一、一般分析方法

(1) 分析思想:移进--归约 (Shift-Reduce)分

方法？算法？

(2) 解决问题:寻找当前句型的句柄;

(3) 分析方法:利用一个符号栈来记录分析的历史

和指示分析下一步动作.

(4) 动作

移进： 将输入串中的一个符号移进栈里;

归约： 当栈顶呈现句柄时用相应的规则替换;

接受： 宣布分析成功, 此时栈顶只有一开始符号;

出错： 栈顶内容与输入串相悖, 分析无法进行;

问题解答:

(1) 句柄为什么会呈现在栈顶?

最左归约是最右推导的逆过程

B→d

栈外

e#

#	a	A	c	d	

分析栈

(2) 如何才能快速准确地找到当前句型的句柄?

需要通过算法解决,而不能利用语法树.

二、自底向上语法分析法

不同的文法有不同的寻找句柄的方法

- 简单优先分析法 → 简单优先文法

  最左归约，解决算术表达式的分析

- 算符优先分析法 → 算符优先文法

  "最左归约"，解决算术表达式的分析

- LR分析法 → LR(0)文法 SLR(1)文法 LR(1)文法

  最左归约，可分析无二义性文法

针对简单优先文法

解决：简单优先文法？

§4.2.2 简单优先分析

一、简单优先关系

1) 设$G=(V_n, V_t, P, S)$是化简的文法，

$V=V_t \cup V_n$　　S_i, S_j属于V

若G中存在这样的**规范句型**，

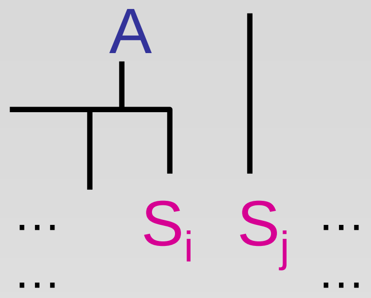
$$\alpha = \cdots S_i S_j \cdots$$

则此相邻的S_i, S_j和 α 的句柄之间的关系必是下述情况之一

(1) 若S_i 在句柄中,而S_j 不在句柄中,

则$S_i \geq S_j$, $S_j \in Vt$, $\geq$优于 (先被归约)

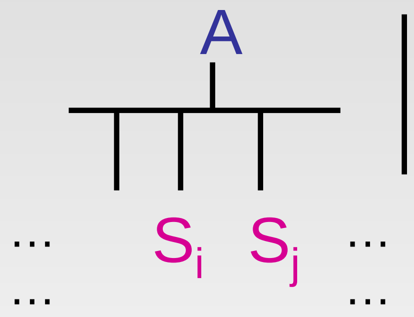
表示: S_i的优先级高于S_j的优先级



(2) 若S_i和S_j 同时处于 α 的句柄中,

则$S_i \equiv S_j$ $\equiv$等于(同时被归约)

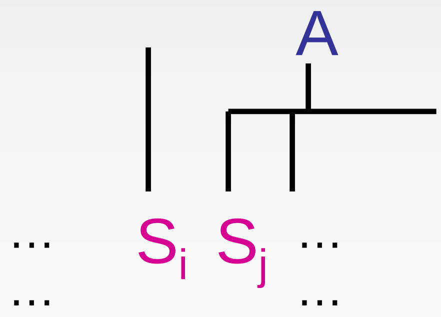
表示: S_i的优先级等于S_j的优先级



(3) 若S_j在句柄中,而S_i不在句柄中,

则$S_i \leq S_j$ $\leq$低于(后被归约)

表示: S_i的优先级低于S_j的优先级



(4) 若S_i 和S_j 均不在 α 句型的句柄中,

则可能在另一规范句型中满足上述三种情况之一.

(5)若G中某S_r,S_t属于V,不可能相邻地出现在某一规范句型中,

则S_r和S_t之间不存在任何优先关系.

注意: 上述关系不具有对称性

即$S_i \leq S_j$ 不意味着 $S_j \geq S_i$

比较:

a>b 与 b<a

$+ \geq +$ 与 $+ \leq +$

二、简单优先文法

若文法G中的任何两个符号之间至多存在一种优先关系($\lessdot$、$\doteq$、$\gtrdot$)，且任意两个不同的产生式均无相同的右部，则称G为简单优先文法。

反例 $A \rightarrow abc$

$B \rightarrow abc$

两个不同的产生式有相同右部

后果：引起二义性

例:G[E] $E \rightarrow E_1$

$E_1 \rightarrow E_1 + T_1 \mid T_1$

$T_1 \rightarrow T$

$T \rightarrow T * F \mid F$

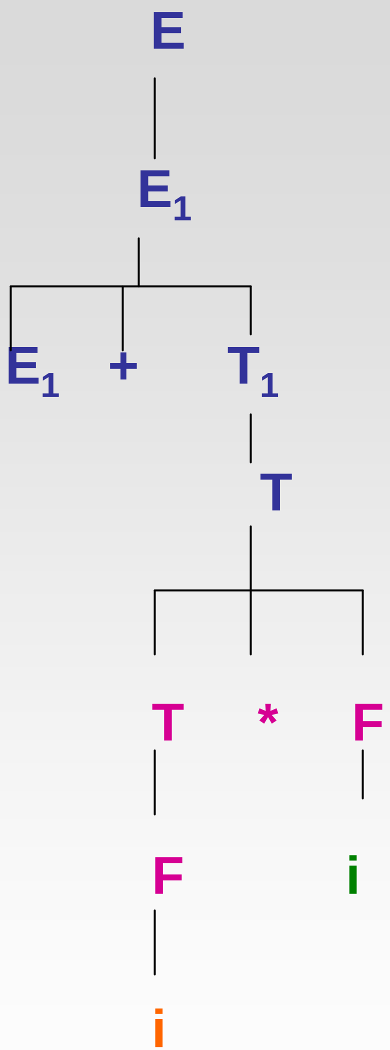
$F \rightarrow (E) \mid i$

问题:给定语法树根据定义找优先关系.

解:由 $E_1 \rightarrow E_1 + T_1$ $E_1 \equiv +$ $+ \equiv T_1$

$T \rightarrow T * F$ $T \equiv *$ $* \equiv F$

$F \rightarrow (E)$ $(\equiv E$ $E \equiv)$



根据定义：

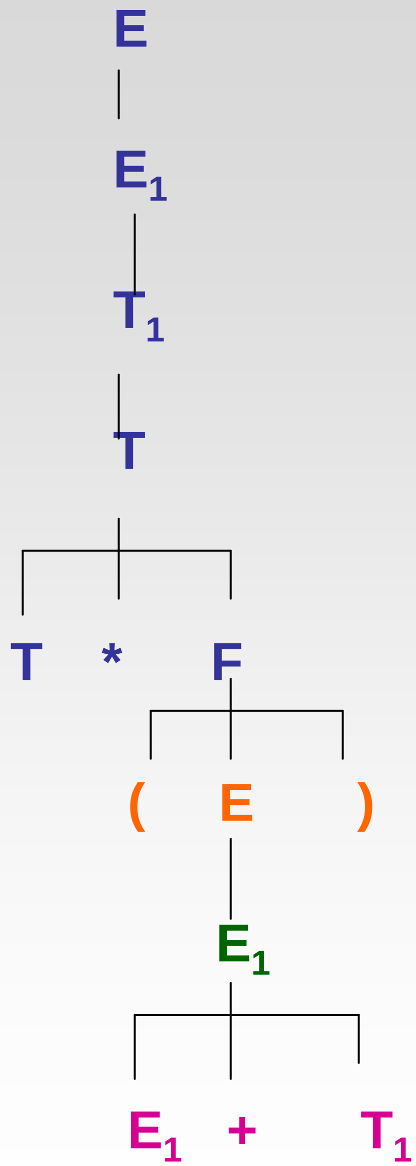
由句型 E_1+i*i

句柄i + <i , i> *

句柄F + < F , F> *

句柄i * < i

句柄T*F +<T



句型:$T*(E_1+T_1)$

句柄 $E_1 + T_1$　　$(< E_1$

$T_1 >)$

句柄 E_1　　$E_1 >)$

句柄 (E)　　$* < ($

- 按照算法可以建立优先关系矩阵(教材132页)

	E	E_1	T_1	T	F	+	*	(	)	i
E									$\doteq$	
E_1						$\doteq$			$\gtrdot$	
T_1						$\gtrdot$			$\gtrdot$	
T						$\gtrdot$	$\doteq$		$\gtrdot$	
F						$\gtrdot$	$\gtrdot$		$\gtrdot$	
+			$\doteq$	$\lessdot$	$\lessdot$			$\lessdot$		$\lessdot$
*					$\doteq$			$\gtrdot$		$\lessdot$
(	$\doteq$	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$			$\lessdot$		$\lessdot$
)						$\gtrdot$	$\gtrdot$		$\gtrdot$	
i						$\gtrdot$	$\gtrdot$		$\gtrdot$	

三、简单优先分析的算法 -----寻找句柄

1、结论:对于简单优先文法的任何规范句型

$x_1x_2 \ldots\ldots x_n$ 而言，句柄是该句型中

满足以下条件的最左子串 $x_ix_{i+1}\ldots\ldots x_{i+k}$

$$X_{i-1} < X_i$$

$$X_i = X_{i+1} = \ldots\ldots = X_{i+k}$$

$$X_{i+k} > X_{i+k+1}$$

$$x_1x_2 \ldots\ldots X_{i-1} \; x_ix_{i+1}\ldots\ldots x_{i+k} \; X_{i+k+1}\ldots\ldots x_n$$

2、算法

数据结构：(1) S为分析栈， i , j 为指示器

(2) Q , R为变量, 放 #, V_t, V_n

(3) R变量,放输入串中的一个符号

(4) $a_1a_2 \ldots \ldots a_n$#为要分析的输入串

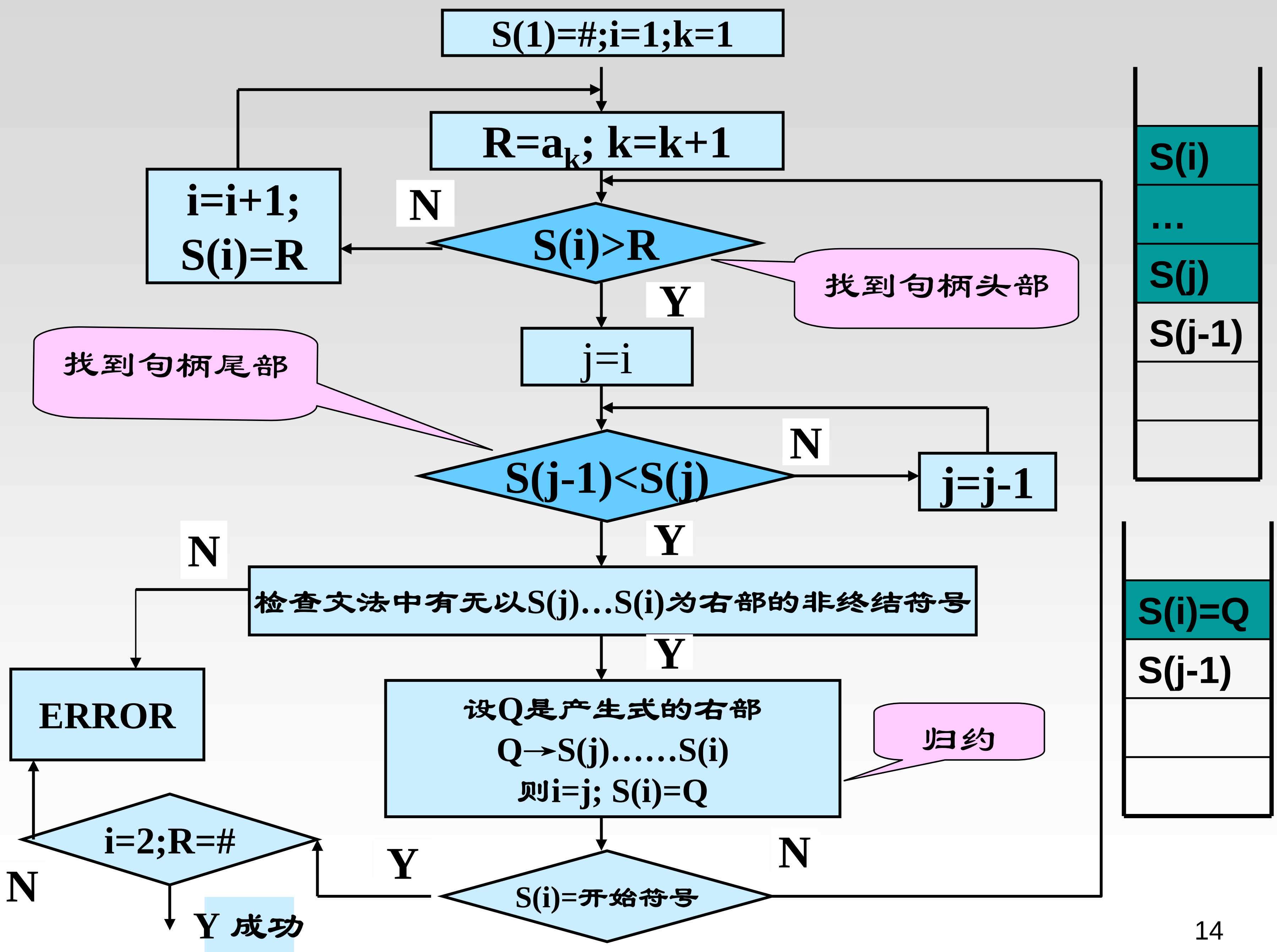
算法思想：(1) 首先找到句柄的头部(在栈顶)

首次满足$X_{i+k} > X_{i+k+1}$ 的位置

(2) 回过来找句柄的尾部(向栈内)

首次$X_{i-1} < X_i$满足的位置

(3) 中间夹的全是优先级相等的.



S(1)=#;i=1;k=1
R=a_k; k=k+1
i=i+1;
S(i)=R
N
S(i)>R
Y
找到句柄头部
j=i
找到句柄尾部
N
S(j-1)<S(j)
j=j-1
Y
N
检查文法中有无以S(j)…S(i)为右部的非终结符号
Y
ERROR
设Q是产生式的右部
Q→S(j)……S(i)
则i=j; S(i)=Q
归约
i=2;R=#
N
Y 成功
Y
N
S(i)=开始符号
S(i)
…
S(j)
S(j-1)
S(i)=Q
S(j-1)

例:输入串 i+i*i 的简单优先分析过程(查优先关系矩阵)

分析栈	优先关系	输入 R
#	#<i	i
# i	#<i>+	+
# F	#<F>+	
# T	#<T>+	
# T_1	$\#<T_1>+$	
# E_1	$E_1=+$	

分析栈	优先关系	输入 R
# E_1 +	+<i	i
# E_1 + i	+<i>*	*
# E_1 + F	+<F>*	
# E_1 + T	T=*	
# E_1 + T *	*<i	i
# E_1 + T * i	*<i>#	#
# E_1 + T * F	+<T=*=F>#	

分析栈	优先关系	输入 R
# E_1 + T	$+<T>\#$	
# E_1 + T_1	$\#<E_1=+=T_1>\#$	
# E_1	$\#<E_1>\#$	
# E		成功

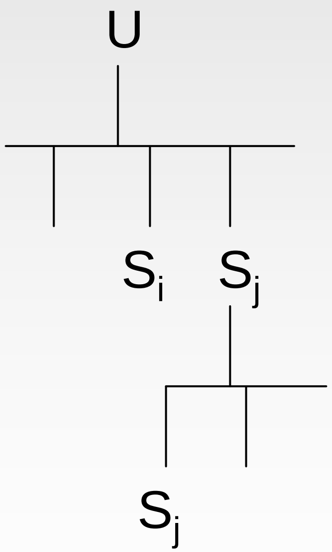
结论：i+i*i是文法的合法句子

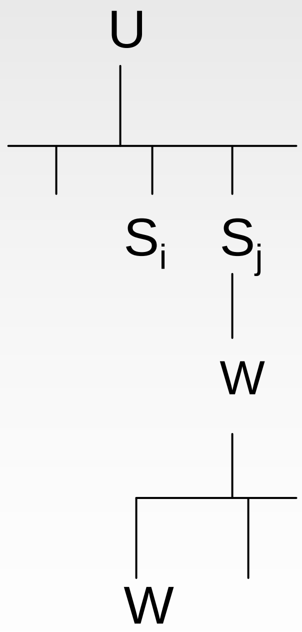
三、简单优先文法的局限性

1、多重定义优先关系----非简单优先文法

$U \rightarrow \ldots\ldots S_iS_j \ldots\ldots$　　　　$S_i \doteq S_j$　多重定义

$S_j \rightarrow S_j \ldots\ldots$　　　　$S_i \lessdot S_j$





$U \rightarrow \ldots\ldots S_iS_j \ldots$

$S_j \rightarrow W$

$W \rightarrow W \ldots\ldots$

解决的方法:改写文法

$G[E]:E \rightarrow E+T \mid T$ 不是简单优先文法

$T \rightarrow T*F \mid F$

$F \rightarrow (E) \mid i$

改写为:

$G[E] \quad E \rightarrow E_1$ 是简单优先文法

$E_1 \rightarrow E_1 + T_1 \mid T_1$

$T_1 \rightarrow T$

$T \rightarrow T*F \mid F$

$F \rightarrow (E) \mid i$

2、文法的复杂性增加,改写不能解决

$U \rightarrow \ldots\ldots S_i S_j \ldots\ldots\ldots$ $\qquad$ $S_i \doteq S_j$ 多重定义

$S_i \rightarrow \ldots\ldots S_i$ $\qquad$ $S_i \gtrdot S_i$

$S_j \rightarrow S_j \ldots\ldots$ $\qquad$ $S_i \lessdot S_j$

解决方法LR(K)分析法

3、在实际的运算过程中,决定运算顺序的是终结符号之间的优先关系.

算符优先分析法:只考虑终结符号之间的优先关系

四、简单优先关系矩阵构造

定义4.2（关系LEAD）当且仅当 G中存在形如

A →B…..的产生式时，有A LEAD B(A一步推导B)；

且仅当$A \stackrel{+}{\Rightarrow} B$…..时，有A LEAD^{+} B(A一步以上推导B).

定义4.3（关系LAST）当且仅当 G中存在形如

A →……B的产生式时，有A LAST B(A以B一步结尾)；

且仅当$A \stackrel{+}{\Rightarrow}$……B时，有A LAST^{+} B.

(A以B一步以上结尾)

传递闭包的Warshall算法

```
A:=M;{M是关系R的关系矩阵}
For i:=1 to n
    for j:=1 to n
      begin
       if A[j,i]=1 then
            for k:=1 to n
          A[j,k]:=A[j,k]+A[i,k];{逻辑加}
      end
```

Warshall算法：

已知B，计算$B^+=B+BB+BBB+\ldots\ldots B^n$

1、置新矩阵A=B

2、置i:=1

3、对所有j，如果A[j,i]=1，那么，

对k=1,……,n A[j,k]=A[j,k]+A[i,k]

4、i=i+1

5、如果i<=n，那么，转到步骤3;

否则停止。

定义4.4:

设R为一关系,R的转置TRANSPOSE(R),且定义为当且仅当aRb时, b TRANSPOSE(R) a

定理:

在同一字母表上的两个关系的乘积,可用表示这两个关系的布尔矩阵的乘积给出.

定理:

设有一n个符号的字母表S, R是S上的一个关系,而B是一个表示R的n×n布尔矩阵, 那么,

$$B^{+}=B+BB+BBB+\ldots\ldots+B^{n}$$

其中:B^{+}表示R的传递闭包R^{+}.

定义4.5

当且仅当G中存在形如$U \rightarrow \ldots S_i S_j \ldots$的产生式时

有$S_i = S_j$

定义4.6

当且仅当G中存在形如$U \rightarrow \ldots S_i W \ldots$的产生式

且$W \overset{+}{=}> S_j \ldots\ldots$时，　有$S_i < S_j$

由$U \rightarrow \ldots S_i W \ldots$　　得$S_i = W$

由$W \overset{+}{=}> S_j \ldots\ldots$　　得$W\ LEAD^+\ S_j$

结论：$S_i (=)(LEAD^+) S_j$ 时, $S_i < S_j$

定义4.7

当且仅当G中存在形如$U \rightarrow \ldots W_1 W_2 \ldots$的产生式

且$W_1 \overset{+}{\Rightarrow} \ldots\ldots S_i$, $W_2 \overset{*}{\Rightarrow} S_j \ldots\ldots$ 时 $S_j \in V_t$

有$S_i > S_j$

由$U \rightarrow \ldots W_1 W_2 \ldots$ 得$W_1 = W_2$

由$W_1 \overset{+}{\Rightarrow} \ldots\ldots S_i$ 得W_1 $LAST^+$ S_i

$S_i (TRANSPOSE(LAST^+)) W_1$

由$W_2 \overset{+}{\Rightarrow} S_j \ldots\ldots$ 得W_2 $(I+LEAD^+)$ S_j

$LEAD^* = (I+LEAD^+)$ (I为恒等关系)

结论：$S_i (TRANSPOSE(LAST^+))(=)(I+LEAD^+) S_j$

$S_i > S_j$

应用举例

文法G[S]：$S \rightarrow Ac \quad A \rightarrow AS|Aa|b$

- 需要计算上面所定义的那些关系的布尔矩阵。这些矩阵的阶数就是文法字汇表所含符号的个数。对于关系R的布尔矩阵，我们用记号B_R表示。

$S_i \, R \, S_j$ 当且仅当$B_R[i,j]=1$

相关的关系矩阵1

S→Ac A →AS|Aa|b

$$B_{=} = \begin{array}{c} \\ S \\ A \\ a \\ b \\ c \end{array} \begin{array}{c} \begin{array}{ccccc} S & A & a & b & c \end{array} \\ \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array}$$

相关的关系矩阵2

$$B_{LEAD} = \begin{array}{c} \\ S \\ A \\ a \\ b \\ c \end{array} \begin{array}{c} \begin{array}{ccccc} S & A & a & b & c \end{array} \\ \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array}$$

$$B_{LEAD^+} = \begin{array}{c} \\ S \\ A \\ a \\ b \\ c \end{array} \begin{array}{c} \begin{array}{ccccc} S & A & a & b & c \end{array} \\ \begin{pmatrix} 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array}$$

S→Ac A →AS|Aa|b

相关的关系矩阵3

$$B_< = B_= B_{LEAD^+} =$$

$$\begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

相关的关系矩阵4

$$B_{LAST} = \begin{array}{c} \\ S \\ A \\ a \\ b \\ c \end{array} \begin{array}{c} \begin{array}{ccccc} S & A & a & b & c \end{array} \\ \begin{pmatrix} 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array}$$

$$B_{LAST^+} = \begin{array}{c} \\ S \\ A \\ a \\ b \\ c \end{array} \begin{array}{c} \begin{array}{ccccc} S & A & a & b & c \end{array} \\ \begin{pmatrix} 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array}$$

$$B_{TRANSPOSE(LAST^+)} = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \end{pmatrix}$$

S→Ac A →AS|Aa|b

相关的关系矩阵5

$$B_{>} = B_{TRANSPOSE(LAST^{+})} B_{=} B_{LEAD^{*}} =$$

$$\begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

$$= \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{pmatrix}$$

相关的关系矩阵6

- 按要求S_j应该为终结符号，所以应该把非终结符号所对应的列置零

$$B'_{>} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{pmatrix} \qquad B_{>} = \begin{pmatrix} 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 \end{pmatrix}$$

作业P176

4-1（1）

4-3（1）（2）

4-4

4-8

4-9

4-13

4-20

4-31

4-33

4-35 (1)

4-36

4-38 (1)