编译系统

第四章
语法分析

哈尔滨工业大学 陈鄞

第7讲（语法分析_4）要点

例：LR(0) 分析过程中的冲突

文法：

(0) $E' \rightarrow E$

(1) $E \rightarrow E+T$

(2) $E \rightarrow T$

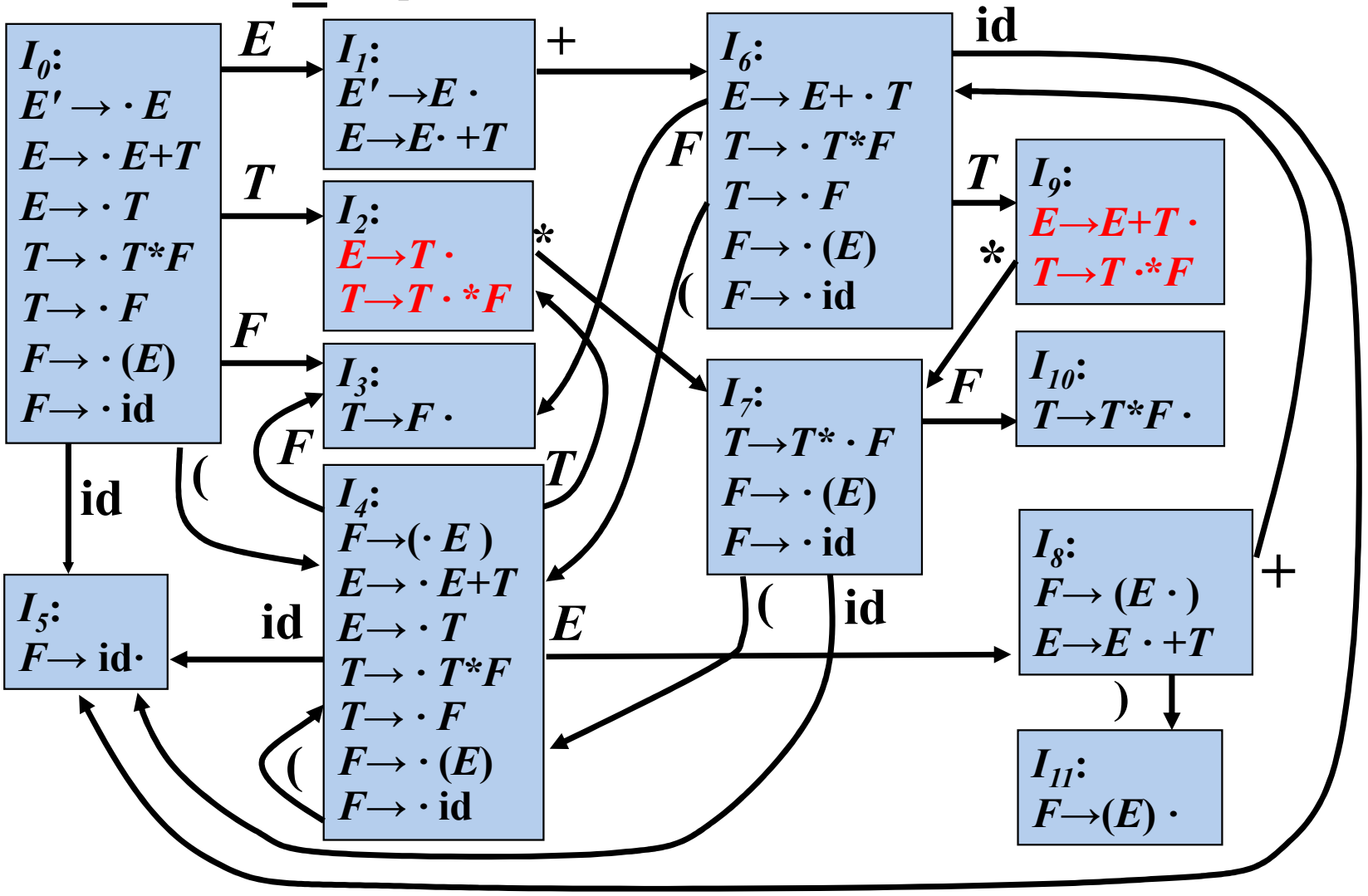
(3) $T \rightarrow T*F$

(4) $T \rightarrow F$

(5) $F \rightarrow (E)$

(6) $F \rightarrow \text{id}$

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



第7讲（语法分析_4）要点

- 句柄都是相对一个句型而言的
- LR(0)分析只关心构成某个成分*A*的各个子成分是否都已出现（如果都出现了就进行归约），但并没有考虑这个成分*A*是否是待分析句子中的一个成分
  - 事实上，如果*A*不是待分析句子中的一个成分，那么即使把它归约出来也是徒劳，在后续分析过程中迟早会发现分析进行不下去的
- 因此应该将句柄的识别放在句型这样一个上下文环境中考虑
- 受技术上的限制，考虑整个上下文实现起来很困难，因此，LR(1)分析只考虑成分*A*下文的第1个（终结）符号

规范*LR*(1)项目

- 将一般形式为 [$A \to \alpha \cdot \beta$, a]的项称为 ***LR*(1) 项**，其中$A \to \alpha\beta$ 是一个产生式，**a** 是一个终结符(这里将$视为一个特殊的终结符)它表示在当前状态下，*A*后面必须紧跟的终结符，称为该项的展望符(*lookahead*)
  - 在形如[$A \to \alpha \cdot \beta$, a]且$\beta \neq \varepsilon$的项中，展望符a没有任何作用
  - 但是一个形如[$A \to \alpha \cdot$, a]的项在只有在下一个输入符号等于a时才可以按照$A \to \alpha$ 进行归约
    - 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

各种LR分析表构造方法的不同之处在于归约项目的处理上

- *if* $[A \to \alpha \cdot, c] \in I_i$

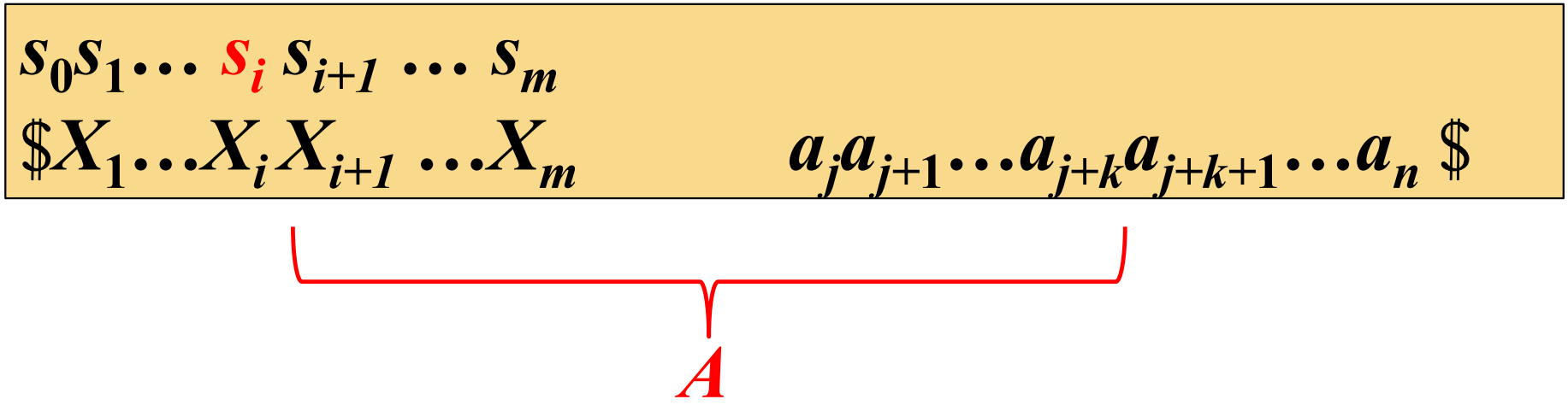
$A = S'$ $\quad ACTION[\, i, \$\,] = acc$

$A \neq S'$

- *LR*(0) $\quad for\ \forall a \in V_T \cup \{\$\} \quad do\ ACTION[\, i, a\,] = r_j$ (j是产生式$A \to \alpha$的编号)
- *SLR*(1) $\quad for\ \forall a \in FOLLOW(A)\ do\ ACTION[\, i, a\,] = r_j$
- *LR*(1) $\quad$ 精准归约 $\quad ACTION[\, i, c\,] = r_j$

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

恐慌模式错误恢复



- 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。
- 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

结束