

第六章:自底向上LR语法分析方法

提纲

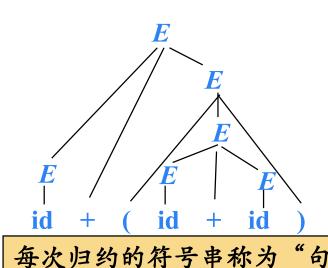
- 6.1 自底向上分析概述
- 6.2 LR分析概述
- 6.3 LR (0) 分析
- 6.4 SLR分析
- 6.5 LR (1) 分析
- 6.6 LALR分析
- 6.7 二义性文法的LR分析
- 6.8 LR分析中的错误处理

自底向上的语法分析

- ▶从分析树的底部(叶节点)向顶部(根节点)方向构造分析树
- ▶可以看成是将输入串w归约为文法开始符号S的过程
- ▶自顶向下的语法分析采用最左推导方式 自底向上的语法分析采用最左归约方式(反向构造最右推导)
- ▶自底向上语法分析的通用框架
 - ▶移入-归约分析(Shift-Reduce Parsing)

例:移入-归约分析表

文法 (1) $E \rightarrow E + E$ (2) $E \rightarrow E^*E$ $\mathfrak{B} E \to (E)$ $(4) E \rightarrow id$



剩余输入

(id+id) **\$**

id+id) \$

+id) \$

+id) \$

id) \$

id+(id+id) \$

+(id+id) \$ \$ id +(id+id) \$ \$E

\$ E+

\$ E+(\$ *E*+(id

\$ E+(E

\$ E+(E+

E+(E+id)\$ *E*+(*E*+*E*

\$ E+(E

E+(E)

\$ *E*+*E*

动作

移入 归约: $E \rightarrow id$

> 移入 移入

移入

归约: $E \rightarrow id$ 移入

) \$ 移入

) \$ 归约: $E \rightarrow id$) \$

归约: $E \rightarrow E + E$ 移入

归约: $E \rightarrow (E)$

归约: $E \rightarrow E + E$ \$E

例:移入-归约分析 栈内符号串+剩余输入="规范句型" 剩余输入 动化

	② ③	E – E – E –		\boldsymbol{E}		
<i>E</i>	+		E did	E + +	<i>E</i> id	

\$	id+(id+id) \$	
\$ id	+(id+id) \$	移入
\$ <i>E</i>	+(id+id) \$	归约: E→id

(id+id) **\$** 移入 \$E+移入 \$ E+(id+id) \$

\$ E+(E+

id) \$

移入

\$ E+E $归约: E \rightarrow (E)$ \$E归约: $E \rightarrow E + E$

移入-归约分析的工作过程

- 定在对输入串的一次从左到右扫描过程中,语法分析器将零个或多个输入符号移入到栈的顶端,直到它可以对栈顶的一个文法符号串β进行归约为止
- \triangleright 然后,它将 β 归约为某个产生式的左部
- ▶语法分析器不断地重复这个循环,直到它检测到一个语法错误,或者栈中包含了开始符号且输入缓冲区为空(当出现这种情况时,语法分析器停止运行,并宣称成功完成了语法分析)为止

移入-归约分析器可采取的4种动作

- >移入:将下一个输入符号移到栈的顶端
- 》归约:被归约的符号串的右端必然处于栈顶。语法分析器在栈中确定这个串的左端,并决定用哪个非终结符来替换这个串
- ▶接收:宣布语法分析过程成功完成
- ▶报错:发现一个语法错误,并调用错误恢复子例程

移入-归约分析中存在的问题

var

例:	栈	剩余输入	动作
$(1) \rightarrow var : $	\$	var i_A , i_B : real \$	•
$(2) < IDS > \rightarrow i$	\$ var	i_A , i_B : real \$	移入
$(3) < IDS > \rightarrow < IDS >$, i	$\$$ var i $_A$, i _B : real \$	移入
	\$ var < <i>IDS</i> >	$, i_B : real \$$	归约
$(4) < T > \rightarrow \text{real} \mid \text{int}$	\$ var < <i>IDS</i> >,	i _B : real \$	移入
	var	i_B : real \$	移入
	\$ var < <i>IDS</i> > ,	< <i>IDS</i> > : real \$	归约
	\$ var < <i>IDS</i> >,	< <i>IDS</i> >: real \$	移入
	\$ var < <i>IDS</i> >,	< <i>IDS</i> > : real \$	移入
	\$ var < <i>IDS</i> >,	< <i>IDS</i> > :< <i>T</i> > \$	归约
< <u>IDS</u> > < <u>IDS</u> > < <u>T</u> >			

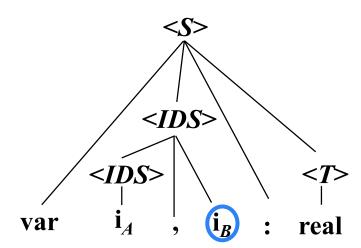
移入-归约分析中存在的问题

造成错误的原因:

错误地识别了句柄

例:

- $(1) <S> \rightarrow var <IDS>: <T>$
- $(2) < IDS > \rightarrow i$
- $(3) < IDS > \rightarrow < IDS >$, i
- $(4) < T > \rightarrow real \mid int$



栈

- \$ var
- \$ var i_A
- \$ var <*IDS*>
- \$ var < IDS>, | \$ var < IDS>
- \$ var <*IDS*>
- \$ var <*IDS*> :
- \$ var <*IDS*> : real
- \$ var <*IDS*> :<*T*>
 - \$ <S>

剩余输入

- var i_A , i_B : real \$
 - i_A, i_B : real \$
 - $, i_B : real$ \$
 - $, i_B : real$ \$
 - i_B: real \$
 : real \$
 - : real \$ 移入 : real \$ 归约
 - real \$ 移入
 - 移入

动作

移入

移入

归约

移入

- 归约 归约
- 句柄: 句型的最左直接短语

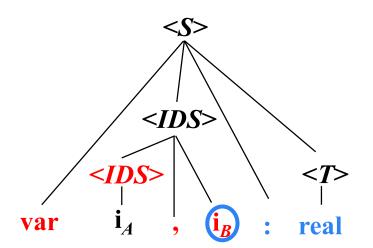
移入-归约分析中存在的问题

造成错误的原因:

错误地识别了句柄

例:

- $(1) < S \rightarrow var < IDS > : < T >$
- $(2) < IDS > \rightarrow i$
- $(3) < IDS > \rightarrow < IDS >$, i
- $(4) < T > \rightarrow real \mid int$



栈	
\$	v
\$ var	
\$ var i _A	
\$ var < <i>IDS</i> >	

Ψ	, 661		
\$	var	< <i>IDS</i> >	•

\$ var <
$$IDS>$$
, (i_B) \$ var < $IDS>$

\$ var <*IDS>* \$ var <*IDS*≥

\$ var <*IDS*

\$ var < IDS> :< T>

\$ <**S**>

剩余输入 动作

var i_A, i_B: real \$
i_A, i_B: real \$

A, i_B: real \$ 移入, i_R: real \$ 移入

, i_B: real \$ 归约

i_B: real \$ 移入: real \$ 移入

: real \$ 归约

如何正确地识别句柄?

句柄: 句型的最左直接短语

- 1在自顶向下的语法分析方法中,分析的关键是(D)。
- A.寻找句柄
- B.寻找句型
- C.消除递归
- D.选择候选式

- 2在自底向上的语法分析方法中,分析的关键是(A)。
- A.寻找句柄
- B.寻找句型
- C.消除递归
- D.选择候选式

- 3一个句型中的(A)称为该句型的句柄。
- A.最左直接短语
- B.最右直接短语
- C.终结符
- D.非终结符

- 4在规范归约中,用(B)来刻画可归约串。
- A.直接短语
- B.句柄
- C.最左素短语
- D.素短语

5下列动作中,不是自下而上分析动作的是(B)。

- A.移进
- B.展开
- C.接受
- D.报错

6下列动作中,不是自上而下分析动作的是(C)。

- A.匹配
- B.展开
- C.移进
- D.报错

7设有文法G[T]:

 $T \rightarrow T*F|F$

 $F \rightarrow F \uparrow P | P$

 $P \rightarrow (T)|a$

该文法句型T*P↑(T*F)的句柄是下列符号串(C);

该文法句型T*F↑(T*F)的句柄是下列符号串(B)。

 $\mathbf{A.} \ (\mathsf{T}^*\mathsf{F})$

B.T*F

C.P

 $\mathbf{D.P} \uparrow (T*F)$

提纲

- 6.1 自底向上分析概述
- 6.2 LR分析概述
- 6.3 LR (0) 分析
- 6.4 SLR分析
- 6.5 LR (1) 分析
- 6.6 LALR分析
- 6.7 二义性文法的LR分析
- 6.8 LR分析中的错误处理

LR 分析法

- ▶LR文法(Knuth, 1963) 是最大的、可以构造出相应 移入-归约语法分析器的文法类
 - >L: 对输入进行从左到右的扫描
 - ▶R: 反向构造出一个最右推导序列
- ►LR(k)分析
 - ▶需要向前查看k个输入符号的LR分析

k=0 和 k=1 这两种情况具有实践意义 当省略(k)时,表示k=1

LR 分析法的基本原理

- ▶自底向上分析的关键问题是什么?
 - >如何正确地识别句柄
- ▶句柄是逐步形成的,用"状态"表示句柄识别的进 展程度

- 待约状态

 \triangleright 例: $S \rightarrow bBB$

$$\triangleright S$$
→ ·bBB ← 移进状态

$$> S \rightarrow b \cdot BB$$

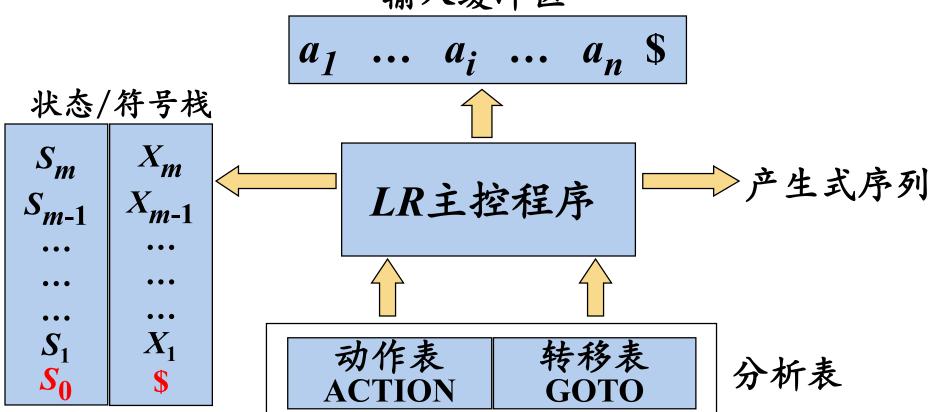
$$> S \rightarrow bB \cdot B$$

$$\triangleright S \rightarrow bBB \cdot \longleftarrow$$
 归约状态

LR分析器基于这样一些状态来 构造自动机进行句柄的识别

LR 分析器 (自动机) 的总体结构

输入缓冲区



- 〉例
 - 文文法
 - \bigcirc $S \rightarrow BB$
 - $\bigcirc B \rightarrow aB$
 - $\bigcirc B \rightarrow b$

sn: 将符号a、状态n 压入栈

rn: 用第n个产生式进行归约

状态	A	CTIO	N	GOTO		
	a	b	\$	S	B	
0	s 3	s4		1	2	
1			acc			
2	s 3	s4			5	
3	s 3	s4			6	
4	r3	r3	r3			
5	r1	r1	r1			
6	r2	r2	r2			

>例

户文法

① $S \rightarrow BB$

② $B \rightarrow aB$

 $\textcircled{3} B \rightarrow b$

北大	A	ACTION			ТО
状态	a	b	\$	S	B
0	s 3	s4		1	2
1			acc		
2	s 3	s4			5
3	s 3	s4			6
4	r3	r3	r3		
5	r1	r1	r1		
6	r2	r2	r2		

a b

栈

剩余输入

04

\$B

bab\$

- >例
 - 户文法
 - ① $S \rightarrow BB$
 - $\bigcirc B \rightarrow aB$
 - $\textcircled{3} B \rightarrow b$

北大	ACTION			N GOTO		
状态	a	b	\$	S	B	
0	s 3	s4		1	2	
1			acc			
2	s3	s4			5	
3	s3	s4			6	
4	r3	r3	r3			
5	r1	r1	r1			
6	r2	r2	r2			

栈 剩余输入

0234

\$B

ab \$

- >例
 - 〉文法
 - ① $S \rightarrow BB$
 - $\bigcirc B \rightarrow aB$
 - $\bigcirc B \rightarrow b$

北大	A	ACTION GOTO			
状态	a	b	\$	S	В
0	s 3	s4		1	2
1			acc		
2	s 3	s4			5
3	s 3	s4			6
4	r3	r3	r3		
5	r1	r1	r1		
6	r2	r2	r2		

B| **a b**

0236 \$BaB

\$

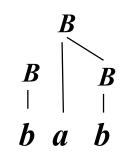
>例

〉文法

① $S \rightarrow BB$

 $\bigcirc B \rightarrow aB$

 $\bigcirc B \rightarrow b$



北大	ACTION			GOTO		
状态	a	b	\$	S	В	
0	s 3	s4		1	2	
1			acc			
2	s3	s4			5	
3	s 3	s4			6	
4	r3	r3	r3			
5	r1	r1	r1			
6	r2	r2	r2			

钱 剩余输入

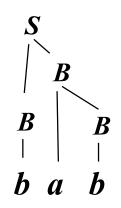
0 2 5 \$ B B

\$

>例

户文法

- ① $S \rightarrow BB$
- ② $B \rightarrow aB$
- $\textcircled{3} B \rightarrow b$



北大	ACTION			N GOTO		
状态	a	b	\$	S	В	
0	s 3	s4		1	2	
1			acc			
2	s 3	s4			5	
3	s 3	s4			6	
4	r3	r3	r3			
5	r1	r1	r1			
6	r2	r2	r2			

栈 剩余输入

\$ **S**

\$

LR 分析器的工作过程

》初始化

 s_0 $a_1a_2...a_n$

>一般情况下

$$S_0S_1...S_m$$

 $SX_1...X_m$ $a_ia_{i+1}...a_n$ \$

①如果ACTION $[s_m, a_i] = sx$,那么当前形式变为:

$$S_0S_1...S_mX$$

 $SX_1...X_ma_i$ $a_{i+1}...a_n$ \$

LR 分析器的工作过程

▶初始化

 $a_1a_2...a_n$ \$

〉一般情况下

$$S_0S_1...S_m$$

 $SX_1...X_m$ $a_ia_{i+1}...a_n$ \$

②如果ACTION $[s_m, a_i] = rx$ 表示用第x个产生式 $A \rightarrow X_{m-(k-1)}...X_m$

进行归约,那么形式变为:

$$S_0 S_1 \dots S_{m-k}$$

 $X_1 \dots X_{m-k} A \quad a_i a_{i+1} \dots a_n$

如果 $GOTO[s_{m-k}, A] = y$, 那么形式变为:

LR 分析器的工作过程

▶初始化

>一般情况下

```
s_0
a_1a_2...a_n
```

$$S_0S_1...S_m$$

 $SX_1...X_m$ $a_ia_{i+1}...a_n$ \$

- ③如果ACTION $[s_m, a_i] = acc$,那么分析成功
- ④如果ACTION $[s_m, a_i] = err$, 那么出现语法错误

LR 分析算法

- ▶ 输入: 串w和LR语法分析表,该表描述了文法G的ACTION函数和GOTO函数。
- \triangleright 输出:如果w在L(G)中,则输出w的自底向上语法分析过程中的归约步骤;否则给出一个错误指示。
- \triangleright 方法:初始时,语法分析器栈中的内容为初始状态 s_0 ,输入缓冲区中的内容为w\$。然后,语法分

析器执行下面的程序:

```
令a为w$的第一个符号;
while(1) { /* 永远重复*/
    令s是栈顶的状态:
     if (ACTION [s, a] = st) {
         将t压入栈中:
         令a为下一个输入符号:
     } else if (ACTION [s, a] =归约A \rightarrow \beta) {
         从栈中弹出 β 个符号;
         将GOTO [t, A]压入栈中;
         输出产生式A \rightarrow \beta;
     } else if (ACTION [s, a] =接受) break; /* 语法分析完成*/
     else调用错误恢复例程:
```

如何构造给定文法的LR分析表?

- ► LR(0)分析
- >SLR分析
- ►LR(1)分析
- ►LALR分析

1若a为终结符,则 $A\rightarrow \alpha$ •aβ为(B)项目。

- A.归约
- B.移进
- C.接受
- D.待约

- 2若B为非终结符,则 A→a•Bb 为(D)。
- A.移进项目
- B.归约项目
- C.接受项目
- D.待约项目

3若B为非终结符,则A→α• 为(A)项目。

- A.归约
- B.移进
- C.接受
- D.待约

4 LR分析器的核心部分是一张分析表,该表由(D)组成。

A.ACTION表

B.GOTO表

C.预测分析表

D.ACTION表和GOTO表

exercise

- 5 LR分析表中的动作表 (action) 是以 (D) 作为列标题的。
- A.终结符
- B.非终结符
- C.终结符或非终结符
- D.终结符和结束符\$

exercise

6LR分析表中的转移表 (goto) 是以 (B) 作为列标题的。

- A.终结符
- B.非终结符
- C.终结符或非终结符
- D.表示状态的整型数

提纲

- 6.1 自底向上分析概述
- 6.2 LR分析概述
- 6.3 LR (0) 分析
- 6.4 SLR分析
- 6.5 LR (1) 分析
- 6.6 LALR分析
- 6.7 二义性文法的LR分析
- 6.8 LR分析中的错误处理

LR(0) 项目

 \triangleright 右部某位置标有圆点的产生式称为相应文法的一个LR(0)项目(简称为项目)

$$A \rightarrow \alpha_1 \cdot \alpha_2$$

例: $S \rightarrow bBB$

$$\triangleright S \rightarrow b \cdot BB$$

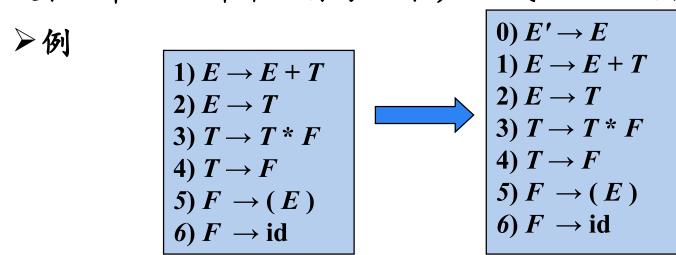
$$\triangleright S \rightarrow bBB$$
· 一归约项目

产生式 $A \rightarrow \varepsilon$ 只生成一个项目 $A \rightarrow \cdot$

项目描述了句柄识别的状态

增广文法 (Augmented Grammar)

》如果G是一个以S为开始符号的文法,则G的增广文法 G' 就是在G中加上新开始符号S'和产生式 $S' \rightarrow S$ 而得到的文法



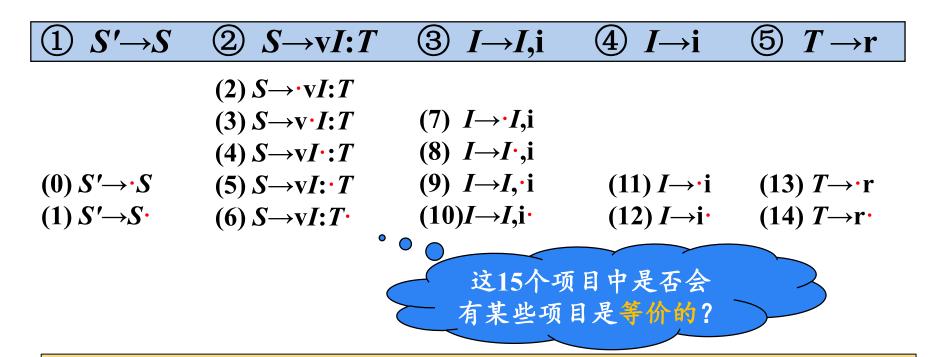
引入这个新的开始产生式的目的是使得文法开始符号仅出现在一个产生式的左边,从而使得分析器只有一个接受状态

文法中的项目

文法中的项目

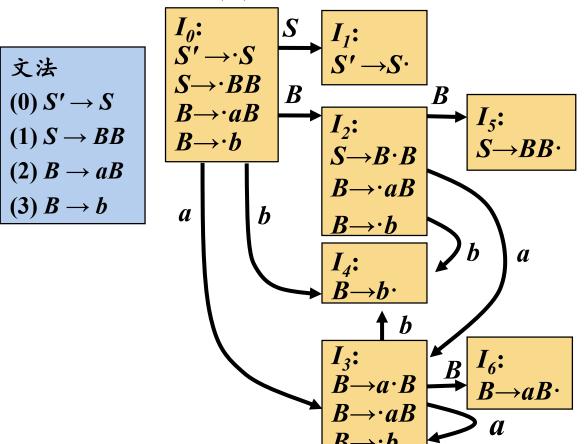
- ▶后继项目 (Successive Item)
 - 》同属于一个产生式的项目,但圆点的位置只相差一个符号, 则称后者是前者的后继项目
 - $\triangleright A \rightarrow \alpha \cdot X \beta$ 的后继项目是 $A \rightarrow \alpha X \cdot \beta$

文法中的项目



可以把等价的项目组成一个项目集(I), 称为项目集闭包 (Closure of Item Sets), 每个项目集闭包对应着自动机的一个状态

例: LR(0)自动机



LR(0)分析表

			•		
状	A	CTIC	N	GC	ТО
态	a	b	\$	S	В
0	s 3	s4		1	2
1			acc		
2	s3	s4			5
3	s 3	s4			6
4	r3	r3	r3		
5	r1	r1	r1		
6	r2	r2	r2		

CLOSURE()函数

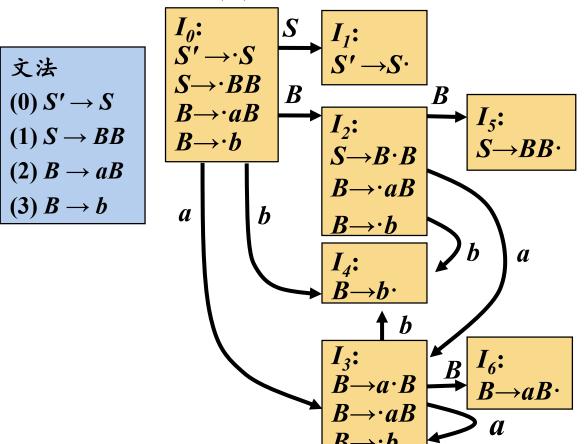
>计算给定项目集I的闭包

CLOSURE(I) = $I \cup \{B \rightarrow \gamma \mid A \rightarrow \alpha \cdot B\beta \in CLOSURE(I), B \rightarrow \gamma \in P\}$

CLOSURE()函数

```
SetofItems CLOSURE (1) {
     J=I:
     repeat
           for (J中的每个项A \rightarrow \alpha \cdot B\beta)
               for (G的每个产生式B \rightarrow \gamma)
                     将B \rightarrow \gamma m \lambda J中;
     until 在某一轮中没有新的项被加入到J中:
     return J:
```

例: LR(0)自动机



LR(0)分析表

			•		
状	A	CTIC	N	GC	ТО
态	a	b	\$	S	В
0	s 3	s4		1	2
1			acc		
2	s3	s4			5
3	s 3	s4			6
4	r3	r3	r3		
5	r1	r1	r1		
6	r2	r2	r2		

GOTO()函数

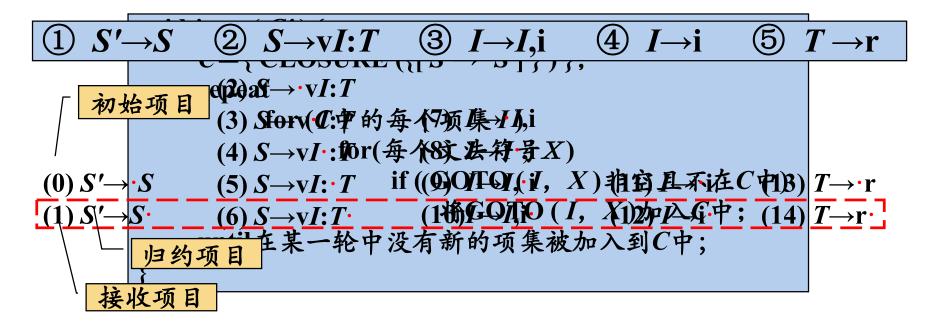
▶返回项目集I对应于文法符号X的后继项目集闭包 GOTO(I,X)=CLOSURE($\{A \rightarrow \alpha X \cdot \beta \mid A \rightarrow \alpha \cdot X \beta \in I\}$)

```
SetOfltems GOTO (I, X) { 将J 初始化为空集; for (I \text{ 中的每个项} A \rightarrow \alpha \cdot X\beta) 将项 A \rightarrow \alpha X \cdot \beta 加入到集合J 中; return CLOSURE (J); }
```

构造LR(0)自动机的状态集

▶规范LR(0) 项集族(Canonical LR(0) Collection)

 $C=\{I_{\theta}\}\cup\{I\mid\exists J\in C,X\in V_{N}\cup V_{T},I=GOTO(J,X)\}$



LR(0)分析表构造算法

如果G是一个以S为开始符号的 文法,则G的增广文法 G' 就是 在G中加上新开始符号S'和产 生式 $S' \to S$ 而得到的文法 $1)E \to E + T$ 2) $E \to T$ 3) $T \to T * F$

 $0) E' \rightarrow E$ $1) E \rightarrow E + T$ $2) E \rightarrow T$ $3) T \rightarrow T * F$ $4) T \rightarrow F$ $5) F \rightarrow (E)$ $6) F \rightarrow id$

4 *I*→i

- \triangleright 构造G'的规范LR(0)项集族 $C = \{I_0, I_1, \ldots, I_n\}$
- \triangleright 令 I_i 对应状态i。状态i的语法分析动作按照下面的方法决定:
 - $> if A \rightarrow \alpha \cdot a\beta \in I_i$ and $GOTO(I_i, a) = I_j$ then ACTION[i, a] = sj
 - \triangleright if $A \rightarrow \alpha.B\beta \subseteq I_i$ and $GOTO(I_i, B) = I_j$ then GOTO[i, B] = j
 - ightharpoonup if $A
 ightharpoonup \alpha \cdot \in I_i$ 且 $A \neq S'$ then for $\forall a \in V_T \cup \{\$\}$ do ACTION[i, a] = rj (j是产生式 $A
 ightharpoonup \alpha$ 的编号)
 - $> if S' \rightarrow S' \in I_i then ACTION [i, $] = acc$
- ▶没有定义的所有条目都设置为"error"

初始項目 (3) S→v·I:T(7) I→·I,i (4) S→vI·:T(8) I→I·,i (0) S'→·S(5) S→vI:T(9) I→I,·i(11) I→·i(13) T→·r (1) S'→S·(6) S→vI:T·(10)I→I,i·(12) I→i·(14) T→r· 接收項目 归约項目

③ *I→I*,i

② $S \rightarrow vI:T$

(2) $S \rightarrow vI:T$

5) $F \rightarrow (E)$

6) $F \rightarrow id$

LR(0) 自动机的形式化定义

户文法

$$G = (V_N, V_T, P, S)$$

≻LR(0)自动机

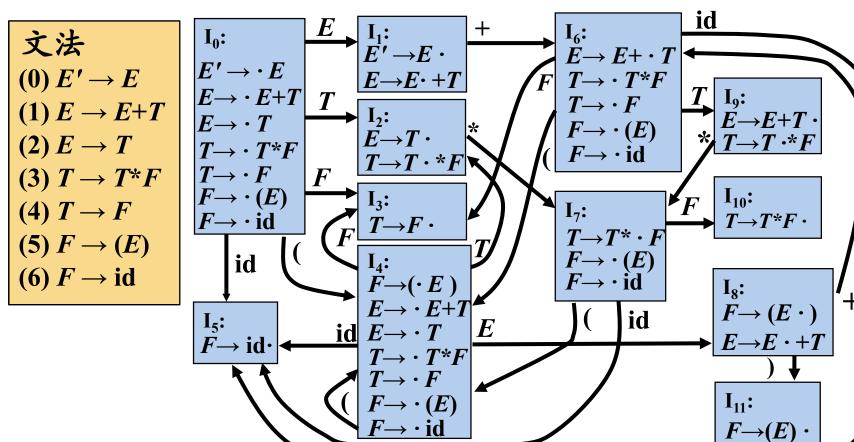
$$M = (C, V_N \cup V_T, GOTO, I_0, F)$$

$$\succ C = \{I_{\theta}\} \cup \{I \mid \exists J \in C, X \in V_N \cup V_T, I = GOTO(J,X)\}$$

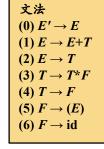
$$>I_0=CLOSURE(\{S'\rightarrow .S\})$$

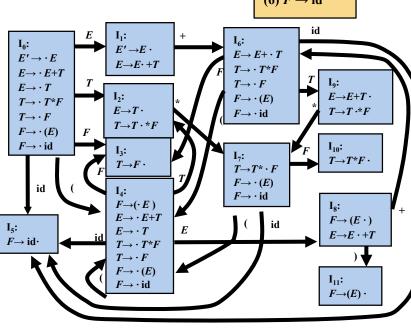
$$F = \{ CLOSURE(\{S' \rightarrow S.\}) \}$$

例



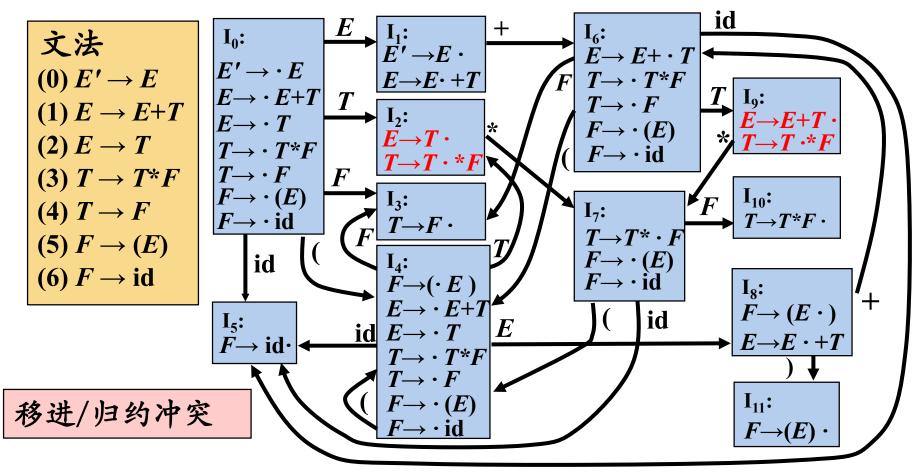






	状			ACTI	ION			(GOTO)
	态	id	+	*	()	\$	E	T	F
	0	s 5			s4			1	2	3
	1		s6				acc			
	2	r2	r2	r2/s7	r2	r2	r2			
\mathbb{N}	3	r4	r4	r4	r4	r4	r4			
$I \setminus$	4	s 5			s4			8	2	3
	5	r6	r6	r6	r6	r6	r6			
	6	s 5			s4				9	3
	7	s 5			s4					10
	8		s6			s11				
	9	r1	r1	r1/s7	r1	r1	r1			
	10	r3	r3	r3	r3	r3	r3			
	11	r5	r5	r5	r5	r5	r5			

例:移进规约冲突



表达式文法的LR(0)分析表含有移进/归约冲突

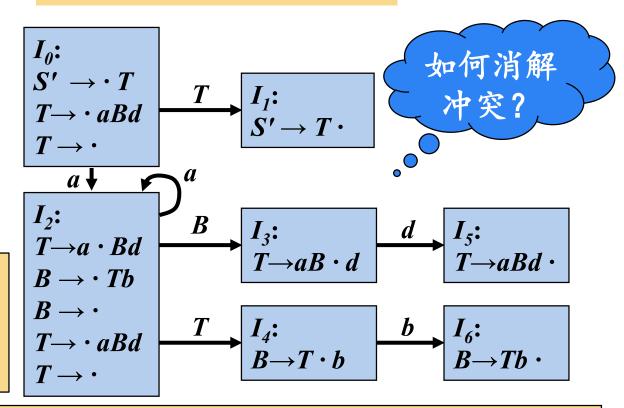
状态			AC	TION			GOTO		
水 芯	id	+	*	()	\$	E	T	F
0	s 5			s_4			1	2	3
1		s6				acc			
2	r2	r2	r2/s7	r2	r2	r2			
3	r4	r4	r4	r4	r4	r4			
4	s 5			s4			8	2	3
5	r6	r6	r6	r6	r6	r6			
6	s 5			s4				9	3
7	s 5			s4					10
8		s6			s11				
9	r1	r1	r1/s7	r1	r1	r1			
10	r3	r3	r3	r3	r3	r3			
11	r5	r5	r5	r5	r5	r5			

移进/归约冲突和归约/归约冲突



- $(0) S' \rightarrow T$
- (1) $T \rightarrow aBd$
- (2) $T \rightarrow \varepsilon$
- (3) $B \rightarrow Tb$
- (4) $B \rightarrow \varepsilon$

如果LR(0)分析表中 没有语法分析动作冲 突,那么给定的文法 就称为LR(0)文法

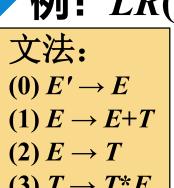


不是所有CFG都能用LR(0)方法进行分析,也就是说,CFG不总是LR(0)文法

提纲

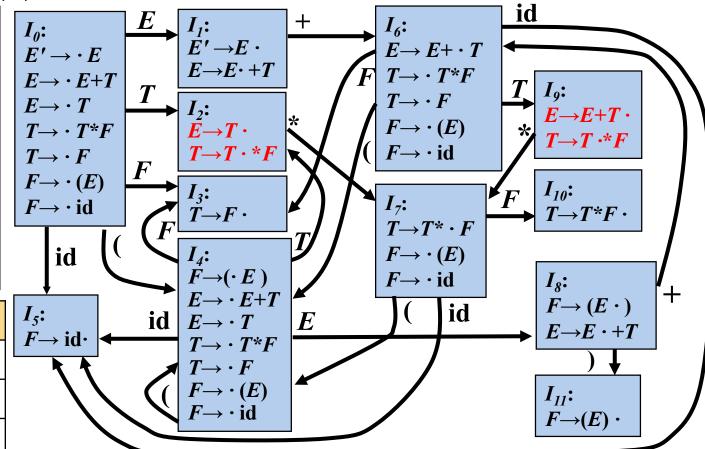
- 6.1 自底向上分析概述
- 6.2 LR分析概述
- 6.3 LR (0) 分析
- 6.4 SLR分析
- 6.5 LR (1) 分析
- 6.6 LALR分析
- 6.7 二义性文法的LR分析
- 6.8 LR分析中的错误处理

例: LR(0) 分析过程中的冲突



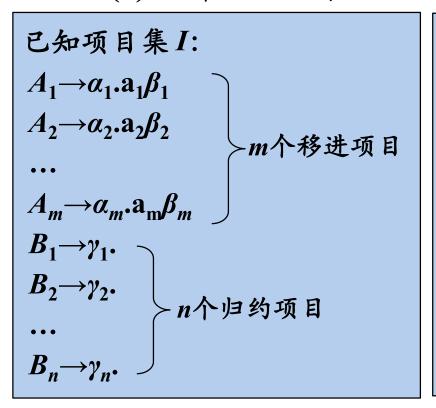
- $(3) T \to T^*F$
- $(4) T \rightarrow F$
- $(5) F \rightarrow (E)$
- (6) $F \rightarrow id$

X	FOLLOW(X)
E), +, \$
T), +, \$, *
F), +, \$,*



SLR (1)分析

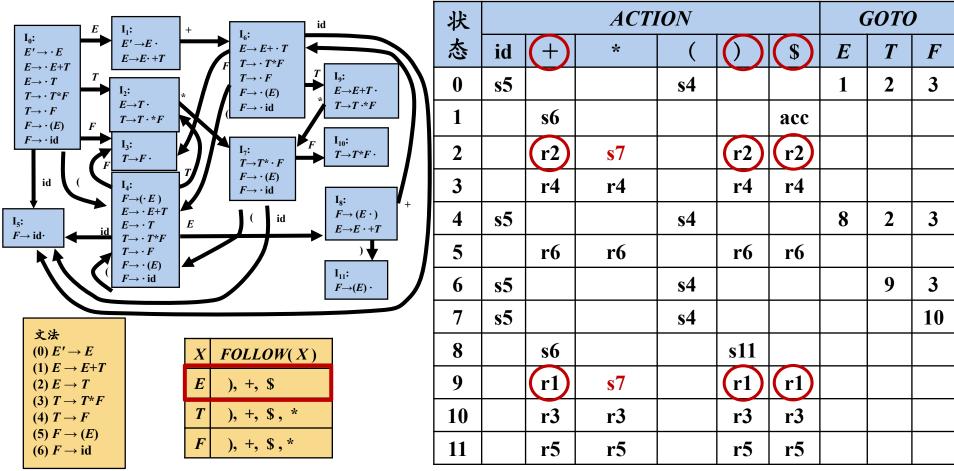
►SLR(1)分析法的基本思想



如果集合{a₁, a₂, ..., a_m}和 $FOLLOW(B_1), FOLLOW(B_2), \ldots,$ $FOLLOW(B_n)$ 两两不相交,则项目 集1中的冲突可以按以下原则解决: 设a是下一个输入符号 > 若a∈{a₁, a₂, ..., a_m}, 则移进a > 若a∈ $FOLLOW(B_i)$,则用产生式 $B_i \rightarrow \gamma_i$ 归约

>此外,报错

表达式文法的SLR(1)分析表





文法

- $(0) S' \rightarrow T$
- (1) $T \rightarrow aBd$ (2) $T \rightarrow \varepsilon$

FOLLOW(S')={ \$ } $FOLLOW(T) = \{ \$, b \}$ $FOLLOW(B)=\{d\}$

SLR(1)分析表

$ \begin{array}{c ccccccccccccccccccccccccccccccccccc$	$egin{array}{c} I_{ heta} \colon \ S' ightarrow \cdot T \end{array}$	$(3) B \rightarrow Tb$ $(4) B \rightarrow \varepsilon$
$I_2:$ $T \rightarrow a \cdot Bd$ $I_3:$ $T \rightarrow aB \cdot d$ $I_5:$ $T \rightarrow aBd$	$T \rightarrow \cdot$	→ 11·
$R \rightarrow Th$	I_2 :	
$ \begin{array}{c ccccccccccccccccccccccccccccccccccc$		T I_4 : I_6 :

状		AC'	TION	-	GOTO		
状态	а	b	d	\$	T	В	
0	s2	r2		r2	1		
1				acc			
2	s2	r2	r4	r2	4	3	
3			s5				
4		s6					
5		r1		r1			
6			r3				

SLR(1) 分析表构造算法

- \triangleright 构造G'的规范LR(0)项集族 $C = \{I_0, I_1, \dots, I_n\}$ 。
- ▶根据I_i构造得到状态i。状态i的语法分析动作按照下面的方法决定:
 - $\geq if A \rightarrow \alpha \cdot a\beta \in I_i \text{ and } GOTO(I_i, a) = I_i \text{ then } ACTION[i, a] = sj$
 - $> if A \rightarrow \alpha.B\beta \in I_i$ and $GOTO(I_i, B) = I_j$ then GOTO[i, B] = j
 - $rac{}{>}$ if $A
 ightharpoonup \alpha \cdot \in I_i$ 且 $A \neq S'$ then for $\forall a \in FOLLOW(A)$ do ACTION[i, a] = rj (j是产生式 $A
 ightharpoonup \alpha$ 的编号)
 - $\triangleright if S' \rightarrow S \in I_i then ACTION [i, \$] = acc;$
- ▶ 没有定义的所有条目都设置为"error"。

如果给定文法的SLR(1)分析表中不存在有冲突的动作,那么该文法称为SLR(1)文法

表达式的SLR(1)分析

- ユ 34
文法
$(0) E' \rightarrow E$
$(1) E \rightarrow E + T$
$(2) E \to T$ $(3) T \to T *F$
$(3) T \rightarrow T T$ $(4) T \rightarrow F$
$(5) F \rightarrow (E)$
$\begin{array}{c} (3) \ F \rightarrow (L) \\ (6) \ F \rightarrow \text{id} \end{array}$
$(0) I \rightarrow 10$

给出i+i*i的	SLR(1))分析过程	呈
----------	--------	-------	---

•										
	状			ACT	ION			(GOTO)
	态	id	+	*	()	\$	E	T	F
	0	s 5			s4			1	2	3
	1		s6				acc			
	2		r2	s7		r2	r2			
	3		r4	r4		r4	r4			
	4	s 5			s4			8	2	3
	5		r6	r6		r6	r6			
	6	s 5			s4				9	3
	7	s 5			s4					10
	8		s6			s11				
	9		r1	s 7		r1	r1			
	10		r3	r3		r3	r3			
	11		r5	r5		r5	r5			

表达式的SLR(1)分析

+id*id\$

+id*id\$

id*id\$

*id\$

*id\$

*id\$

id\$

02

01

016

0165

0163

0169

01697

016975

0169710

0169

01

\$T

\$E

\$E+

\$E+id

E+F

E+T

\$E+T *

\$E+T *id

\$E+T*F

\$E+T

\$E

i+i*i的SLR(1)分析过程 (4) $T \rightarrow F$ 状态栈 符号栈 输入 action g_0 (5) $F \rightarrow (E)$ (6) $F \rightarrow id$ id+id*id\$ go(0,F)=3**ACTION GOTO** $r_6 \rightarrow id$ 状 \$id 05 +id*id\$ id + \$ \boldsymbol{E}

文法 $(0) E' \rightarrow E$

(1) $E \rightarrow E + T$ (2) $E \rightarrow T$ (3) $T \rightarrow T^*F$

2

2

9

acc

r2

r4

r6

r1

r3

r5

8

r2

r4

r6

s11

r1

r3

r5

F

3

3

3

10

$r_4 T \rightarrow F$ go(0,T)=203 \$F +id*id\$

 $r_2 \to T$

 S_6

 S_5

 $r_6 F \rightarrow id$

 $r_4 \rightarrow F$

 S_5

acc

 $r_6 \rightarrow id go(7,F)=10$

 $r_3 T \rightarrow T^*F \quad go(6,T)=9$

 $r_1 \to E+T go(0,E)=1$

go(0,E)=1

go(6,F)=3

go(6,T)=9

s5

s5

s5

s5

3

6

7

8

9

10

11

s6

r2

r4

r6

s6

r1

r3

r5

s7

r4

r6

s7

r3

r5

s4

s4

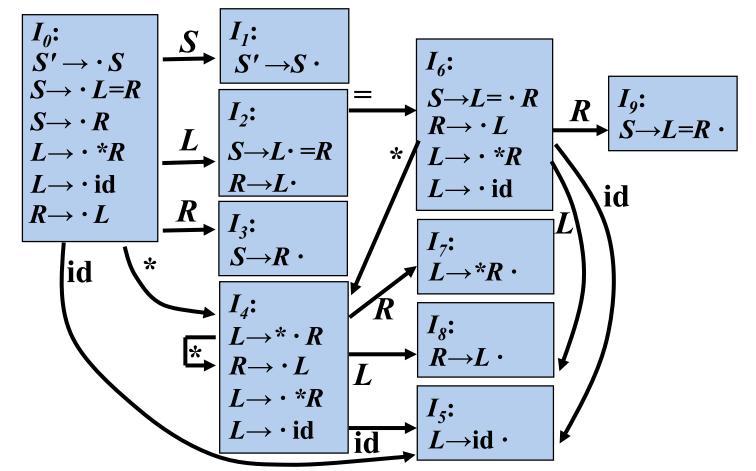
s4

s4

SLR(1) 分析举例

〉例

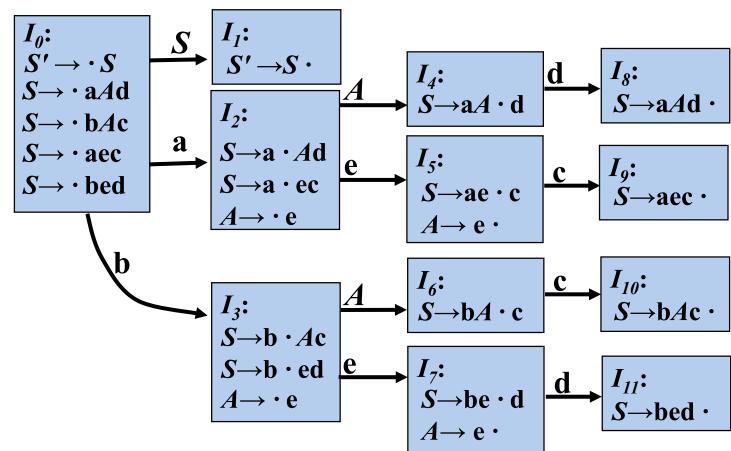
- $0) S' \rightarrow S$
- 1) $S \rightarrow L = R$
- 2) $S \rightarrow R$
- 3) $L \rightarrow *R$
- 4) $L \rightarrow id$
- 5) $R \rightarrow L$



课堂练习

〉例

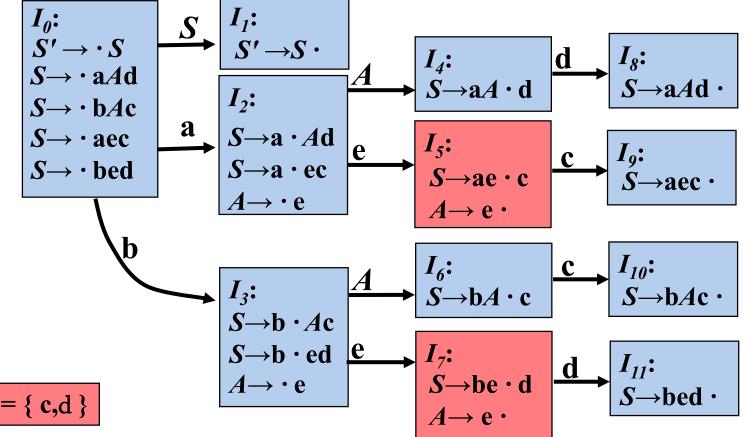
- $0) S' \rightarrow S$
- 1) $S \rightarrow aAd$
- 2) $S \rightarrow bAc$
- 3) $S \rightarrow aec$
- 4) $S \rightarrow bed$
- **5)** *A*→**e**



SLR (1)分析中的冲突



- $0) S' \rightarrow S$
- 1) $S \rightarrow aAd$
- 2) $S \rightarrow bAc$
- 3) $S \rightarrow aec$
- 4) $S \rightarrow bed$
- **5)** *A*→**e**

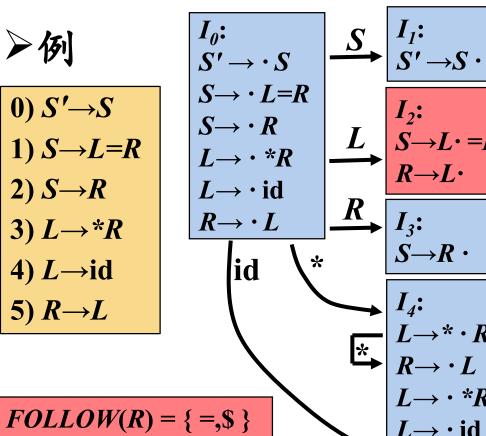


 $FOLLOW(A) = \{ c,d \}$

SLR (1)分析中的冲突



- $0) S' \rightarrow S$
- 1) $S \rightarrow L = R$
- 2) $S \rightarrow R$
- 3) $L \rightarrow *R$
- 4) $L \rightarrow id$
- 5) $R \rightarrow L$



 $egin{array}{c} I_1: \ S' {
ightarrow} S \end{array}$

 $S \rightarrow L \cdot = R$ $R \rightarrow L$

 I_3 : $S \rightarrow R$.

R $L \rightarrow * R$

id

 $L \rightarrow \cdot *R$ $L \rightarrow \cdot id$

冲突? $S \rightarrow L = \cdot R$ I_9 :

如何消解

id

 $S \rightarrow L = R$

 $L \rightarrow \cdot *R$ $L \rightarrow \cdot id$

 I_7 : $L \rightarrow *R$.

 $R \rightarrow L$

 I_8 : $R \rightarrow L$.

 I_6 :

 I_5 :

 $L \rightarrow \mathrm{id}$.

提纲

- 6.1 自底向上分析概述
- 6.2 LR分析概述
- 6.3 LR (0) 分析
- 6.4 SLR分析
- 6.5 LR (1) 分析
- 6.6 LALR分析
- 6.7 二义性文法的LR分析
- 6.8 LR分析中的错误处理

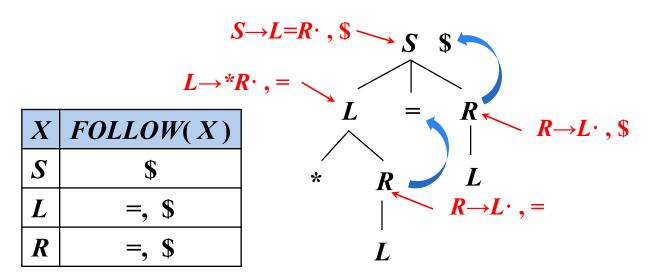
LR(1)分析法的提出

- ➤SLR(1)分析存在的问题
 - 》SLR(1)只是简单地考察下一个输入符号b是否属于与归约项目 $A\to \alpha$ 相关联的FOLLOW(A),但 $b\in FOLLOW(A)$ 只是归约 α 的一个必要条件,而非充分条件

LR(1)分析法的提出

ho对于产生式A
ightarrow lpha的归约,在不同的使用位置,A会要求不同的后继符号

- $0) S' \rightarrow S$
- 1) $S \rightarrow L = R$
- 2) $S \rightarrow R$
- 3) $L \rightarrow *R$
- 4) $L \rightarrow id$
- 5) $R \rightarrow L$



 \triangleright 在特定位置,A的后继符集合是FOLLOW(A)的子集

规范LR(1)项目

- 》将一般形式为 $[A \rightarrow \alpha \cdot \beta, a]$ 的项称为 LR(1) 项,其中 $A \rightarrow \alpha \beta$ 是一个产生式,a 是一个终结符(这里将\$视为一个特殊的终结符)它表示在当前状态下,A后面必须紧跟的终结符,称为该项的展望符(lookahead)
 - ► LR(1) 中的1指的是项的第二个分量的长度
 - \triangleright 在形如[$A \rightarrow \alpha \cdot \beta$, a] 且 $\beta \neq \epsilon$ 的项中,展望符a没有任何作用
 - ightharpoonup但是一个形如 $[A
 ightharpoonup lpha^*,a]$ 的项在只有在下一个输入符号等于a时才可以按照A
 ightharpoonup lpha 进行归约
 - \triangleright 这样的a的集合总是FOLLOW(A)的子集,而且它通常是一个真子集

等价*LR*(1)项目

$$[A \rightarrow \alpha \cdot B\beta, \mathbf{a}]$$

$$B \rightarrow \gamma \in P$$
?

等价LR(1)项目

$$[A \rightarrow \alpha \cdot B\beta, \mathbf{a}]$$

$$B \rightarrow \gamma \in P$$

$$[B \rightarrow \gamma, b]$$

$$b \in FIRST(\beta \mathbf{a})$$

当 $\beta \Rightarrow^+ \varepsilon$ 时,此时b=a叫继承的后继符,否则叫自生的后继符

例: LR(1)自动机 $S' \rightarrow S \cdot , \$$ $S \rightarrow L = R \cdot ,$ $S \rightarrow L = R,$ 0) $S' \rightarrow S$ I_0 : $S \rightarrow L = R$, \$ $R \rightarrow L$, $R \rightarrow L \cdot , \$$ $S' \rightarrow \cdot S$, \$ 1) $S \rightarrow L = R$ $R \rightarrow L \cdot , \$$ $L\rightarrow *R,$ $S \rightarrow L = R$, § 2) $S \rightarrow R$ $L \rightarrow \cdot id$, \$ $S \rightarrow R .$ R I_3 : 3) $L \rightarrow *R$ $L \rightarrow \cdot *R =$ $S \rightarrow R \cdot ,$ \$ $L\rightarrow^* \cdot R$, \$ I_7 : 4) $L \rightarrow id$ $R \rightarrow L,$ \$ $L \rightarrow \cdot id$, = id $L \rightarrow *R \cdot . =$ I_{\prime} : 5) $R \rightarrow L$ $L \rightarrow \cdot *R \cdot \$$ $R \rightarrow L.$ $L\rightarrow *R\cdot, \$$ $L \rightarrow *R =$ $L \rightarrow \cdot *R.$ $L \rightarrow \cdot id$, R $L \rightarrow *R.$ $L \rightarrow \cdot id$, \$ id $R \rightarrow L =$ $R \rightarrow L \cdot , =$ $R \rightarrow L$, \$ I_{12} : $R \rightarrow L \cdot , \$$ $L \rightarrow *R =$ $L \rightarrow id \cdot, \$$ id $L \rightarrow \cdot *R$, § id $L \rightarrow \cdot id$, = I_{13} : $L \rightarrow id \cdot , =$ $L \rightarrow \cdot id$, \$ $L\rightarrow *R\cdot, \$$ $L \rightarrow id \cdot, \$$

例: LR(1)自动机 $S' \rightarrow S \cdot , \$$ $S \rightarrow L = R \cdot ,$ \$ $S \rightarrow L = R,$ I_0 : 0) $S' \rightarrow S$ $S \rightarrow L = R$, \$ $R \rightarrow L,$ \$ $R \rightarrow L \cdot, \$$ $S' \rightarrow \cdot S$, \$ 1) $S \rightarrow L = R$ $R \rightarrow L$, \$ $L\rightarrow *R,$ $S \rightarrow L = R$, \$ 2) $S \rightarrow R$ $L \rightarrow : id, \$$ $S \rightarrow R \cdot S$ R I_3 : 3) $L \rightarrow *R$ $L \rightarrow \cdot *R =$ $S \rightarrow R \cdot ,$ \$ $L\rightarrow^* \cdot R$, \$ I_7 : 4) $L \rightarrow id$ $L \rightarrow \cdot id$, = id $R \rightarrow L,$ $L \rightarrow *R \cdot . =$ I_{\prime} : 5) $R \rightarrow L$ $L \rightarrow *R.$ $R \rightarrow L.$ $L\rightarrow *R\cdot, \$$ $L \rightarrow *R =$ $L \rightarrow \cdot *R.$ $L \rightarrow \cdot id$, \$ $L\rightarrow *R.$ $L \rightarrow \cdot id$, \$ id $R \rightarrow L, =$ $R \rightarrow L \cdot , =$ $R \rightarrow L$, \$ I_{12} : $R \rightarrow L \cdot , \$$ $L \rightarrow *R, =$ $L \rightarrow id \cdot, \$$ id $L\rightarrow *R.$ id $L \rightarrow \cdot id$, = I_{13} : $FOLLOW(R) = \{ =, \$ \}$ $L \rightarrow id \cdot , =$ $L \rightarrow \cdot id$, \$ $L\rightarrow *R\cdot, \$$ $L \rightarrow id \cdot, \$$

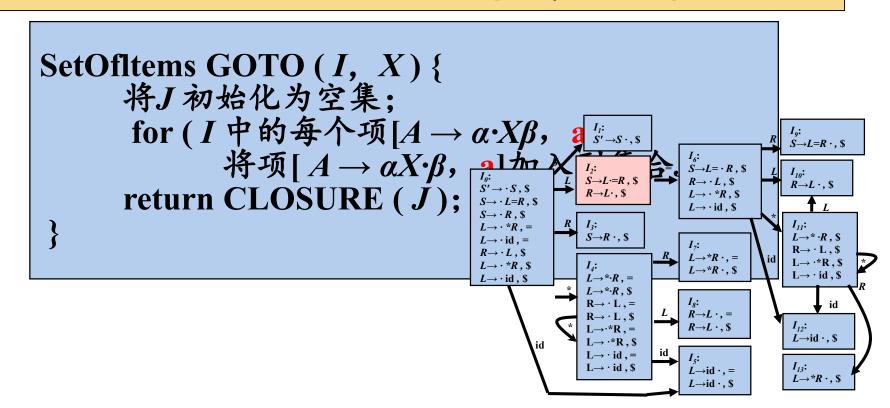
LR(1)项目集闭包

```
CLOSURE(I) = I \cup \{ [B \rightarrow \gamma, b] \mid [A \rightarrow \alpha \cdot B\beta, a] \in CLOSURE(I), B \rightarrow \gamma \in P, b \in FIRST(\beta a) \}
```

```
SetOfltems CLOSURE (1) {
      repeat
            for (I中的每个项[A \rightarrow \alpha \cdot B\beta, a ])
                for (G'的每个产生式B \rightarrow \gamma)
                       for (FIRST (βa)中的每个符号b)
                            将[B \rightarrow \cdot \gamma, b]加入到集合I中;
      until 不能向I中加入更多的项:
      until I:
```

GOTO 函数

 $GOTO(I, X) = CLOSURE(\{[A \rightarrow \alpha X \cdot \beta, \mathbf{a}] | [A \rightarrow \alpha \cdot X \beta, \mathbf{a}] \in I\})$



为文法G'构造LR(1)项集族

```
void items (G') {
    将C初始化为{CLOSURE({[S' \rightarrow \cdot S, \$]})};
    repeat
        for(C中的每个项集I)
            for(每个文法符号X)
                if (GOTO(I, X)非空且不在C中)
                    将GOTO(I, X)加入C中;
    until 不再有新的项集加入到C中:
```

LR(1)自动机的形式化定义

户文法

$$G = (V_N, V_T, P, S)$$

▶LR(1)自动机

$$M = (C, V_N \cup V_T, GOTO, I_0, F)$$

$$\gt C = \{I_{\theta}\} \cup \{I \mid \exists J \in C, X \in V_N \cup V_T, I = GOTO(J,X)\}$$

$$>I_0=CLOSURE(\{S'\rightarrow S, \$\})$$

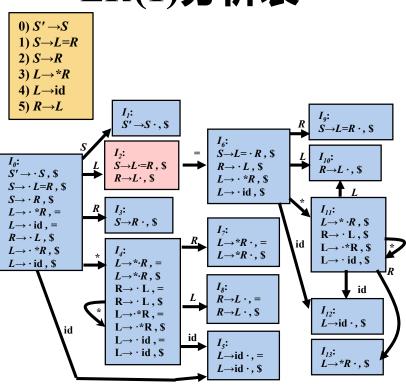
$$F = \{ CLOSURE(\{S' \rightarrow S^{\bullet}, \$\}) \}$$

LR分析表构造算法

- \triangleright 构造G'的规范LR(1)项集族 $C = \{I_0, I_1, \dots, I_n\}$
- ▶根据I_i构造得到状态i。状态i 的语法分析动作按照下面的方法决定:
 - $\succ if[A \rightarrow \alpha \cdot a\beta, b] \in I_i \text{ and } GOTO(I_i, a) = I_i \text{ then } ACTION[i, a] = sj$
 - $\geq if[A \rightarrow \alpha \cdot B\beta, b] \in I_i \text{ and } GOTO(I_i, B) = I_i \text{ then } GOTO[i, B] = j$
 - if $[A \rightarrow \alpha \cdot, \mathbf{a}]$ ∈ $I_i \coprod A \neq S'$ then $ACTION[i, \mathbf{a}] = rj$ (j是产生式 $A \rightarrow \alpha$ 的编号)
 - \triangleright if $[S' \rightarrow S; \$] \in I_i$ then ACTION[i, \$] = acc;
- ▶没有定义的所有条目都设置为"error"

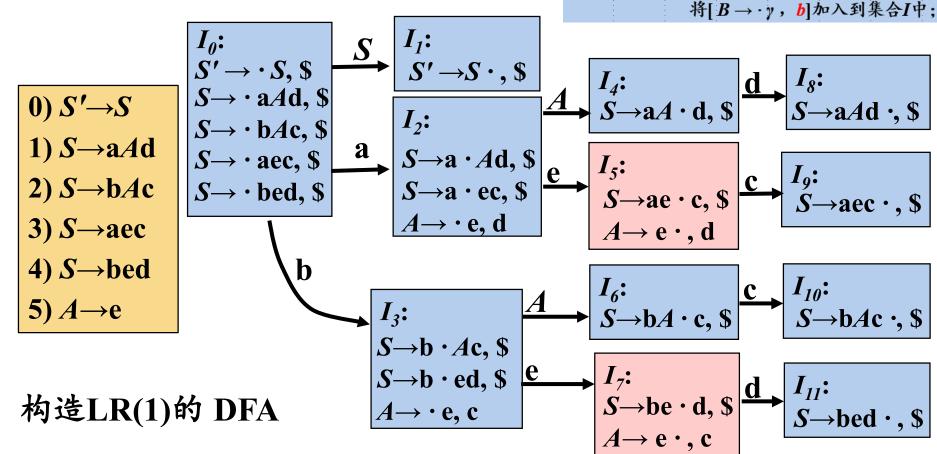
如果LR(1)分析表中没有语法分析动作冲突,那么给定的文法就称为LR(1)文法

赋值语句文法的 LR(1)分析表



状态		ACT	TION	GOTO				
大	恋	*	id	=	\$	S	L	R
0)	s4	s5			1	2	3
1	-				acc			
2),			s6	r5			
3	3				r2			
4	ļ	s4	s 5				8	7
5	;			r4	r4			
6	5	s11	s12				10	9
7	7			r3	r3			
8	3			r5	r5			
9)				r1			
1	0				r5			
1	1	s11	s12				10	13
12	2				r4			
1.	3				r3			

课堂练习

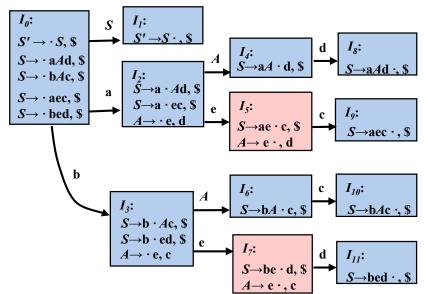


for (I中的每个项[$A \rightarrow \alpha \cdot B\beta$, **a**]) for (G'的每个产生式 $B \rightarrow \gamma$)

for (FIRST (βa)中的每个符号b)

课堂练习

- 0) S'→S
- 1) $S \rightarrow aAd$
- 2) *S*→b*A*c
- 3) $S \rightarrow aec$
- 4) $S \rightarrow bed$
- 5) *A*→e



构造LR(1)分析表

状				GOTO				
状态	a	b	c	d	e	\$	S	A
0	s2	s 3					1	
1						acc		
2					s 5			4
3					s7			6
4				s8				
5			s9	r5				
6			s10					
7			r5	s11				
8						r1		
9						r3		
10						r2		
11						r4		

课堂练习

分析bec和beec是否为该文法的句子?

0) S'→S
1) <i>S</i> →a <i>A</i> d
2) <i>S</i> →b <i>A</i> c
3) <i>S</i> →aec
4) <i>S</i> → bed
5) <i>A</i> →e

bec

状态			GOTO					
态	a	b	c	d	e	\$	S	A
0	s2	s 3					1	
1						acc		
2					s5			4
3					s7			6
4				s8				
5			s9	r5				
6			s10					
7			r5	s11				
8						r1		
9						r3		
10						r2		
11						r4		

状态栈	符号栈	输入	ACTION	GOTO
0	\$	bec\$	s3	
03	b\$	ec\$	s7	
037	eb\$	c\$	r5	goto6
036	Ab\$	c\$	s10	
0 36 <u>10</u>	cAb\$	\$	r2	goto1
01	S\$	\$	acc	

beec

状态栈	符号栈	输入	ACTION	GOTO
0	\$	beec\$	s3	
03	b\$	eec\$	s7	
037	eb\$	ec\$	err	

LR(1)自动机与SLR(1)对比分析

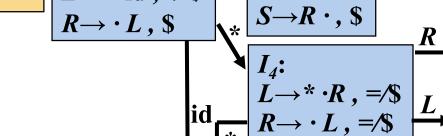
$$\begin{array}{c|c} LK(1) \boxminus \Box J \uparrow \downarrow - \exists SLK(1) X \downarrow C \Box J \uparrow \uparrow \uparrow \\ \hline 0) S' \rightarrow S \\ 1) S \rightarrow L = R \\ 2) S \rightarrow R \\ 3) L \rightarrow *R \\ 4) L \rightarrow \mathrm{id} \\ \end{array}$$

3)
$$L \rightarrow *R$$

4) $L \rightarrow \text{id}$
5) $R \rightarrow L$
$$S \rightarrow \cdot R$$
, \$\begin{align*} S \rightarrow \cdot R, \$\\ L \rightarrow \cdot \

 $FOLLOW(R) = \{ =, \$ \}$

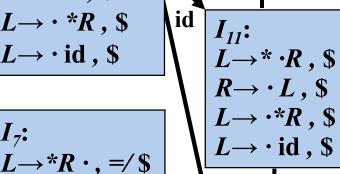
 $FOLLOW(L) = \{ =, \$ \}$



 $L\rightarrow \cdot R = /$

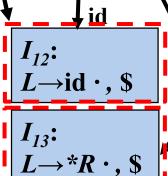
 $L \rightarrow \cdot id$, =/\$

id



 $R \rightarrow L \cdot , =/$

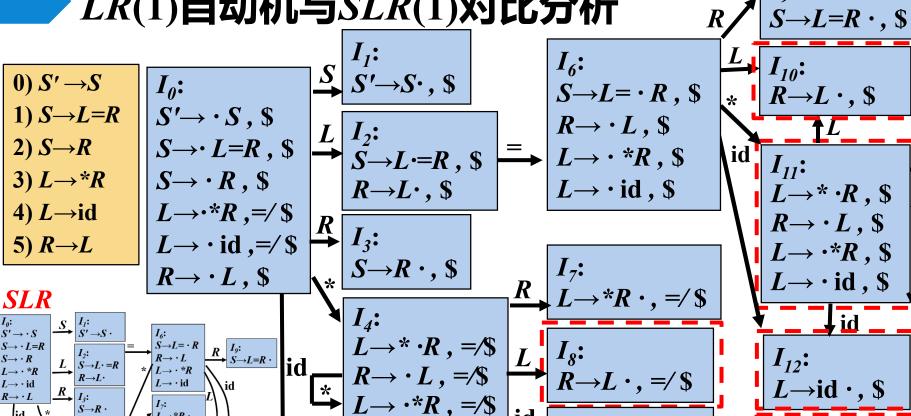
 $L \rightarrow id \cdot, =/$



 $S \rightarrow L = R \cdot ,$ \$

 $R \rightarrow L \cdot,$ \$

LR(1)自动机与SLR(1)对比分析



 $L \rightarrow id \cdot, \$$ id $L \rightarrow \cdot id$, =/\$ $L \rightarrow id \cdot, =/$ $L{\rightarrow}*R\cdot$, \$

LR(1)自动机中的同心项集 $S' \rightarrow S'$, \$ $0) S' \rightarrow S$ $S' \rightarrow \cdot S$, \$

1) $S \rightarrow L = R$ 2) $S \rightarrow R$ $S \rightarrow L = R$, \$ 3) $L \rightarrow *R$ $S \rightarrow R,$

4) $L \rightarrow id$ $L\rightarrow *R,=/$ $L \rightarrow \cdot id = /$ 5) $R \rightarrow L$ $R \rightarrow L,$ \$

 $S \rightarrow L = R$, \$ $R \rightarrow L$, \$

 $S \rightarrow R \cdot , \$$

 $L\rightarrow \cdot *R$, =/\$

 $L \rightarrow \cdot id = /$

 $L\rightarrow^*\cdot R$, =/\$ $R \rightarrow L, = /$

如果除展望符外, 两个 LR(1)项目集是相同的, 则称这两个LR(1)项目

集是同心的

 $S \rightarrow L = \cdot R$, \$ $R \rightarrow L,$ \$

 $L \rightarrow \cdot *R$, \$ $L \rightarrow \cdot \mathrm{id}$, \$

 $L\rightarrow *R\cdot, =/\$$

 $R \rightarrow L \cdot , =/$ $L \rightarrow id \cdot, =/$ $L \rightarrow \cdot \mathrm{id}$, \$ $L \rightarrow id \cdot , \$$

 $L\rightarrow^* \cdot R$, \$

 $R \rightarrow L$, \$

 $L{
ightarrow}$ *R , \$

 $S \rightarrow L = R \cdot ,$ \$

 $\star \mid R \rightarrow L \cdot, \$$

id

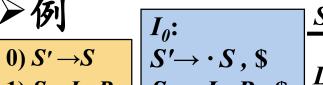
 I_{13} : $L\rightarrow *R\cdot , \$$

提纲

- 6.1 自底向上分析概述
- 6.2 LR分析概述
- 6.3 LR (0) 分析
- 6.4 SLR分析
- 6.5 LR (1) 分析
- 6.6 LALR分析
- 6.7 二义性文法的LR分析
- 6.8 LR分析中的错误处理

LALR分析法的提出





$$\begin{array}{c|c}
0) S' \rightarrow S \\
1) S \rightarrow L = R \\
2) S \rightarrow R \\
\end{array}$$

$$S' \rightarrow S, S$$

$$S \rightarrow L = R, S$$

2)
$$S \rightarrow R$$

3) $L \rightarrow *R$ $S \rightarrow `R, \$$
 $L \rightarrow `*R, =/\$$

4) $L \rightarrow id$

〉例

5) $R \rightarrow L$

 $S' \rightarrow S'$, \$

 $S \rightarrow L = R$, \$ $R \rightarrow L$., \$

 $L\rightarrow \cdot *R = /$

 $L \rightarrow \cdot id = /$

id

 $L \rightarrow \cdot id = /$

 $S \rightarrow R \cdot , \$$ $R \rightarrow L,$ \$

> $L\rightarrow * \cdot R$, =/\$

LR(1)分析器 R

 $S \rightarrow L = \cdot R$, \$ $* \mid R \rightarrow L \cdot, \$$ $R \rightarrow L,$

 $L \rightarrow \cdot *R.$ $L \rightarrow \cdot id$, \$

 $L \rightarrow R \cdot = /$

*I*₁₂: $R \rightarrow L \cdot, =/$ $L \rightarrow id \cdot , \$$

id

 $S \rightarrow L = R \cdot ,$ \$

 $L\rightarrow^* R,$

 $R \rightarrow L.$

 $L{
ightarrow}$ *R , \$

 $L{
ightarrow}\cdot {\sf id}$, \$

 I_{13} : $L \rightarrow id \cdot, =/$ $L\rightarrow *R\cdot , \$$

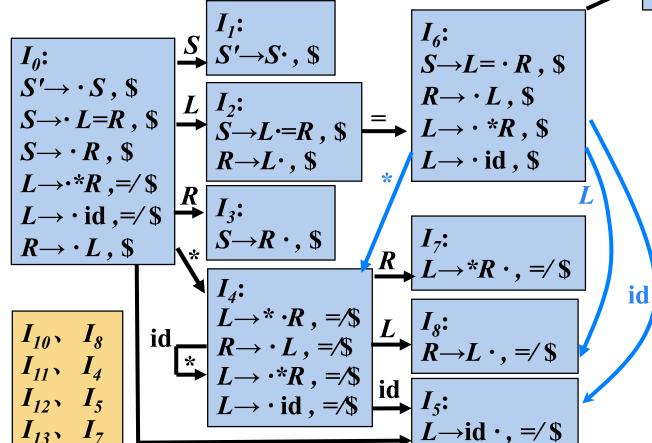
LALR (lookahead-LR)分析的基本思想

- ▶寻找具有相同核心的LR (1) 项集,并将这些项集合并为一个项集。所谓项集的核心就是其第一分量的集合
- >然后根据合并后得到的项集族构造语法分析表
- ▶如果分析表中没有语法分析动作冲突,给定的文法就称为LALR (1) 文法,就可以根据该分析表进行语法分析

例:合并同心项集 LALR自动机 $S \rightarrow L = R \cdot ,$ \$ I_6 : $S \supset S' \rightarrow S'$, \$ I_{θ} : $S \rightarrow L = R,$ $R \rightarrow L$, \$ $S' \rightarrow \cdot S$, \$ id $R \rightarrow L$, \$ $S \rightarrow L = R$, \$ $L \rightarrow \cdot *R$, \$ $S \rightarrow L = R$, \$ I_{11} : $S \rightarrow R$, \$ $R \rightarrow L$, \$ $L \rightarrow \cdot \mathrm{id}$, \$ $L\rightarrow^* R,$ $L\rightarrow *R,=/$ $R \rightarrow L,$ \$ $L \rightarrow \cdot id = /$ $L \rightarrow \cdot *R$, \$ $S \rightarrow R \cdot , \$$ $R \rightarrow L$, \$ $L \rightarrow \cdot id$, \$ $L \rightarrow *R \cdot , =/\$$ lid id $L\rightarrow^*\cdot R$, =/\$ I_{12} : I_{10}, I_{8} $\stackrel{\text{id}}{\Longrightarrow} R \rightarrow \cdot L, = /\$$ $L \rightarrow \cdot *R, = /\$$ $R \rightarrow L \cdot, =/$ $L \rightarrow id \cdot, \$$ id $L \rightarrow \cdot id$, =/\$ *I*₁₃: $L \rightarrow id \cdot, =/$ $L\rightarrow *R\cdot, \$$

例:合并同心项集 LALR自动机

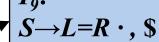
 $S \rightarrow L = R \cdot ,$ \$



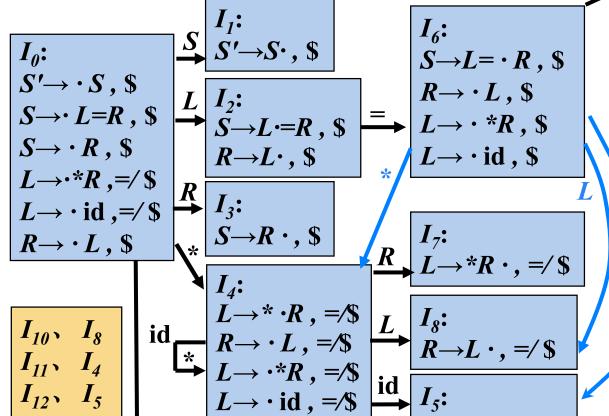
LALR分析表的构造 与LR(1)分析表一样

例:合并同心项集 LALR自动机

 $L \rightarrow id \cdot, =/$



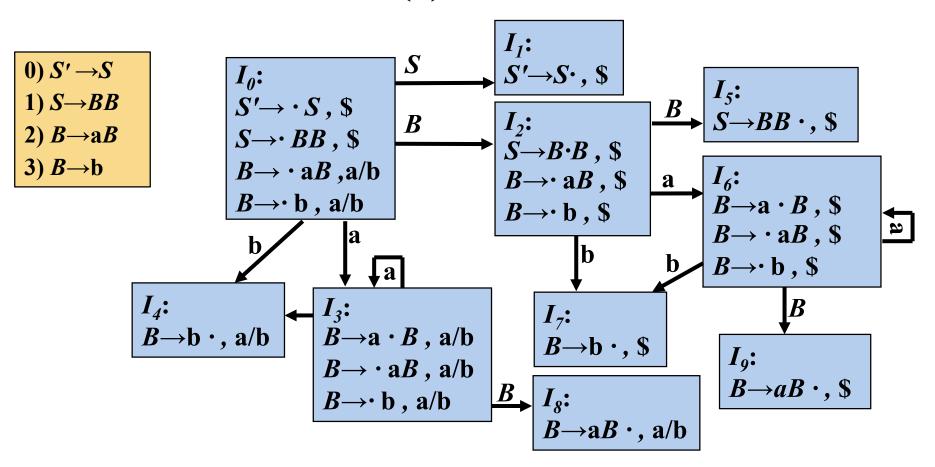
id



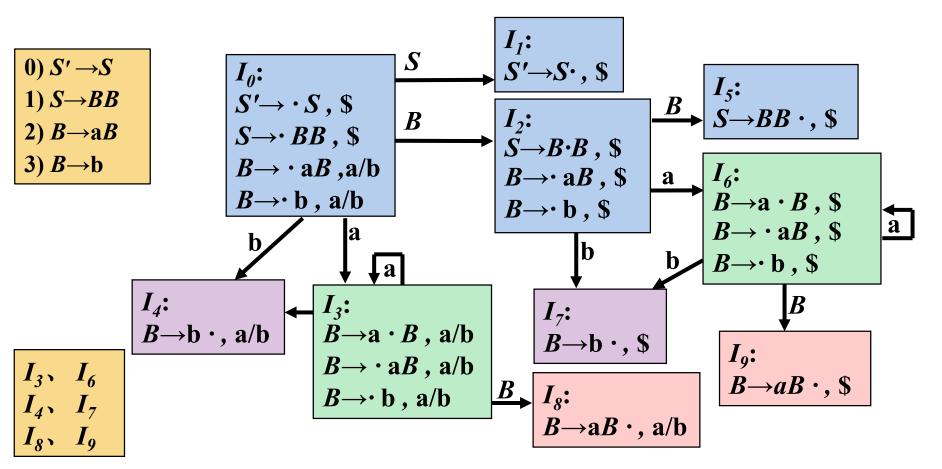
LALR分析表

状		AC	TION		G	GOTO		
态	*	id	=	\$	S	L	R	
0	s4	s 5			1	2	3	
1				acc				
2			s6	r5				
3				r2				
4	s4	s 5				8	7	
5			r4	r4				
6	s4	s5				8	9	
7			r3	r3				
8			r5	r5				
9				r1				

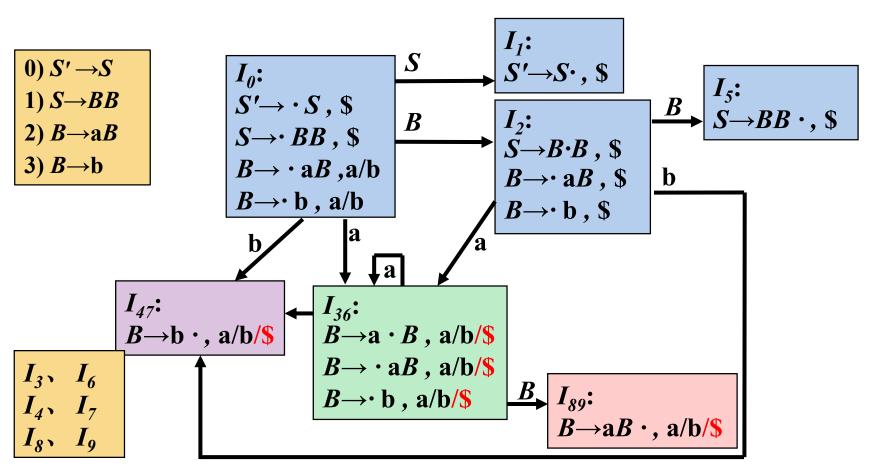
课堂练习-构造LR(1)项目集DFA



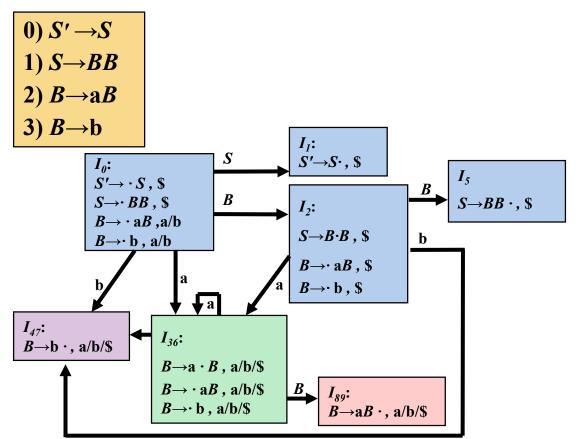
课堂练习-合并LR(1)同心项目集



课堂练习-合并LR(1)同心项目集



课堂练习-构造LALR分析表



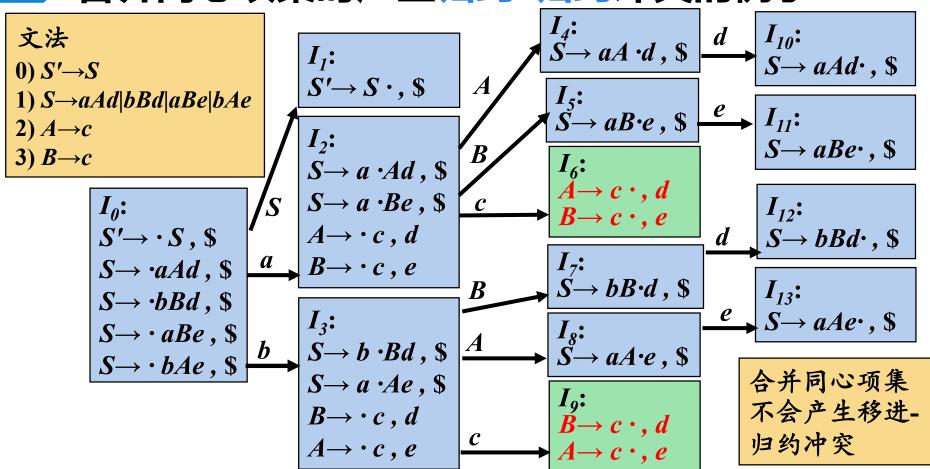
LALR分析表

状态	1	<i>ACTIO</i>	GOTO		
态	a	b	\$	S	В
0	s36	s47		1	2
1			acc		
2	s36	s47			5
36	s36	s47			89
47	r3	r3	r3		
5			r1		
89	r2	r2	r2		

课堂练习-对输入串aabaab用LALR进行分析

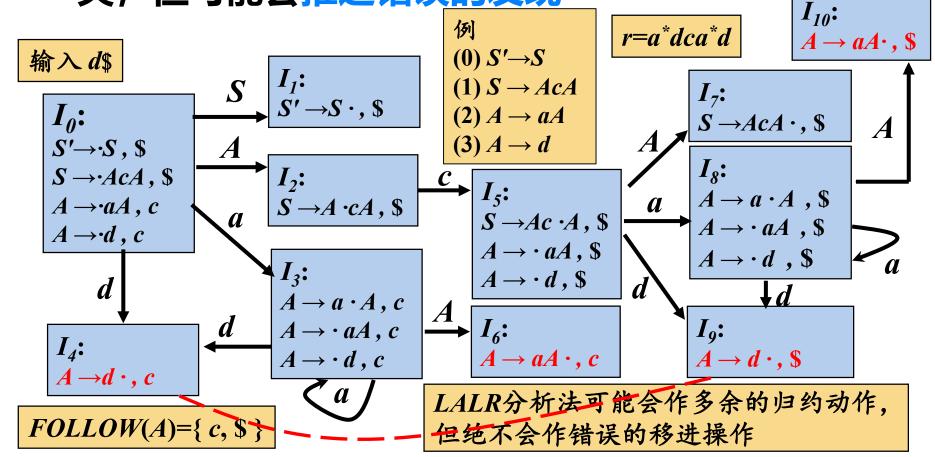
	<i>→BB</i>					状态栈 0	符号栈	输入串 aabaab\$	action s36	goto
2) B- 3) B-		LALR分析表			0 <u>36</u>	a\$	abaab\$	s36		
状		ACTIO)N	GO	TO	0 <u>36</u> 36	aa\$	baab\$	s47	
态						0 <u>36</u> <u>36</u> <u>47</u>	baa\$	aab\$	r3	89
	a	b	\$	S	B	0 <u>36</u> <u>36</u> <u>89</u>	Baa\$	aab\$	r2	89
0	s36	s47		1	2	0 <u>36</u> <u>89</u>	Ba\$	aab\$	r2	2
1			асс			02	B\$	aab\$	s36	
			ucc			02 <u>36</u>	aB\$	ab\$	s36	
2	s36	s47			5	02 <u>36</u> <u>36</u>	aaB\$	b\$	s47	
36	s36	s47			89	02 <u>36</u> <u>36</u> <u>47</u>	baaB\$	\$	r3	89
47	2	2	2			02 <u>36</u> <u>36</u> <u>89</u>	BaaB\$	\$	r2	89
47	r3	r3	r3			02 <u>36</u> <u>89</u>	BaB\$	\$	r2	5
5			r1			025	BB\$	\$	r1	1
89	r2	r2	r2			01	S\$	\$	acc	

合并同心项集时产生归约-归约冲突的例子

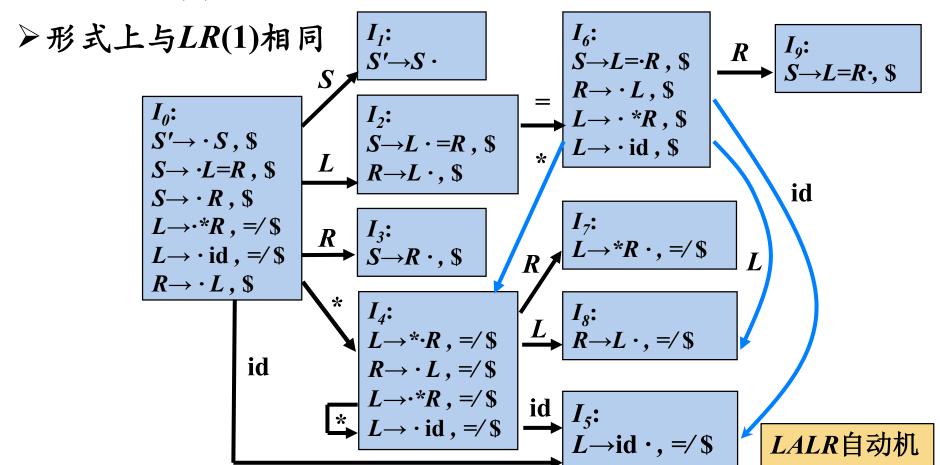


合并同心项集后,虽然节省了空间,也不产生冲

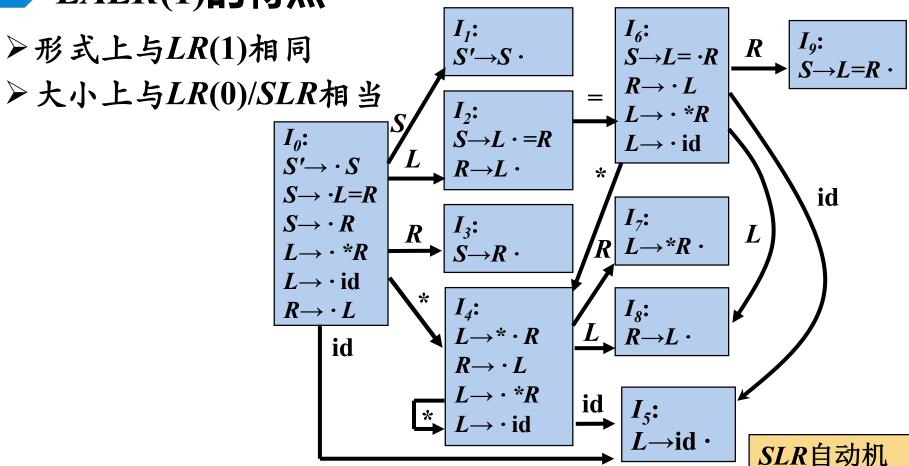
突,但可能会推迟错误的发现



LALR(1)的特点



LALR(1)的特点



LALR(1)的特点

- ▶形式上与LR(1)相同
- ▶大小上与LR(0)/SLR相当
- ▶分析能力介于SLR和LR(1)二者之间

SLR < LALR(1) < LR(1)

▶合并后的展望符集合仍为FOLLOW集的子集

Exercise

- $1- \uparrow LR(1)$ 文法合并同心集后若不是LALR(1)文法()。
- A.则可能存在移进/归约冲突
- B.则可能存在归约/归约冲突
- C.则可能存在移进/归约冲突和归约/归约冲突
- D.以上说法都不对

Exercise

2若状态k含有项目 " $A \rightarrow \alpha$ ·", 且仅当输入符号 $a \in FOLLOW(A)$ 时, 才用规则 " $A \rightarrow \alpha$ "归约的语法分析 方法是()。

- A. LALR分析法
- B. R(0)分析法
- C. LR(1)分析法
- D. SLR(1)分析法

Exercise

3同心集合并可能会产生新的()冲突。

- A.二义
- B.移进/移进
- C.移进/归约
- D.归约/归约

4就文法的描述能力来说,有()。

 $A.SLR(1) \subset LR(0)$

 $B.LR(1) \subset LR(0)$

 $C.SLR(1) \subset LR(1)$

D.无二义文法 ⊂ LR(1)

5在LR(0)的Action表中,如果某行中存在标记为"rj"的栏,则()。

A.该行必定填满"rj"

B.该行未必填满"rj"

C.其他行也有"rj"

D.goto表中也有"rj"

6若状态k含有项目 " $A \rightarrow \alpha$ ·", 对任意非终结符a, 都用规则 " $A \rightarrow \alpha$ "归约的语法分析方法是()。

A.LALR分析法

B.LR(0)分析法

C.LR(1)分析法

D.SLR(1)分析法

7在SLR(1)的Action表中,如果某行中存在标记为"rj"的栏,则()。

A.该行必定填满"rj"

B.该行未必填满"rj"

C.其他行也有"rj"

D.goto表中也有"rj"

提纲

- 6.1 自底向上分析概述
- 6.2 LR分析概述
- 6.3 LR (0) 分析
- 6.4 SLR分析
- 6.5 LR (1) 分析
- 6.6 LALR分析
- 6.7 二义性文法的LR分析
- 6.8 LR分析中的错误处理

二义性文法的特点

- ▶每个二义性文法都不是LR的
- 产某些类型的二义性文法在语言的描述和实现中很有用
 - ▶更简短、更自然
 - 〉例

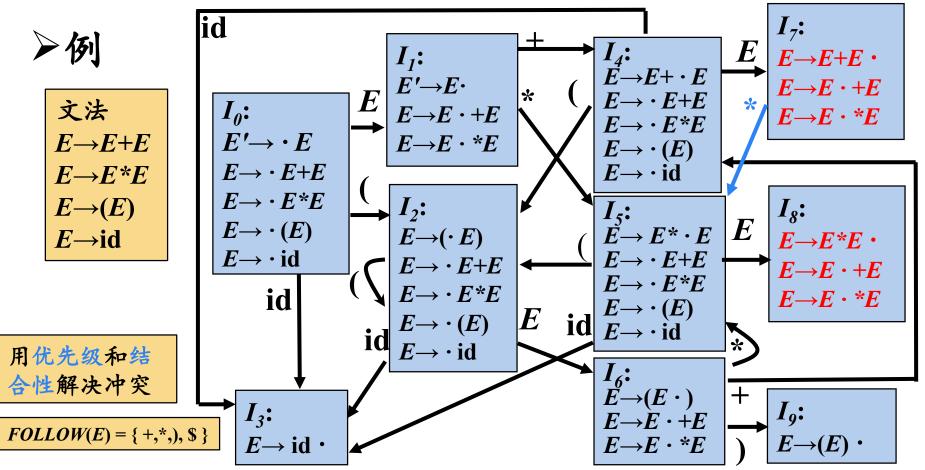
二义性文法

- $\textcircled{1} E \rightarrow E + E$
- $\bigcirc E \rightarrow E * E$
- $\textcircled{4} E \rightarrow \operatorname{id}$

非二义性文法

- $\textcircled{1} E \to E + T$
- $2E \rightarrow T$
- $\textcircled{4} T \to F$
- $\bigcirc F \rightarrow (E)$
- $\bigcirc F \rightarrow id$

二义性算术表达式文法的LR(0)分析器



二义性算术表达式文法的SLR分析表

文法 $E \rightarrow E + E$ $E \rightarrow E * E$ $E \rightarrow (E)$ $E \rightarrow id$

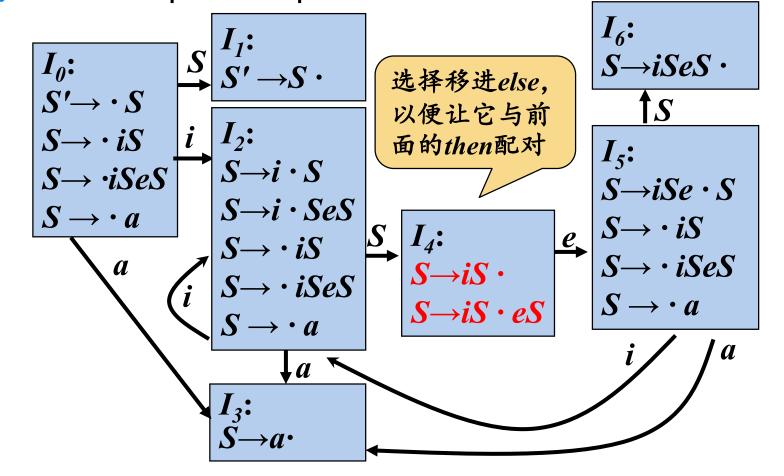
状	ACTION						
状态	id	+	*	()	\$	E
0	s 3			s2			1
1		s4	s 5			acc	
2	s 3			s2			6
3		r4	r4		r4	r4	
4	s 3			s2			7
5	s 3			s2			8
6		s4	s 5		s9		
7		r1	s 5		r1	r1	
8		r2	r2		r2	r2	
9		r3	r3		r3	r3	

 $FOLLOW(E) = \{ +, *,), \$ \}$

例:二义性if语句文法的LR分析

>S
$$\rightarrow$$
 if expr then S
|if expr then S else S
|other | e
| a

$S \rightarrow i S \mid i S e S \mid a$ LR(0) DFA



二义性if语句文法的SLR分析表

状		GOTO			
状态	i	e	a	\$	S
0	s2		s 3		1
1				acc	
2	s2		s 3		4
3		r3		r3	
4		s 5		r1	
5	s2		s 3		6
6		r2		r2	

二义性文法的使用

▶应该保守地使用二义性文法,并且必须在严格 控制之下使用,因为稍有不慎就会导致语法分 析器所识别的语言出现偏差

- 1LR(1)文法都是()。
- A.无二义性且无左递归
- B.可能有二义性但无左递归
- C.无二义性但可能是左递归
- D.可以既有二义性又有左递归

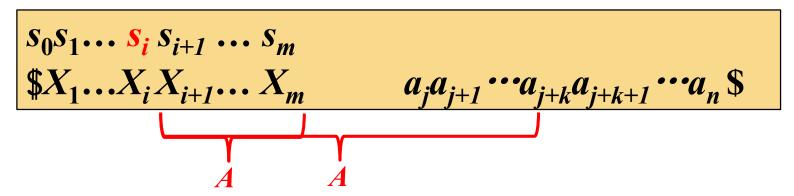
提纲

- 6.1 自底向上分析概述
- 6.2 LR分析概述
- 6.3 LR (0) 分析
- 6.4 SLR分析
- 6.5 LR (1) 分析
- 6.6 LALR分析
- 6.7 二义性文法的LR分析
- 6.8 LR分析中的错误处理

LR分析中的错误处理

- 产语法错误的检测
 - ▶当LR分析器在查询分析表并发现一个报错条目时, 就检测到了一个语法错误
- > 错误恢复策略
 - > 恐慌模式错误恢复
 - ▶短语层次错误恢复

恐慌模式错误恢复

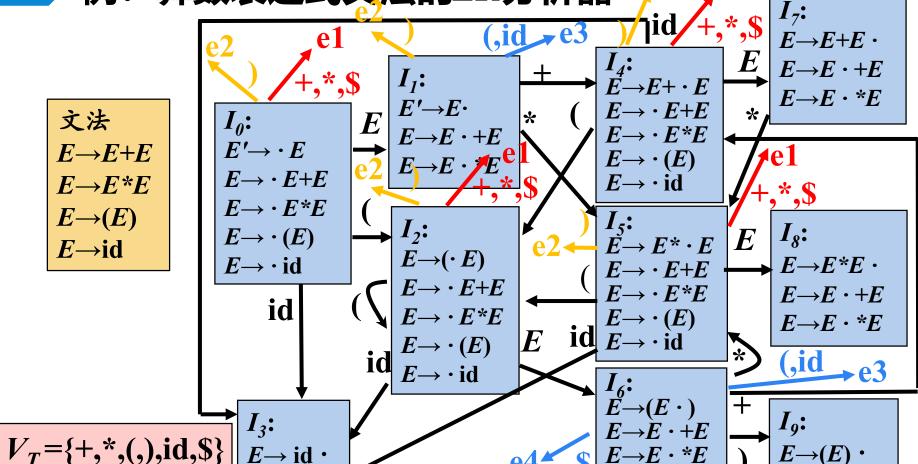


- \triangleright 从栈顶向下扫描,直到发现某个状态 s_i ,它有一个对应于某个非终结符A的GOTO目标,可以认为从这个A推导出的串中包含错误
- ► 然后丢弃0个或多个输入符号,直到发现一个可能合法地跟在A之后的符号a为止。
- \triangleright 之后将 s_{i+1} = $GOTO(s_i, A)$ 压入栈中,继续进行正常的语法分析

短语层次错误恢复

- ▶检查LR分析表中的每一个报错条目,并根据语言的使用方法来决定程序员所犯的何种错误最有可能引起这个语法错误
- > 然后构造出适当的恢复过程

例: 算数表达式文法的LR分析器



带有错误处理子程序的算术表达式文法LR分析表

状	ACTION						GOTO
态	id	+	*	()	\$	E
0	s 3	e1	e1	s2	e2	e1	1
1	e3	s4	s 5	e3	e2	acc	
2	s 3	e1	e1	s2	e2	e1	6
3	r4	r4	r4	r4	r4	r4	
4	s 3	e1	e1	s2	e2	e1	7
5	s 3	e1	e1	s2	e2	e1	8
6	e3	s4	s 5	e3	s9	e4	
7	r1	r1	s 5	r1	r1	r1	
8	r2	r2	r2	r2	r2	r2	
9	r3	r3	r3	r3	r3	r3	

> 错误处理例程

- ▶ e1: 将状态3压入栈中,发出 诊断信息"缺少运算分量"
- ► e2: 从输入中删除")",发 出诊断信息"不匹配的右括号"
- ▶ e3:将状态4压入栈中,发出 诊断信息"缺少运算符"
- 》 e4: 将状态9压入栈中,发出 诊断信息"缺少右括号"

自底向上语法分析小结

	Top_down	Bottom_up		
通用框架	递归下降分析	移入-规约分析		
主要问题	候选式冲突	冲突 [规约-规约冲突 移入-规约冲突		

例:规约-规约冲突

例:

 $(1) < S \rightarrow var < IDS > : < T >$

 $(2) < IDS > \rightarrow i$

 $(3) < IDS > \rightarrow < IDS >$, i

 $(4) < T > \rightarrow real \mid int$

栈	剩余输入	动作
\$	var i_A , i_B : real \$	
\$ var	i_A, i_B : real \$	移入
$\$$ var i_A	, i _B : real \$	移入
\$ var < <i>IDS</i> >	$, i_B : real $ \$	归约
\$ var < <i>IDS</i> >,	i_B : real \$	移入

: real \$

移入

例:移入-规约冲突

```
文法:
(1) E \rightarrow E+T
(2) E \rightarrow T
(3) T \rightarrow T*F
(4) T \rightarrow F
```

 $(5) F \rightarrow (E)$

(6) $F \rightarrow id$

```
栈
$ id<sub>A</sub>
$ F
$ T
```

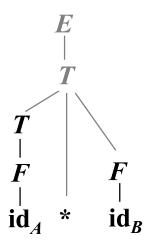
剩余输入 id_A*id_B\$ *id_B\$

* id_R\$

* id_B\$



动作



自底向上语法分析小结

	Top_down	Bottom_up		
通用框架	递归下降分析	移入-规约分析		
主要问题	候选式冲突	冲突 [规约-规约冲突 移入-规约冲突		
关键问题	如何正确选择候选式	如何正确识别句柄 (每次应该归约的符号串)		

[移入-规约:是句柄就规约,不是就移入规约-规约:哪个是句柄就用哪个规约

如何正确地识别句柄? -- LR分析法

- > 基本原理
 - > 句柄是逐步形成的,用"状态"表示句柄识别的进展程度,状态是用项目来描述的
 - \triangleright 例: $S \rightarrow bBB$

$$S \rightarrow \cdot bBB \longrightarrow$$
 移进项目 $S \rightarrow b \cdot BB \longrightarrow$ 待约项目 $S \rightarrow bBB \cdot \longleftarrow$ 归约项目

项目描述了句柄识别的状态

LR分析器基于由项目集表示的状态构造自动机进行句柄的识别

等价项目

(2) $S \rightarrow vI:T$ \bigcirc $I \rightarrow I, i$ $S' \rightarrow S$ (4) $I \rightarrow i$ $T \rightarrow r$ (2) $S \rightarrow vI:T$ (3) $S \rightarrow v \cdot I : T$ (7) $I \rightarrow I$, i (8) $I \rightarrow I$, i (4) $S \rightarrow vI : T$ (5) $S \rightarrow vI: T$ $(0) S' \rightarrow \cdot S$ (9) $I \rightarrow I$, i $(11) I \rightarrow i$ (13) $T \rightarrow r$ (6) $S \rightarrow vI: T$ (14) $T \rightarrow r$ $(10)I \rightarrow I, i$ $(1) S' \rightarrow S$ $(12) I \rightarrow i$

把等价的项目组成一个项目集1,项目集作为状态来构造自动机

LR 分析器(自动机)的总体结构

输入缓冲区 状态/符号栈 ➡产生式序列 LR主控程序 S_{m-1}

 S_{m}

动作表

转移表

分析表

LR 分析表的结构

- 〉例
 - 〉文法
 - ① $S \rightarrow BB$
 - ② $B \rightarrow aB$
 - $\bigcirc B \rightarrow b$

sn: 将符号a、A	状态n 压入栈
------------	---------

rn: 用第n个产生式进行归约

北大	ACTION			GOTO	
状态	a	b	\$	S	В
0	s 3	s4		1	2
1			acc		
2	s 3	s4			5
3	s 3	s4			6
4	r3	r3	r3		
5	r1	r1	r1		
6	r2	r2	r2		

$$S_i \xrightarrow{a(8\lambda)} S_j$$

如何构造给定文法的LR分析表?

- ► LR(0)分析
- >SLR分析
- ►LR(1)分析
- ►LALR分析

LR(0)分析表构造算法

- 构造G'的规范LR(0)项集族 $C = \{ I_0, I_1, \ldots, I_n \}$
- \triangleright 令 I_i 对应状态i。状态i的语法分析动作按照下面的方法决定:
 - \triangleright if $A \rightarrow \alpha \cdot a\beta \in I_i$ and $GOTO(I_i, a) = I_i$ then ACTION[i, a] = sj
 - \triangleright if $A \rightarrow \alpha.B\beta \in I_i$ and $GOTO(I_i, B) = I_j$ then GOTO[i, B] = j
 - ightharpoonup if $A
 ightharpoonup \alpha \cdot \in I_i$ 且 $A \neq S'$ then for $\forall a \in V_T \cup \{\$\}$ do ACTION[i, a] = rj (j是产生式 $A
 ightharpoonup \alpha$ 的编号)
 - $\triangleright if S' \rightarrow S \in I_i then ACTION[i, \$] = acc$
- ▶没有定义的所有条目都设置为"error"

各种LR分析表构造方法的不同之处在于规约项目的处理上

$$\triangleright if [A \rightarrow \alpha \cdot, c] \in I_i$$

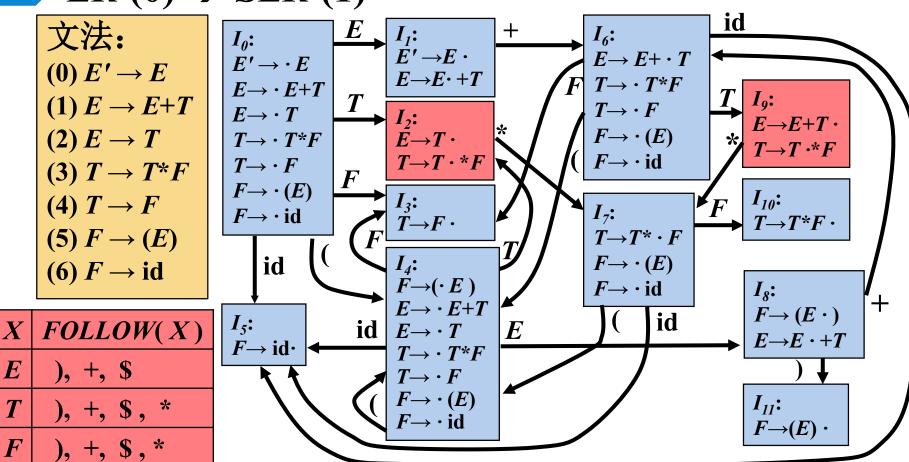
$$\begin{cases} A = S' & ACTION[i, \$] = acc \\ A \neq S' & LR(0) & for \forall a \in V_T \cup \{\$\} & do \ ACTION[i, a] = r_j \ (j \neq j \neq j) \end{cases}$$

$$\begin{cases} LR(1) & for \forall a \in FOLLOW(A) & do \ ACTION[i, a] = r_j \\ LR(1) & 精准规约 & ACTION[i, a] = r_j \end{cases}$$

LR(1)分析实际上是根据后继符集合的不同, 将原始的LR(0)状态分裂成不同的LR(1)状态

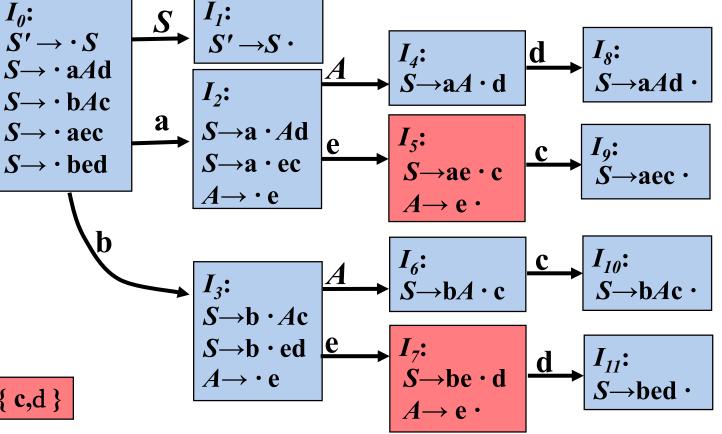
$LR(0) \rightarrow SLR(1)$

例: LR(0)分析过程中的冲突



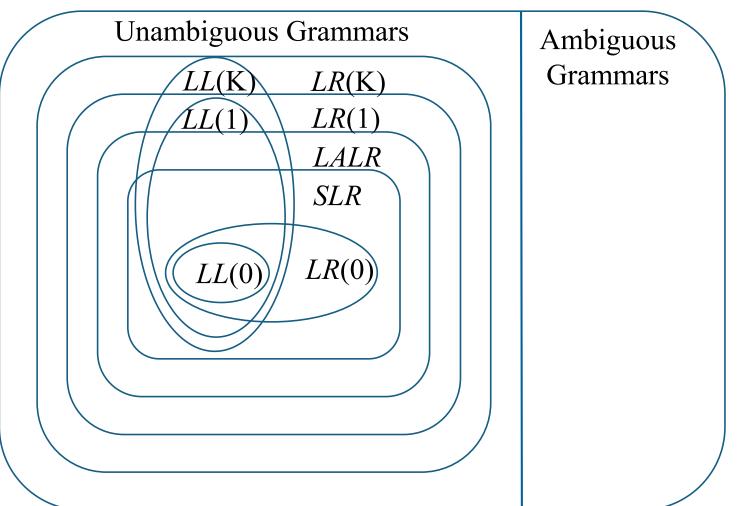
$SLR(1) \rightarrow LR(1)$

- $0) S' \rightarrow S$
- 1) $S \rightarrow aAd$
- 2) $S \rightarrow bAc$
- 3) $S \rightarrow aec$
- 4) $S \rightarrow bed$
- **5)** *A*→**e**



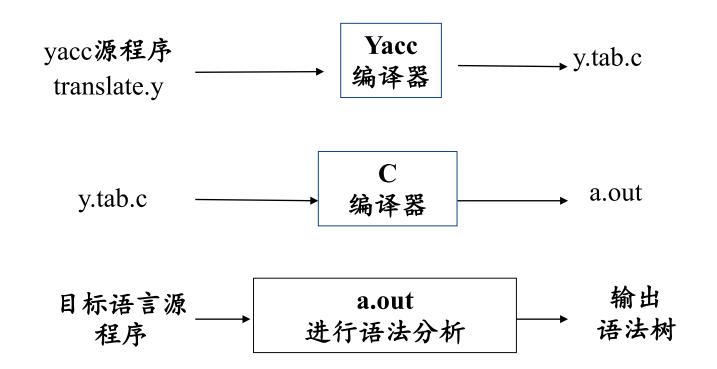
 $FOLLOW(A) = \{ c,d \}$

$LR(1) \rightarrow LALR$ $0) S' \rightarrow S$ 1) $S \rightarrow BB$ I_1 : S $S' \rightarrow S'$, \$ 2) $B \rightarrow aB$ I_5 : I_{θ} : 3) $B \rightarrow b$ $S' \rightarrow \cdot S$, \$ $S \rightarrow BB \cdot , \$$ В I_2 : $S \rightarrow BB$, \$ $B \rightarrow a B$, a/b $S \rightarrow B \cdot B$, \$ I_6 : $B \rightarrow b$, a/b $B \rightarrow aB$, \$ $B \rightarrow a \cdot B$, \$ $B \rightarrow b$, \$ I_1 : $B\rightarrow aB$, \$ $S' \rightarrow S'$, \$ I_{θ} : $B \rightarrow b$, \$ I_5 $S' \rightarrow \cdot S$, \$ I_4 : I_7 : I_3 : $S \rightarrow BB \cdot, \$$ $S \rightarrow BB$, \$ I_2 : $B \rightarrow b \cdot , a/b$ $B \rightarrow b \cdot , \$$ $B \rightarrow a B$, a/b $B \rightarrow a \cdot B$, a/b I_{g} : $S \rightarrow B \cdot B$, \$ b $B \rightarrow b$, a/b $B \rightarrow aB \cdot, \$$ $B \rightarrow aB$, a/b $B \rightarrow aB$, \$ I_8 : $B \rightarrow b$, a/b $B \rightarrow aB \cdot , a/b$ $B \rightarrow b$, \$ a I_{47} : I_{36} : $B \rightarrow b \cdot , a/b/$ $B \rightarrow a \cdot B$, a/b/\$ $B \rightarrow aB$, a/b/\$ $B \rightarrow aB \cdot , a/b/$ \$ $B \rightarrow b$, a/b/\$



文法类的谱系

一. 用生成器Yacc构造翻译器的过程



二. Yacc源程序有三部分组成 声明 %% 翻译规则 %% C写的子程序

```
%{
# include <ctype.h>
%}
% token digit
%%
line
      : expr'\n' {printf("%d\n",$1);}
       : expr '+' term {$$=$1+$3;}
expr
        term
```

```
: term '*' facter {$$=$1*$3;}
term
         : facter
facter : '('expr')' {$$=$2;}
         : digit
%%
yylex(){ int c;
         c=getchar();
         if (isdigit( c ){ yylval=c- '0';
                         return digit; }
          return c;
```

声明部分

有任选的两节。

第一节处于分界符%{和%}之间,它是一些普通的C的声明;第二节是文法记号的声明。

翻译规则部分 每条翻译规则由一个文法产生式和有关的语义动作组成。

支持例程部分 一些C写的子程序。例:词法分析器,错误恢复例程等。

1. 文法G[S]为:

 $S \rightarrow AB$

A→aBa | ε

 $B \rightarrow bAb \mid \epsilon$

该文法是SLR(1)文法吗?若 是,构造其分析表并给出输 入串baab\$的分析过程。

2. 若有文法G[S]:

 $S \rightarrow S; M \mid M$

 $M \rightarrow MbD \mid D$

 $D \rightarrow D(S) \mid \varepsilon$

给出G[S]的LR(1)项目集 规范族中的 I_0 。

3. 给定文法G[S]:

 $S \rightarrow AdD \mid \epsilon$

 $A \rightarrow aAd \mid \varepsilon$

 $D \rightarrow DdA \mid b \mid \epsilon$

(1)证明G[S]不是LR(0)和 SLR(1)文法。

(2)判断G[S]是LR(1)和 LALR(1)文法,并构造相应 的分析表。