text{数}]$:

$\langle \text{数} \rangle \rightarrow \langle \text{数字串} \rangle$

$\langle \text{数字串} \rangle \rightarrow \langle \text{数字串} \rangle \langle \text{数字} \rangle \mid \langle \text{数字} \rangle$

$\langle \text{数字} \rangle \rightarrow 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9$

$w = \langle \text{数字串} \rangle 1$ 是它的一个句型。

我们不能因为有 $\langle \text{数} \rangle \Rightarrow \langle \text{数字串} \rangle$ 就断定 $\langle \text{数字串} \rangle$ 就是 w 的简单短语。因为“ $\langle \text{数} \rangle 1$ ”不是一个句型。事实上， w 含有两个短语：“ $\langle \text{数字串} \rangle 1$ ”和“1”，“1”是简单短语而且是句柄，因为有

- a) $\langle \text{数} \rangle \xRightarrow{*} \langle \text{数字串} \rangle$ 且 $\langle \text{数字串} \rangle \xRightarrow{*} \langle \text{数字串} \rangle 1$
- b) $\langle \text{数} \rangle \xRightarrow{*} \langle \text{数字串} \rangle \langle \text{数字} \rangle$ 且 $\langle \text{数字} \rangle \Rightarrow 1$

- 语法树与短语的关系密切。语法树的子树是由该树的某个结点(称为子树的根)及其所有分枝构成的部分树称为原树的一颗子树。
- 语法树的简单子树是只有单层分枝的子树——这颗子树只有且必须有父子两代，没有第三代，由此，引出下述结论：
 - 每个句型(句子)都对应一颗语法树；
 - 每颗语法树的端末结点自左至右排列构成一个句型(句子)；
 - 每颗子树的端末结自左至右排列构成一短语；
 - 每颗子树的端末结自左至右排列构成一简单短语；
 - 最左简单子树的端末结自左至右排列构成一句柄。

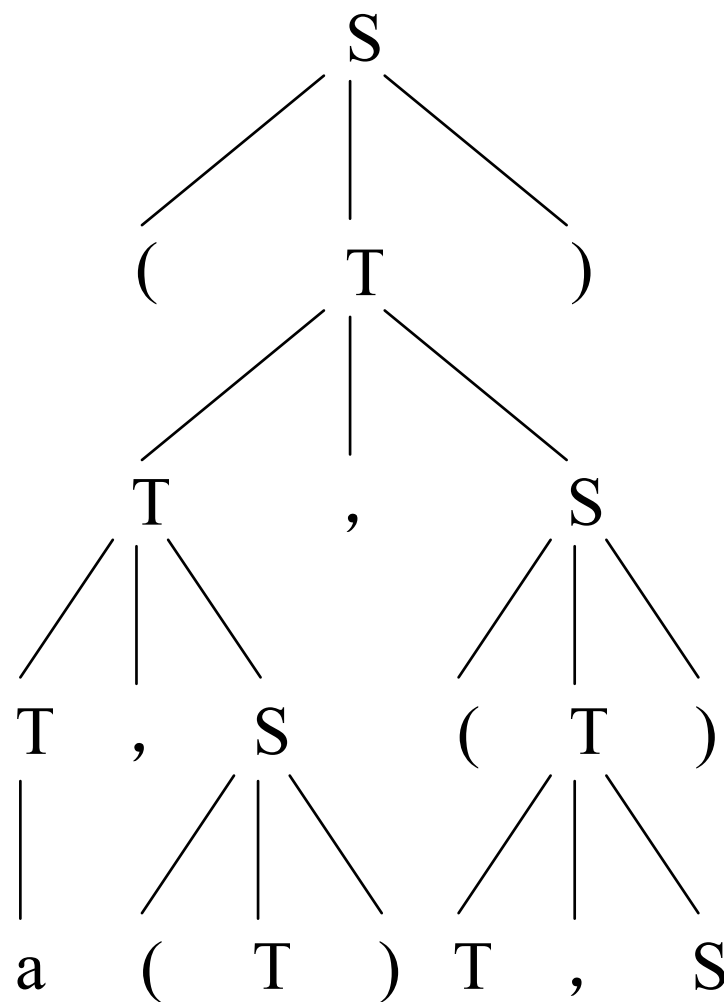
- 例6.3 句型 $\alpha = (a, (T), (T, S))$ 的语法树如右图所示。它有7个分枝结点，分别以这7个分枝结点为根结点，得7棵子树。再分别由这7棵子树的叶子结点（从左到右）组成7个符号串，它们是句型 α 的7个短语。这7个短语是：

- ① $(a, (T), (T, S))$
- ② $a, (T), (T, S)$
- ③ $a, (T)$
- ④ (T, S)
- ⑤ a
- ⑥ (T)
- ⑦ T, S

- 句型 α 的语法树有3棵简单子树。由这3棵简单子树的叶子结点组成的符号串，则是句型 α 的3个简单短语：

- ① a ② (T) ③ T, S

- 句型 α 的语法树的最左简单子树的叶子结点 a 便是句型 α 的句柄。



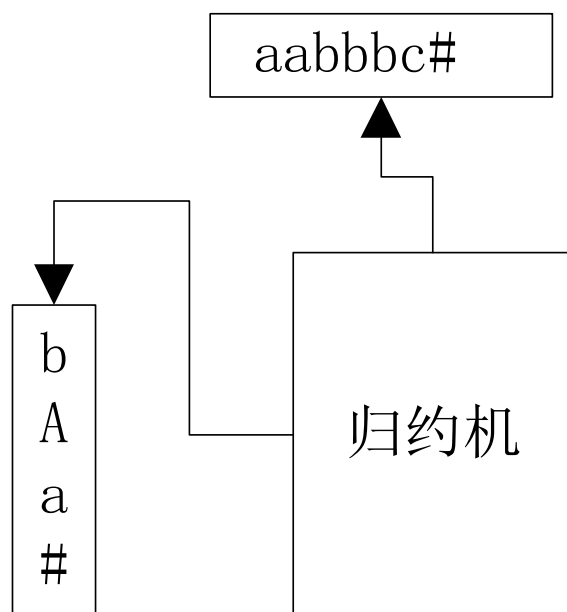
6. 2 移进-归约方法

- 例 6.4 $G_{39}[Z]$

$Z \rightarrow AB$

$A \rightarrow aAb \mid ab$

$B \rightarrow bB \mid c$



步骤	栈符号	输入符号串	操作
1	#	aabbbc#	移进
2	#a	abbbc#	移进
3	#aa	bbbc#	移进
4	#aab	bbc#	归约
5	#aA	bbc#	移进
6	#aAb	bc#	归约
7	#A	bc#	移进
8	#Ab	c#	移进
9	#Abc	#	归约
10	#AbB	#	归约
11	#AB	#	归约
12	#Z	#	OK

6.3 非确定的自下而上分析器

非确定的自下而上分析器在以下三方面不同于非确定的自上而下分析器：

- ① 栈顶在右端，且在任何时刻栈中存有一个句型 $\alpha w \beta$ 的某个左前缀 αw ，此外，在恰好执行一应用动作之前，句柄 w 总出现在栈顶，它将被某个非终结符 A 所替换，其中 $A \rightarrow w$ 是文法中的一个产生式。替换后的结果 αA 又存于栈中。
- ② 该分析器允许一次从栈中逐出零个或多个符号（注意，自上而下的PDA一次决无必要逐出一个以上的符号）。
- ③ 从一个文法构造出的NDPDA将含有两个状态（注意，这样构造的自上而下PDA仅含有一个状态）。多出的这个状态仅用于限定停止条件。

下面给出这种分析器的形式定义：

一个自下而上PDA P 是一个七元组： $(Q, \Sigma, H, \delta, q_0, \$, F)$

其中：

- Q 是状态的有穷集，它的每个元素称为一个状态；
- Σ 是一个有穷输入字母表，它的每个元素是一个输入符号；
- H 是一个有穷栈符号字母表，它的每个元素称为一个栈符号；
- δ 是从 $Q \times H^* \times (\Sigma \cup \{ \varepsilon \})$ 到 $Q \times H^*$ 有穷子集的一个有穷映射；
- $q_0 \in Q$ 是该PDA的初态；
- $\$ \in H$ 是下推栈的初始符号；
- F 是一个终态集(或接收状态集)；它的每个元素称为终态；(可空)。

- 这种PDA的一个构形记为(状态, 栈, 输入串)(注意, 栈顶在右端)。类似的, 一个移动定义为从一个构形到另一个构形的转换:

$$(p, zx, bw) \vdash (q, zy, w)$$

- 其中 $b \in \Sigma \cup \{ \varepsilon \}$, x, y, z 属于 H^* , $w \in \Sigma^*$, 而且 $\delta(p, x, b)$ 包含 (q, y)
- 这样的PDA称为非确定的, 如果某个 $\delta(p, x, b)$ 包含一个以上的对偶, 或者其中的 x 或 b 为空。
- 由 p 所定义的语言 $L(P)$ 记为 $\{w \mid (q_0, Z_0, w) \vdash^* (q, x, \varepsilon), q \in F \& x \in H^*\}$

给定CFG $G=(N, \Sigma, P, S)$ ，用下述方法可以为它构造一个拓广的PDA R 使得 $L(R)=L(G)$ ：

1. R 有两个状态 q 和 r ；
2. R 得输入字母表是 Σ ；
3. 栈字母表 H 由 $N \cup (\Sigma \cup \{\$ \})$ 组成；
4. 栈的初始符号是 $\$$ ；
5. 初态为 q ，终态集 $F=\{r\}$ ；
6. 映射函数 δ 定义如下：
 - ① 移进规则 对每个终结符 $b \in \Sigma$ ， $\delta(q, \varepsilon, b)$ 包含 $\{(q, b)\}$ 。这些移动的作用是：将输入串的符号移进栈。
 - ② 归约规则 对 P 中的每个产生式 $A \rightarrow w$ ， $\delta(q, w, \varepsilon)$ 包含 (q, A) ，这些移动的作用是把位于栈顶的句柄 w 归约到相应的非终结符 A 。
 - ③ 停止规则 $\delta(q, \$S, \varepsilon)$ 包含 (r, ε) ，其中 S 是文法 G 的开始符号。这个规则表示分析成功，停止任何动作。注意，它要求栈只含有 $\$S$ 。

例6.5 考虑算术表达式文法G3[E]:

$E \rightarrow E+T$

$E \rightarrow T$

$T \rightarrow T * F$

$T \rightarrow F$

$F \rightarrow (E)$

$F \rightarrow i$

该文法的PDA映射函数 δ 为

$\delta(q, \varepsilon, a) = \{(q, a)\}$ 根据规则6(①)

$\delta(q, \varepsilon, +) = \{(q, +)\}$

$\delta(q, \varepsilon, *) = \{(q, *)\}$

$\delta(q, \varepsilon, '(') = \{(q, '(')\}$

$\delta(q, \varepsilon, ')') = \{(q, ')')\}$

$\delta(q, E+T, \varepsilon) = \{(q, E)\}$ 根据规则6(②)

$\delta(q, T, \varepsilon) = \{(q, E)\}$

$\delta(q, T * F, \varepsilon) = \{(q, T)\}$

$\delta(q, F, \varepsilon) = \{(q, T)\}$

$\delta(q, (E), \varepsilon) = \{(q, F)\}$

$\delta(q, a, \varepsilon) = \{(q, F)\}$

$\delta(q, \$E, \varepsilon) = \{(r, \varepsilon)\}$ 根据规则6(③)

尽管上面这些映射函数都不包含一个以上的对偶，但这个PDA仍是非确定的。其中，前五个转换函数用于把输入符号移进栈。接下来的六个用于完成归约工作，该PDA恰好识别 $L(G)$ 。

考虑输入串 $a*a+a$ ，
只有下面的移动序列才能够到达终止状态：

1 $(q, \$, a*a+a) \vdash$
2 $(q, \$a, *a+a) \vdash$
3 $(q, \$F, *a+a) \vdash$
4 $(q, \$T, *a+a) \vdash$
5 $(q, \$T*, a+a) \vdash$
6 $(q, \$T*a, +a) \vdash$

7 $(q, \$T*F, +a) \vdash$
8 $(q, \$T, +a) \vdash$
9 $(q, \$E, +a) \vdash$
10 $(q, \$E+, a) \vdash$
11 $(q, \$E+a, \epsilon) \vdash$
12 $(q, \$E+F, \epsilon) \vdash$
13 $(q, \$E+T, \epsilon) \vdash$
14 $(q, \$E, \epsilon) \vdash$
15 $(r, \epsilon, \epsilon) \vdash$

- 现在，我们考虑第7个构形 $(q, \$T*F, +a)$ 。对它也可应用移进规则，产生 $(q, \$T*F+, a)$ 。但是如果这样做，就没有规则用来归约位于栈顶的 ‘+’。
- 因此，我们必须再次应用移进规则，得到 $(q, \$T*F+a, \varepsilon)$ 。此时，可将 a 归约到 F ，产生 $(q, \$T*F+F, \varepsilon)$ ，然后，再将 F 归约为 T ， T 归约为 E ，得到 $(q, \$T*F+E, \varepsilon)$ 。
- 至此我们不得不停止分析工作，因为我们不可能进行进一步的移进或归约了，而且，现在还没有到达允许停止的构形 $(q, \$E, \varepsilon)$ ，这里也不存在任何其它的移动。因此在第7个构形中应用归约规则而不采用移进，其主要原因是此时 $T*F$ 已是一个句柄。
- 事实上，一个规范归约的过程就是在自左至右扫描输入串时，一旦发现栈顶符号串已构成一个句柄，就立即去归约它的过程。

把用上面方法构造的PDA说成是非确定的是因为：

- ① 只要输入串还有输入符号，总可以应用移进规则；
- ② 虽然仅当栈顶的符号串与某个产生式的右部匹配时才可应用归约规则，但是，可能存在若干不同的产生式，它们的右部都于栈顶的符号串相匹配。因此，若栈顶符号串为 αw ，那么由归约规则，任何产生式 $A \rightarrow w$ 的左部都可用来替代 w 从而得到 αA 。注意，这个PDA并没有定义 w 的长度；
- ③ 作为上一点的特例，在任何时刻，都可以选用空产生式 $A \rightarrow \varepsilon$ ，因为它根本不需要与之匹配的栈顶符号。与这种产生式相关的映射函数是包含 (q, A) 的 $\delta(q, \varepsilon, \varepsilon)$ ，即，仅下推非终结符 A 进栈。

- 自下而上分析法是一种“移进 - 归约”法，
优先分析程序和LR(k)分析器是最常用的两类
确定的自下而上分析器。
- 它们各自又有许多不同的变种，其中有些在
功能上有所增强，有些则在“体积”上有所
减少。
- 对于所谓的“可进行归约的符号串”，在不
同的自下而上分析方法中有不同的称呼，在
简单优先分析方法中称之为“句柄”，而在
算符优先分析方法中称之为“素短语”。

6. 4 有关文法的一些关系

- 关系：一个 n 元关系 R 定义为一个有序的 n 元组集合。

$x_1, x_2, x_3, \dots, x_n$ 满足关系 R ，当且仅当有序 n 元组

$$(x_1, x_2, x_3, \dots, x_n) \in R$$

- 关系具有以下基本性质：

(1) 自反性：任意 $x \in \Sigma$ ，有 xRx

(2) 对称性：任意 $x, y \in \Sigma$ ，任意 xRy ，有 yRx

(3) 传递性：任意 $x, y, z \in \Sigma$ ，有 xRy ， yRz 则有 xRz

与文法有关的一些关系:

(1) 关系和 $x, y \in \Sigma$, R_1 和 R_2 是 Σ 上的两个关系, R_1 与 R_2 的两个关系和记为 R_1+R_2 。

$x(R_1+R_2)y$ 当且仅当 xR_1y 或 xR_2y

(2) 关系积 R_1 和 R_2 是 Σ 上的两个关系, R_1 与 R_2 的两个关系积记为 R_1R_2 。

xR_1R_2y 当且仅当 存在 $w \in \Sigma$, 使得 xR_1w 与 wR_2y

(3) 传递闭包 关系 R 的传递闭包记为 R^+ , $x, y \in \Sigma$,

xR^+y 当且仅当存在 $n>0$, 使得 xR^ny

(4) 自反传递闭包 关系 R 的传递闭包记为 R^* , $x, y \in \Sigma$,

xR^*y 当且仅当 存在 $n \geq 0$, 使得 xR^ny , 其中 R^0 为恒等

布尔矩阵和关系

- 关系可用集合定义，也可用布尔矩阵表示
 xRy 当且仅当 $M[x,y]=1$

定理6.1 $M(R^T)=M(R)^T$

定理6.2 $M(R_1+R_2)=M(R_1)+M(R_2)$

定理6.3 $M(R_1R_2)=M(R_1)M(R_2)$

定理6.4 $M(R^+)=M(R)^+$

传递闭包的Warshall算法

for j:=1 until N do

for i:=1 until N do

if $A[i,j]=1$ then for k:=1 until N do

$A[i,k] := A[i,k] \vee A[j,k]$

$$M(R) = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 \end{bmatrix}$$

$$M(R^+) = \begin{bmatrix} 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 \end{bmatrix}$$

关系FIRST和LAST

$$\textcircled{1} A \text{ FIRST } B \quad A \rightarrow B\alpha$$

$$\textcircled{2} A \text{ LAST } B \quad A \rightarrow \alpha B$$

结论:

$$\textcircled{1} A \text{ FIRST}^+ B \iff A \overset{+}{\Rightarrow} B\alpha$$

$$\textcircled{2} A \text{ LAST}^+ B \iff A \overset{+}{\Rightarrow} \alpha B$$

- 例6.8 设文法 $G_{33}[S]$:

$S \rightarrow (A) \mid a \mid b$

$A \rightarrow B$

$B \rightarrow S \mid Scb$

FIRST

	S	A	B	a	b	c	()
S				1	1		1	
A			1					
B	1							
a								
b								
c								
(
)								

for j:=1 until N do

for i:=1 until N do

if $A[i,j]=1$ then for k:=1 until N do

$A[i,k] := A[i,k] \vee A[j,k]$

FIRST⁺

	S	A	B	a	b	c	()
S				1	1		1	
A	1		1	1	1		1	
B	1			1	1		1	
a								
b								
c								
(
)								

- 例6.8 设文法G33[S]:

$S \rightarrow (A) \mid a \mid b$

$A \rightarrow B$

$B \rightarrow S \mid Scb$

LAST

	S	A	B	a	b	c	()
S				1	1			1
A			1					
B	1				1			
a								
b								
c								
(
)								

for j:=1 until N do

for i:=1 until N do

if $A[i,j]=1$ then for k:=1 until N do

$A[i,k] := A[i,k] \vee A[j,k]$

LAST⁺

	S	A	B	a	b	c	()
S				1	1			1
A	1		1	1	1			1
B	1			1	1			1
a								
b								
c								
(
)								

- 例6.13 $G_{42}[S]$:

$$S \rightarrow (R) \mid a \mid ^$$

$$R \rightarrow T$$

$$T \rightarrow S, T \mid S$$

对符号串 $((a), a)$ 检查

	R	S	T	a	^	,	()
R								=
S						=		>
T								>
a						>		>
^						>		>
,		<	=	<	<		<	
(=	<	<	<	<		<	
)						>		>

步骤	符号栈	关系	输入串	产生式
1	#	<	((a),a)#	
2	#<(<	(a),a)#	
3	#<(<(<	a),a)#	
4	#<(<(<a	>),a)#	$S \rightarrow a$
5	#<(<(<S	>),a)#	$T \rightarrow S$
6	#<(<(<T	>),a)#	$R \rightarrow T$
7	#<(<(<=R	=),a)#	
8	#<(<(<=R=)	>	,a)#	$S \rightarrow (R)$
9	#<(<S	=	,a)#	
10	#<(<S =,	<	a)#	
11	#<(<S =,<a	>)#	$S \rightarrow a$
12	#<(<S =,<S	>)#	$T \rightarrow S$
13	#<(<S =, =T	>)#	$T \rightarrow S, T$
14	#<(<T	>)#	$R \rightarrow T$
15	#<(<=R	=)#	
16	#<(<=R=)	>	#	$S \rightarrow (R)$
17	#<S	>	#	

6. 5 简单优先分析方法

• 简单优先关系 (\equiv \prec \succ)

$$\textcircled{1} L \equiv R \Leftrightarrow U \rightarrow \dots LR \dots$$

$$\textcircled{2} L \prec R \Leftrightarrow U \rightarrow \dots LP \dots, P \overset{+}{\Rightarrow} R \dots$$

$$\textcircled{3} L \succ R \Leftrightarrow U \rightarrow \dots WP \dots, W \overset{+}{\Rightarrow} \dots L; P \overset{*}{\Rightarrow} R \dots$$

• 简单优先关系的形式化构造

$$\text{公式6.1 } (\prec) \equiv (\equiv)(\text{FIRST}^+)$$

$$\text{公式6.2 } (\succ) \equiv (\text{LAST}^+)^T(\equiv)(\text{FIRST}^*)$$

简单优先文法及其分析方法

- 定义 6.2 简单优先文法满足
 - ① 任意两个符号 ($V_N \cup V_T$) 至多存在一种简单优先关系。
 - ② 产生式没有相同的右部。

• 例6.9G34[S]

$S \rightarrow (R) | a | ^\wedge$

$R \rightarrow T$

$T \rightarrow S, T | S$

$S \rightarrow (R)$ 有 $(\equiv R \quad R \equiv)$

$T \rightarrow S, T$ 有 $\xi \equiv$, \equiv

\wedge T

: $S \rightarrow (R)$

$(R \dots \Rightarrow (T \dots \Rightarrow (S \dots \Rightarrow (a \dots$

$\Rightarrow (^ \dots$

$\Rightarrow ((\dots$

$T \rightarrow S, T$

$\dots, T \Rightarrow \dots, S \dots \Rightarrow \dots, a \dots$

$\Rightarrow \dots, ^ \dots$

$\Rightarrow \dots, (\dots$

$\Rightarrow : \dots R) \Rightarrow \dots T) \Rightarrow \dots S)$

$\Rightarrow \dots a)$

$\Rightarrow \dots ^)$

$\Rightarrow \dots))$

	R	S	T	a	^	,	()
R								\equiv
S						\equiv		\Rightarrow
T								\Rightarrow
a						\Rightarrow		\Rightarrow
^						\Rightarrow		\Rightarrow
,		\wedge	\equiv	\wedge	\wedge		\wedge	
(\equiv	\wedge	\wedge	\wedge	\wedge		\wedge	
)						\Rightarrow		\Rightarrow

- 例6.13 $G_{42}[S]:$

$$S \rightarrow (R) \mid a \mid ^$$

$$R \rightarrow T$$

$$T \rightarrow S, T \mid S$$

对符号串 $((a), a)$ 检查

	R	S	T	a	^	,	()
R								=
S						=		>
T								>
a						>		>
^						>		>
,		<	=	<	<		<	
(=	<	<	<	<		<	
)						>		>

步骤	符号栈	关系	输入串	产生式
1	#	<	((a),a)#	
2	#<(<	(a),a)#	
3	#<(<(<	a),a)#	
4	#<(<(<a	>),a)#	$S \rightarrow a$
5	#<(<(<S	>),a)#	$T \rightarrow S$
6	#<(<(<T	>),a)#	$R \rightarrow T$
7	#<(<(<=R	=),a)#	
8	#<(<(<=R=)	>	,a)#	$S \rightarrow (R)$
9	#<(<S	=	,a)#	
10	#<(<S=,	<	a)#	
11	#<(<S=,<a	>)#	$S \rightarrow a$
12	#<(<S=,<S	>)#	$T \rightarrow S$
13	#<(<S=,=T	>)#	$T \rightarrow S, T$
14	#<(<T	>)#	$R \rightarrow T$
15	#<(<=R	=)#	
16	#<(<=R=)	>	#	$S \rightarrow (R)$
17	#<S	>	#	

简单优先关系的形式化构造

公式6.1 (\prec) \equiv (\preceq)(FIRST⁺)

$$L \preceq P \Rightarrow L \preceq P, P \text{ FIRST}^+ R$$

公式6.2 (\succ) \equiv (LAST⁺)^T(\preceq)(FIRST^{*})

$$\begin{aligned} W \preceq P &\Rightarrow (W \text{ LAST}^+ L), W \preceq P, P \text{ FIRST}^* R \\ &\Rightarrow L(\text{LAST}^+)^T W, W \preceq P, P \text{ FIRST}^* R \end{aligned}$$

简单优先关系的局限性

- G35[E]

$$E \rightarrow E+T \mid T$$

$$T \rightarrow T*F \mid F$$

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

$$E \rightarrow E+T \qquad + \preceq T$$

$$E+T \Rightarrow E+T*F \qquad + \prec T$$

6. 6 算符优先分析方法

- 算符优先文法 $\equiv \prec \succ$

① $a \equiv b \equiv U \rightarrow \dots ab \dots$ 或 $U \rightarrow \dots aVb \dots$

② $a \prec b \equiv U \rightarrow \dots aP \dots, P \xRightarrow{+} b \dots$ 或 $P \xRightarrow{+} Vb \dots$

③ $a \succ b \equiv U \rightarrow \dots Wb \dots, W \xRightarrow{+} \dots a$ 或 $W \xRightarrow{+} \dots aV$

- 定义6.3 产生式的右部不存在相连的 V_N 符号称算符文法（OG）。
- 定义6.4 任意两 V_T 符号间最多存在一种优先关系称算符优先文法（OPG）

OPG优先关系的构造

(1) 由定义构造算符优先关系

- $\text{FIRST}_{\text{vt}}(U)^+ = \{b \mid U \xRightarrow{+} b\dots \text{ 或 } U \xRightarrow{+} Vb\dots, \\ b \in V_T, V \in V_N\}$
 - $\text{LAST}_{\text{vt}}(U)^+ = \{a \mid U \xRightarrow{+} \dots a \text{ 或 } U \xRightarrow{+} \dots aV, \\ a \in V_T, V \in V_N\}$
- ① $a \equiv b$ 由产生式得到 $U \rightarrow \dots ab\dots$ 或 $U \rightarrow \dots aVb\dots$
 - ② $a \lessdot b$ 由产生式 $U \rightarrow \dots aP\dots$, $b \in \text{FIRST}_{\text{vt}}(P)^+$
 - ③ $a \gtrdot b$ 由产生式 $U \rightarrow \dots Wb\dots$, $a \in \text{LAST}_{\text{vt}}(W)^+$

OPG优先关系的构造

(2) 由直观方法构造算符优先关系

- 先乘除、后加減、指数优先
- 先左后右（加減）、先右后左（指数）
- 括号内优先，运算符小于其它 V_T 符号。

OPG优先关系的构造

(3) 由已知关系构造算符优先关系

$$\textcircled{1} \quad U \text{ FIRST}_{vt} b \Leftrightarrow U \rightarrow b... \text{ 或 } U \rightarrow Vb...$$

$$\textcircled{2} \quad U \text{ LAST}_{vt} a \Leftrightarrow U \xrightarrow{+} ...a \text{ 或 } U \xrightarrow{+} ...aV$$

$$\text{FIRST}_{vt}(U)^+ = \{b \mid U \xrightarrow{+} b... \text{ 或 } U \xrightarrow{+} Vb..., b \in V_T, V \in V_N\}$$

$$\text{LAST}_{vt}(U)^+ = \{a \mid U \Rightarrow ...a \text{ 或 } U \Rightarrow ...aV, a \in V_T, V \in V_N\}$$

• 结论:

$$(\prec) \equiv (\equiv) (\text{FIRST}^*)(\text{FIRST}_{vt})$$

$$(\succ) \equiv (\text{LAST}^*)(\text{LAST}_{vt})^T (\equiv)$$

$G_{44}[E]$

$E \rightarrow E+T \mid T$

$T \rightarrow T*F \mid F$

$F \rightarrow (E) \mid i$

$(\underline{=}) =$

	E	T	F	$+$	$*$	$($	$)$	i
E	0	0	0	1	0	0	1	0
T	0	0	0	0	1	0	0	0
F	0	0	0	0	0	0	0	0
$+$	0	1	0	0	0	0	0	0
$*$	0	0	1	0	0	0	0	0
$($	1	0	0	0	0	0	0	0
$)$	0	0	0	0	0	0	0	0
i	0	0	0	0	0	0	0	0

First=

	E	T	F	$+$	$*$	$($	$)$	i
E	1	1	0	0	0	0	0	0
T	0	1	1	0	0	0	0	0
F	0	0	0	0	0	1	0	1
$+$	0	0	0	0	0	0	0	0
$*$	0	0	0	0	0	0	0	0
$($	0	0	0	0	0	0	0	0
$)$	0	0	0	0	0	0	0	0
i	0	0	0	0	0	0	0	0

First*=

	E	T	F	$+$	$*$	$($	$)$	i
E	1	1	1	0	0	1	0	1
T	0	1	1	0	0	1	0	1
F	0	0	1	0	0	1	0	1
$+$	0	0	0	1	0	0	0	0
$*$	0	0	0	0	1	0	0	0
$($	0	0	0	0	0	1	0	0
$)$	0	0	0	0	0	0	1	0
i	0	0	0	0	0	0	0	1

First_{vt}=

	E	T	F	$+$	$*$	$($	$)$	i
E	0	0	0	1	0	0	0	0
T	0	0	0	0	1	0	0	0
F	0	0	0	0	0	1	0	1
$+$	0	0	0	0	0	0	0	0
$*$	0	0	0	0	0	0	0	0
$($	0	0	0	0	0	0	0	0
$)$	0	0	0	0	0	0	0	0
i	0	0	0	0	0	0	0	0

(=).First *=

$$\begin{bmatrix}
 & E & T & F & + & * & (&) & i \\
 E & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\
 T & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
 F & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 + & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
 * & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
 (& 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
) & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 i & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
 \end{bmatrix}
 \times
 \begin{bmatrix}
 & E & T & F & + & * & (&) & i \\
 E & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 1 \\
 T & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 \\
 F & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 \\
 + & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
 * & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
 (& 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
) & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
 i & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
 \end{bmatrix}
 =
 \begin{bmatrix}
 & E & T & F & + & * & (&) & i \\
 E & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\
 T & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
 F & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 + & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 \\
 * & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 \\
 (& 1 & 1 & 1 & 0 & 0 & 1 & 0 & 1 \\
) & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 i & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
 \end{bmatrix}$$

$$(\equiv) \text{First}^* \text{First}_{vt} =$$

$$\begin{bmatrix}
 & E & T & F & + & * & (&) & i \\
 E & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\
 T & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
 F & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 + & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 \\
 * & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 \\
 (& 1 & 1 & 1 & 0 & 0 & 1 & 0 & 1 \\
) & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 i & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
 \end{bmatrix}
 \times
 \begin{bmatrix}
 & E & T & F & + & * & (&) & i \\
 E & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
 T & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
 F & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\
 + & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 * & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 (& 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
) & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 i & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
 \end{bmatrix}
 =
 \begin{bmatrix}
 & E & T & F & + & * & (&) & i \\
 E & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 T & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 F & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 + & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 \\
 * & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\
 (& 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 \\
) & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 i & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
 \end{bmatrix}$$

	+	*	()	i
+					
*					
(
)					
i					

$G_{44}[E]$

$E \rightarrow E+T \mid T$

$T \rightarrow T*F \mid F$

$F \rightarrow (E) \mid i$

$G_{44}[E]$

$E \rightarrow E+T \mid T$

$T \rightarrow T * F \mid F$

$F \rightarrow (E) \mid i$

$(\underline{=}) =$

$$\begin{bmatrix} & E & T & F & + & * & (&) & i \\ E & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ T & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ F & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ + & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ * & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ (& 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\) & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ i & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$\text{Last} =$

$$\begin{bmatrix} & E & T & F & + & * & (&) & i \\ E & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ T & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ F & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ + & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ * & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ (& 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\) & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ i & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$\text{Last}_{vt} =$

$$\begin{bmatrix} & E & T & F & + & * & (&) & i \\ E & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ T & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ F & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ + & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ * & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ (& 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\) & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ i & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$\text{Last}^* =$






















$$\begin{bmatrix} & E & T & F & + & * & (&) & i \\ E & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ T & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ F & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 \\ + & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ * & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ (& 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\) & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ i & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

$\text{Last}^* . \text{Last}_{\text{vt}} =$

$$\begin{bmatrix}
 & E & T & F & + & * & (&) & i \\
 E & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\
 T & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\
 F & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 \\
 + & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
 * & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
 (& 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
) & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
 i & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
 \end{bmatrix}
 \times
 \begin{bmatrix}
 & E & T & F & + & * & (&) & i \\
 E & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
 T & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
 F & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\
 + & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 * & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 (& 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
) & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 i & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
 \end{bmatrix}
 =
 \begin{bmatrix}
 & E & T & F & + & * & (&) & i \\
 E & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 \\
 T & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 \\
 F & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\
 + & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 * & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 (& 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
) & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 i & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
 \end{bmatrix}$$

$$(\text{Last}^* \cdot \text{Last}_{\text{vt}})^T \cdot \begin{pmatrix} \text{ } \\ \text{ } \\ \text{ } \end{pmatrix} =$$

$$\begin{bmatrix} & E & T & F & + & * & (&) & i \\ E & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ T & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ F & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ + & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ * & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ (& 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\) & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ i & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \times \begin{bmatrix} & E & T & F & + & * & (&) & i \\ E & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ T & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ F & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ + & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ * & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ (& 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\) & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ i & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} & E & T & F & + & * & (&) & i \\ E & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ T & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ F & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ + & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ * & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 \\ (& 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\) & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 \\ i & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 \end{bmatrix}$$

	+	*	()	i
+					
*					
(				
)					
i					

$G_{44}[E]$

$E \rightarrow E + T \mid T$

$T \rightarrow T * F \mid F$

$F \rightarrow (E) \mid i$

素短语及句型分析

- 定义6.5 素短语：包含 V_T 符号但不包含其它素短语的短语。
- 定理6.5 句型 $[v_i]a_i \dots [v_j]a_j[v_{j+1}]$ 最左素短语满足
$$a_{i-1} \lt a_i \oplus a_{i+1} \dots \oplus a_j \gt a_{j+1}$$

• $G_{44}[E]$

$E \rightarrow E+T \mid T$

$T \rightarrow T*F \mid F$

$F \rightarrow (E) \mid i$

分析符号串 $i*(i+i)$

	+	*	i	()
+	\nearrow	\nwarrow	\nwarrow	\nwarrow	\nearrow
*	\nearrow	\nearrow	\nwarrow	\nwarrow	\nearrow
i	\nearrow	\nearrow			\nearrow
(\nwarrow	\nwarrow	\nwarrow	\nwarrow	\perp
)	\nearrow	\nearrow			\nearrow

步骤	符号栈	关系	输入串	最左素短语
1	#	<	$i*(i+i)\#$	
2	$\#<i$	>	$*(i+i)\#$	i
3	$\#v$	<	$*(i+i)\#$	
4	$\#<v^*$	<	$(i+i)\#$	
5	$\#<v^*<($	<	$i+i)\#$	
6	$\#<v^*<(<i$	>	$+i)\#$	i
7	$\#<v^*<(v$	<	$+i)\#$	
8	$\#<v^*<(<v+$	<	$i)\#$	
9	$\#<v^*<(<v+<i$	>	$)\#$	i
10	$\#<v^*<(<v+v$	>	$)\#$	$v+v$
11	$\#<v^*<(v$	=	$)\#$	
12	$\#<v^*<(=v)$	>	$\#$	(v)
13	$\#<v^*v$	>	$\#$	$v*v$
14	$\#v$	=	$\#$	

6. 7 优先函数及其构造

• 6.7.1 优先函数

定义6.1 对于优先矩阵 M ，如果存在函数 f 、 g 满足以下条件：

- ① 若 $L=R$ ，有 $f(L)=g(R)$
- ② 若 $L<R$ ，有 $f(L)<g(R)$
- ③ 若 $L>R$ ，有 $f(L)>g(R)$

称 f 、 g 为 M 的优先函数。

注： M 可为简单优先矩阵，也可为算符优先矩阵。

6.7.2 Bell 方法

- **Bell方法：有向图构造法。**
 - ① 作两排结点：一排为 f_L ，另一排为 g_R ；
 $L > R$ ，从 L 到 R 连一有向弧；
 $L < R$ ，从 R 到 L 连一有向弧；
 $L = R$ ，从 L 到 R 和从 R 到 L 各连一有向弧；
 - ② 计算各结点能到达的结点数（包括自己）
为该函数点的值。
 - ③ 按定义6.1的条件判断，若不满足则不存在
优先函数。

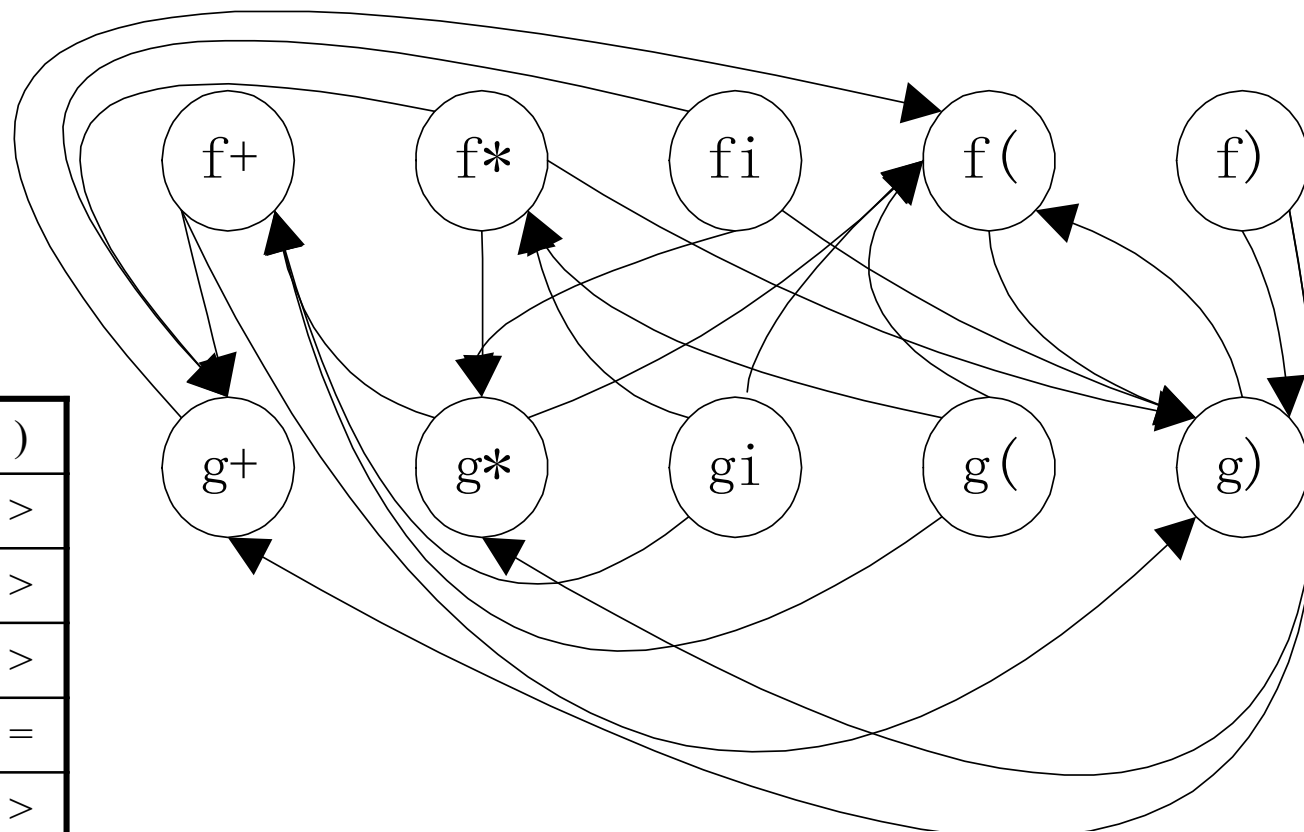
文法G (E) :

$E \rightarrow E+T \mid T$

$T \rightarrow T*F \mid F$

$F \rightarrow (E) \mid i$

	+	*	i	()
+	>	<	<	<	>
*	>	>	<	<	>
i	>	>			>
(<	<	<	<	=
)	>	>			>



	+		*		i		()	
f	4	$f+, g+, f(, g)$	6	$f*, g*, f+, g+, f(, g)$	6	$fi, g*, f+, g+, f(, g)$	2	$f(, g)$	6	$f), g*, f+, g+, f(, g)$
g	3	$g+, f(, g)$	5	$g*, f+, g+, f(, g)$	7	$gi, f*, g*, f+, g+, f(, g)$	7	$g(, f*, g*, f+, g+, (, g)$	2	$f(, g)$

	+	*	i	()
+	>	<	<	<	>
*	>	>	<	<	>
i	>	>			>
(<	<	<	<	=
)	>	>			>

	f+	f*	fī	f(f)	g+	g*	gi	g(g)
f+						1				1
f*						1	1			1
fī						1	1			1
f(1
f)						1	1			1
g+				1						
g*	1			1						
gi	1	1		1						
g(1	1		1						
g)				1						

for j:=1 until N do for i:=1 until N do
 if A[i,j]=1 then for k:=1 until N do A[i,k]:= A[i,k] \vee A[j,k]

		f+	f*	f _i	f(f)	g+	g*	g _i	g(g)
4	f+	1			1		1				1
6	f*	1	1		1		1	1			1
6	f _i	1		1	1		1	1			1
2	f(1						1
6	f)	1			1	1	1	1			1
3	g+				1		1				1
5	g*	1			1		1	1			1
7	g _i	1	1		1		1	1	1		1
7	g(1	1		1		1	1		1	1
2	g)				1						1

		f+	f*	fī	f(f)	g+	g*	gi	g(g)
4	f+	1			1		1				1
6	f*	1	1		1		1	1			1
6	fī	1		1	1		1	1			1
2	f(1						1
6	f)	1			1	1	1	1			1
3	g+				1		1				1
5	g*	1			1		1	1			1
7	gi	1	1		1		1	1	1		1
7	g(1	1		1		1	1		1	1
2	g)				1						1

6.7.3 Floyd方法

• **Floyd方法:**逐次加一法。

- ① $f(A)=g(A)=1$;
- ② if $(L < R)$ and $(f[L] \geq g[R])$ then $g[R] = f[L] + 1$;
- ③ if $(L > R)$ and $(f[L] \leq g[R])$ then $f[L] = g[R] + 1$;
- ④ if $(L = R)$ and $(f[L] \neq g[R])$ then
 $f[L] = g[R] = \max(f[L], g[R])$;
- ⑤ if (②~④有改变) then goto ②;

文法G (E) :

$E \rightarrow E+T \mid T$

$T \rightarrow T * F \mid F$

$F \rightarrow (E) \mid i$

	+	*	i	()
+	>	<	<	<	>
*	>	>	<	<	>
i	>	>			>
(<	<	<	<	=
)	>	>			>

• **Floyd方法:**逐次加一法。

① $f(A)=g(A)=1$;

② if $(L < R)$ and $(f[L] \geq g[R])$ then $g[R]=f[L]+1$;

③ if $(L > R)$ and $(f[L] \leq g[R])$ then $f[L] = g[R]+1$;

④ if $(L=R)$ and $(f[L] \neq g[R])$ then

$f[L]=g[R]=\max(f[L], g[R])$;

⑤ if (②~④有改变) then goto ②;

		+	*	i	()
初值	f	1	1	1	1	1
	g	1	1	1	1	1
第一次迭代	f	1,2	1,2,4	1,2,4	1	1,3,4
	g	1,2	1,3	1,3,5	1,3,5	1
第二次迭代	f	2,3	4,5	4,5	1	4,5
	g	2	3,4	5,6	5,6	1
第三次迭代	f	3	5	5	1	5
	g	2	4	6	6	1

例：设文法

$G[S]: S \rightarrow ABc \mid bc$

$A \rightarrow b$

$B \rightarrow aS \mid S \mid a$

试构造其简单优先矩阵。

• 简单优先关系

① $L \equiv R$ $U \rightarrow \dots LR \dots$ $+$

② $L \prec R$ $U \rightarrow \dots LP \dots, P \Rightarrow R \dots$

③ $L \succ R$ $U \rightarrow \dots WP \dots, W \Rightarrow \dots L; P \overset{+}{\Rightarrow^*} R$

	S	A	B	a	b	c
S						\succ
A	\prec	\prec	\equiv	\prec	\prec	
B						\equiv
a	\equiv	\prec			\prec	\succ
b	\succ	\succ	\succ	\succ	\succ	\equiv
c						\succ

例： 设简单优先矩阵为

	R	S	T	a	^	,	()
R								=
S						=		>
T								>
a						>		>
^						>		>
,		<	=	<	<		<	
(=	<	<	<	<		<	
)						>		>

	R	S	T	a	^	,	()
f	1	12	12	12	12	1	1	12
g	1	12	12	12	12	1	12	1

	R	S	T	a	^	,	()
f	1	2	2	23	23	12	1	12 3
g	1	2	2	23	23	12	23	1

- 用Floyd方法（:逐次加一法）构造其优先函数。

- **Floyd方法:逐次加一法。**

- ① $f(A)=g(A)=1$;
- ② if $(L < R)$ and $(f[L] \geq g[R])$ then $g[R]=f[L]+1$;
- ③ if $(L > R)$ and $(f[L] \leq g[R])$ then $f[L] = g[R]+1$;
- ④ if $(L=R)$ and $(f[L] \neq g[R])$ then
 $f[L]=g[R]= \max(f[L], g[R])$;
- ⑤ if (②~④有改变) then goto ②;

	R	S	T	a	^	,	()
f	1	2	2	3	3	2	1	3
g	1	2	2	3	3	2	3	1