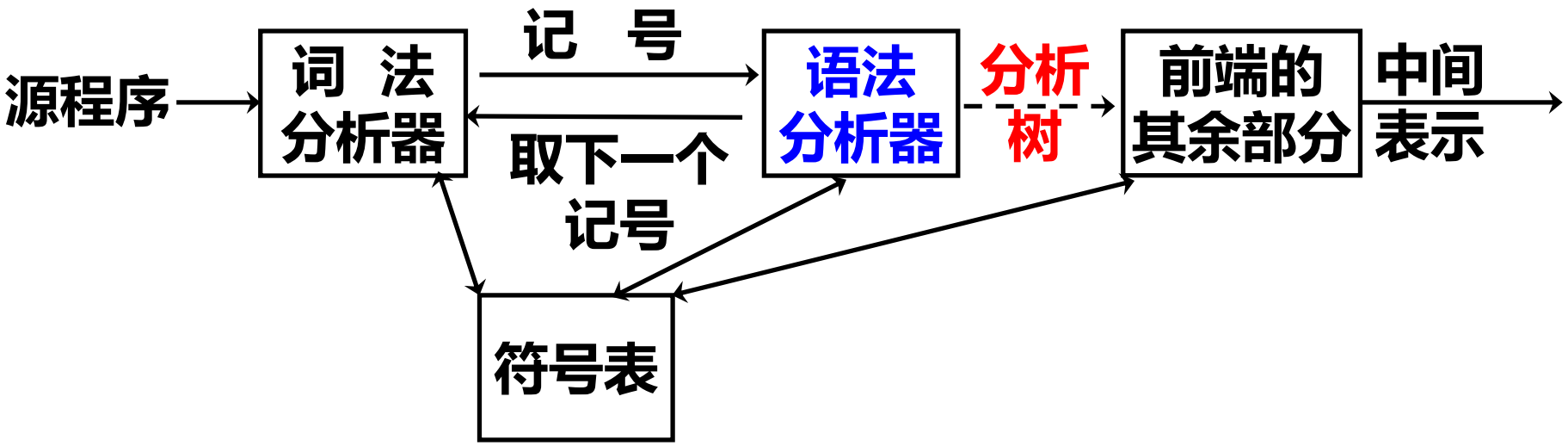
中国科学技术大学
University of Science and Technology of China

USTC, CHINA
ADSLAB
先进数据系统实验室

《编译原理与技术》
语法分析 II

计算机科学与技术学院
李 诚
20/09/2018

主要内容



❑语法分析方法概述

- ❖自顶向下与自底向上方法的区别
- ❖自顶向下分析方法
  - ➢递归下降预测分析方法
  - ➢消除左递归、提取左公因子
  - ➢LL(1)文法及非递归预测分析方法

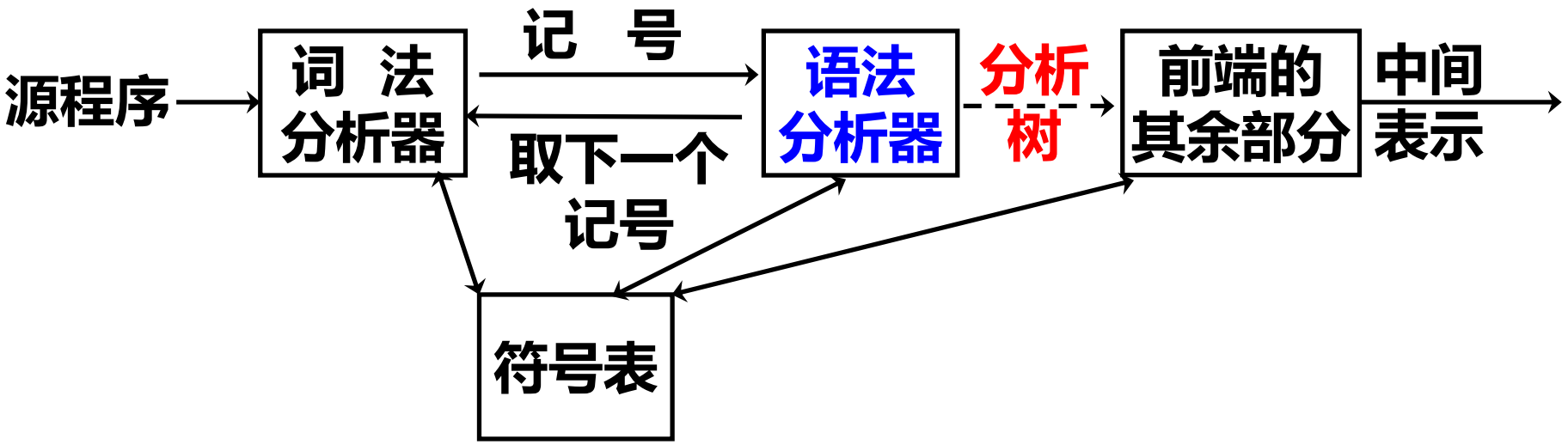
语法分析的主要方法

- 自顶向下（Top-down）
  - 针对输入串，从文法的开始符号出发，尝试根据产生式规则推导（derive）出该输入串。
- 自底向上（Bottom-up）
  - 针对输入串，尝试根据产生式规则规约（reduce）到文法的开始符号。

语法分析的主要方法

- 自顶向下（Top-down）
  - 针对输入串，从文法的开始符号出发，尝试根据产生式规则推导（derive）出该输入串。
  - 分析树的构造方法
    - 从根部开始
- 自底向上（Bottom-up）
  - 针对输入串，尝试根据产生式规则规约（reduce）到文法的开始符号。
  - 分析树的构造方法：
    - 从叶子开始

主要内容



❑语法分析方法概述

- ❖自顶向下与自底向上方法的区别
- ❖自顶向下分析方法
  - ➢递归下降分析方法
  - ➢消除左递归、提取左公因子
  - ➢LL(1)文法及非递归预测分析方法

递归下降分析法

- Recursive Descent Parsing
- 考虑以下文法

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

$$T \rightarrow \text{int} \mid \text{int} * T \mid (E)$$

- 输入串：(int_5)
- 从左到右扫描输入串
- 从开始非终结符E开始
  - 按顺序尝试E的产生式

递归下降分析法

$E \rightarrow T \mid T + E$

$T \rightarrow \mathbf{int} \mid \mathbf{int} * T \mid (E)$

E

(int_5)

↑

递归下降分析法

$E \rightarrow T \mid T + E$

$T \rightarrow \text{int} \mid \text{int} * T \mid (E)$

E
|
T

(int_5)
↑

递归下降分析法

$E \rightarrow T \mid T + E$

$T \rightarrow \mathbf{int} \mid \mathbf{int} * T \mid (E)$

E
|
T
|
int

不匹配
请回溯

(int_5)
↑

递归下降分析法

$E \rightarrow T \mid T + E$

$T \rightarrow \text{int} \mid \text{int} * T \mid (E)$

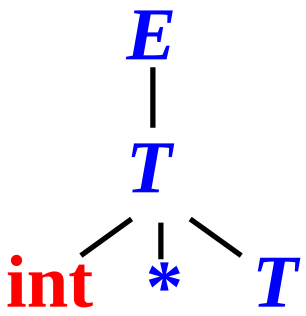
E
|
T

(int_5)
↑

递归下降分析法

$E \rightarrow T \mid T + E$

$T \rightarrow \text{int} \mid \text{int} * T \mid (E)$



不匹配
请回溯

(int_5)

递归下降分析法

$E \rightarrow T \mid T + E$

$T \rightarrow \text{int} \mid \text{int} * T \mid (E)$

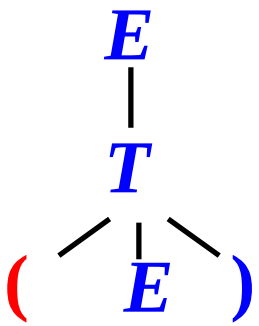
E
|
T

(int_5)
↑

递归下降分析法

$E \rightarrow T \mid T + E$

$T \rightarrow \text{int} \mid \text{int} * T \mid (E)$



匹配
箭头前进

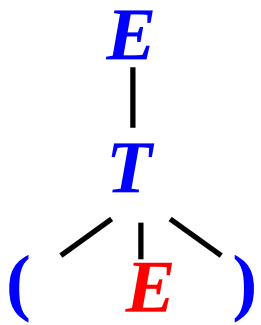
(int_5)

↑

递归下降分析法

$E \rightarrow T \mid T + E$

$T \rightarrow \text{int} \mid \text{int} * T \mid (E)$

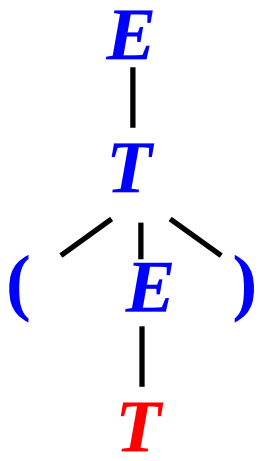


(int_5)

递归下降分析法

$E \rightarrow T \mid T + E$

$T \rightarrow \text{int} \mid \text{int} * T \mid (E)$



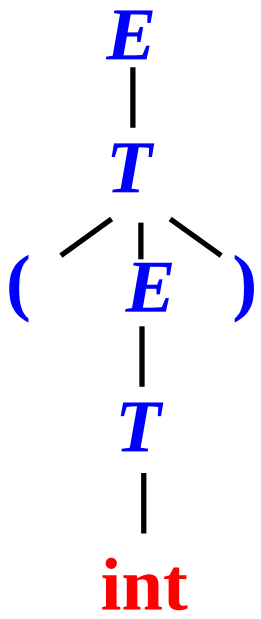
(int_5)

↑

递归下降分析法

$E \rightarrow T \mid T + E$

$T \rightarrow \mathbf{int} \mid \mathbf{int} * T \mid (E)$



(int_5)

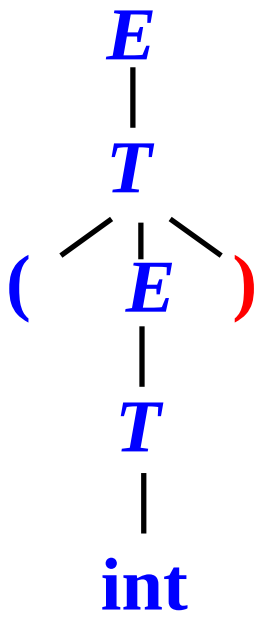
↑

匹配
箭头前进

递归下降分析法

$E \rightarrow T \mid T + E$

$T \rightarrow \text{int} \mid \text{int} * T \mid (E)$



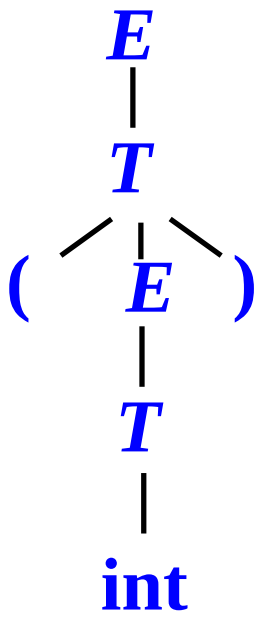
匹配
箭头前进

(int_5)

递归下降分析法

$E \rightarrow T \mid T + E$

$T \rightarrow \text{int} \mid \text{int} * T \mid (E)$



(int_5)

↑

分析完毕
接受该串

递归下降分析法

- 递归下降的预测分析
  - 为每一个非终结符写一个分析过程
  - 这些过程可能是递归的
- 例

type → *simple*
| ↑ id
| array [*simple*] of *type*
simple → integer
| char
| num dotdot num

一个辅助过程

```
void match (terminal t) {
        if (lookahead == t) lookahead =
  nextToken( );
        else error( );
}
```

递归下降分析法


```
void type( ) {
  if ( (lookahead == integer) || (lookahead == char) ||
                                  (lookahead == num) )
        simple( );
 else if ( lookahead == '↑' ) { match('↑'); match(id);}
  else if (lookahead == array) {
        match(array); match( '[' ); simple( );
        match( ']' ); match(of ); type( );
  }
  else error( );
}
```

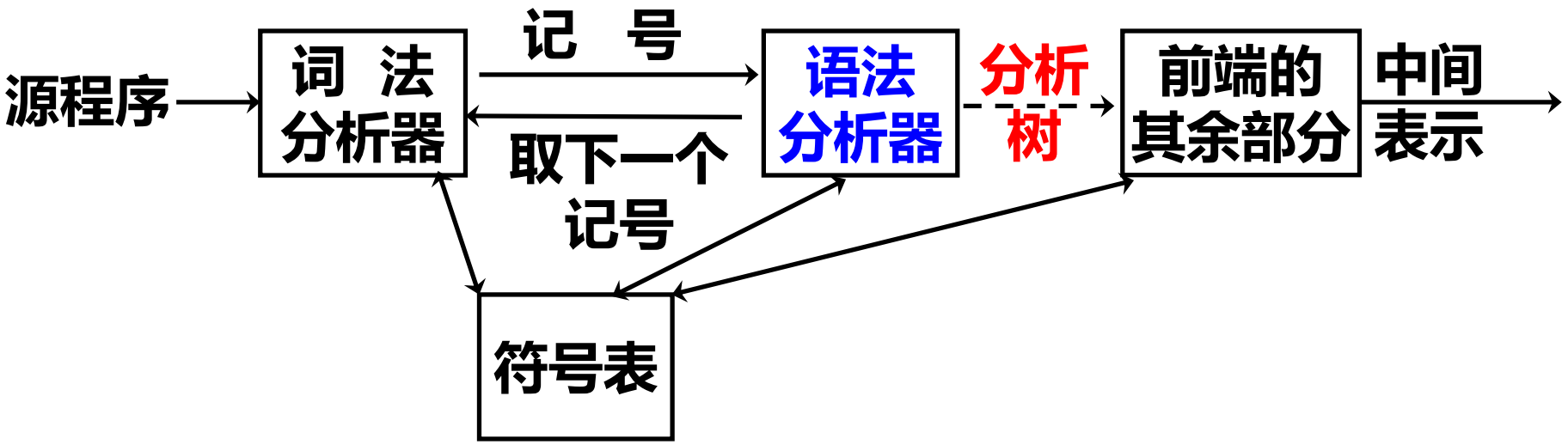
type → simple
| ↑ id
| array [simple] of type

递归下降分析法


```
void simple( ) {
  if ( lookahead == integer) match(integer);
  else if (lookahead == char) match(char);
  else if (lookahead == num) {
        match(num); match(dotdot); match(num);
  }
  else error( );
}
```

simple → integer
| char
| num dotdot num

主要内容



❑语法分析方法概述

- ❖自顶向下与自底向上方法的区别
- ❖自顶向下分析方法
  - ➢递归下降分析方法
  - ➢消除左递归、提取左公因子
  - ➢LL(1)文法及非递归预测分析方法

递归下降的问题1

- 可能进入无限循环
- 考虑以下文法

$$S \rightarrow Sa$$

- 该文法是左递归的(left-recursive)

递归下降的问题1

- 可能进入无限循环
- 考虑以下文法

$$S \rightarrow Sa$$

- 该文法是左递归的(left-recursive)
- 自顶向下分析方法无法处理左递归
  - Why?

消除左递归

❑文法左递归　　$A \Rightarrow^{+} A\alpha$

❑直接左递归

$A \rightarrow A\alpha \mid \beta$，其中$\alpha$，$\beta$不以A开头

❖串的特点 $\beta\alpha \ldots \alpha$

❑消除直接左递归

$A \rightarrow \beta A'$

$A' \rightarrow \alpha A' \mid \varepsilon$

消除左递归

❑例 算术表达文法

$$E \to E + T \mid T$$
$$T \to T * F \mid F$$
$$F \to (E) \mid \mathrm{id}$$

$(T + T \ldots + T)$

$(F * F \ldots * F)$

❑消除左递归后文法

$$E \to TE'$$
$$E' \to + TE' \mid \varepsilon$$
$$T \to FT'$$
$$T' \to * F T' \mid \varepsilon$$
$$F \to (E) \mid \mathrm{id}$$

消除间接左递归

- 非直接左递归

$$S \rightarrow Aa \mid b$$
$$A \rightarrow Sd \mid \varepsilon$$

消除间接左递归

- 非直接左递归

$$S \rightarrow Aa \mid b$$
$$A \rightarrow Sd \mid \varepsilon$$

- 先变换成直接左递归

$$S \rightarrow Aa \mid b$$
$$A \rightarrow Aad \mid bd \mid \varepsilon$$

- 再消除左递归

$$S \rightarrow Aa \mid b$$
$$A \rightarrow bdA' \mid A'$$
$$A' \rightarrow adA' \mid \varepsilon$$

递归下降的问题2

- 复杂的回溯→代价太高
  - 非终结符有可能有多个产生式
  - 由于信息缺失，无法准确预测选择哪一个

- 有左因子的(left -factored)文法：
  - $A \rightarrow \alpha\beta_1 \mid \alpha\beta_2$

提左因子(left factoring)

- 推后选择产生式的时机，以便获取更多信息

$A \to \alpha\beta_1 \mid \alpha\beta_2$ 等价于

$$A \to \alpha A'$$
$$A' \to \beta_1 \mid \beta_2$$

提左因子(left factoring)

❑例 悬空*else*的文法

stmt → if *expr* then *stmt* else *stmt*
| if *expr* then *stmt*
| other

提左因子

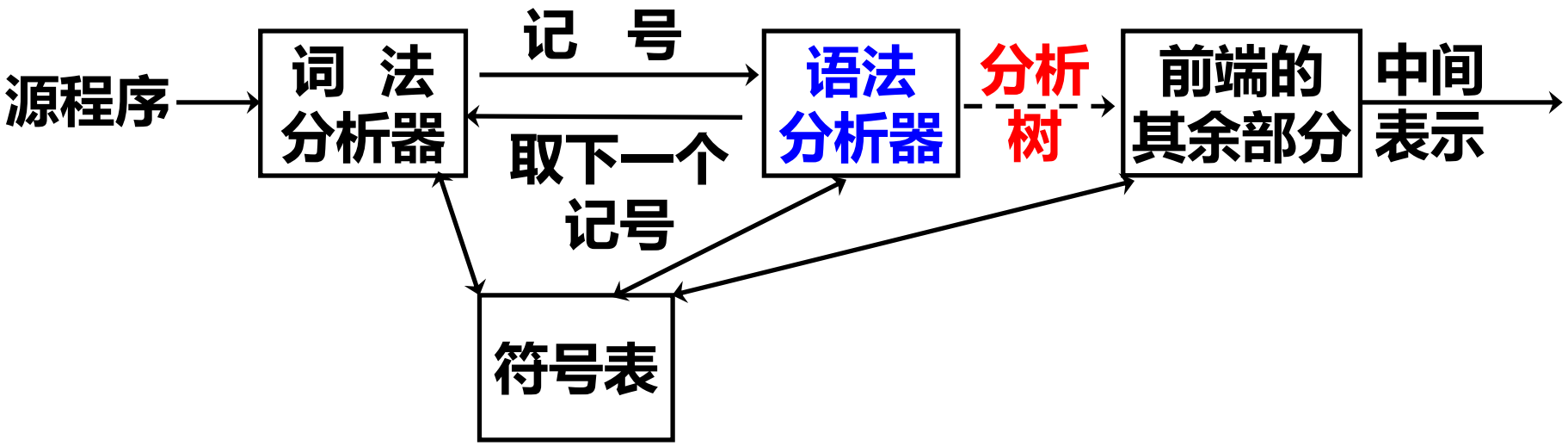
stmt → if *expr* then *stmt optional_else_part*
| other

optional_else_part → else *stmt*
| ε

关于递归下降法的总结

- 简单、一般化的语法分析方法
  - 但左递归必须先消除
  - 可自动化地消除左递归
- 因为回溯而不受欢迎
  - 效率太低
- 因此，在构造编译器时，可通过对文法加以限制来避免回溯

主要内容



❑语法分析方法概述

- ❖自顶向下与自底向上方法的区别
- ❖自顶向下分析方法
  - ➢递归下降分析方法
  - ➢消除左递归、提取左公因子
  - ➢LL(1)文法及非递归预测分析方法

预测分析法

- Predictive parsing

- 与递归下降法相似，但
  - 没有回溯
  - 通过向前看一些记号来预测需要用到的产生式

- 此方法接受LL(k)文法
  - L-means “left-to-right” scan of input
  - L-means “leftmost derivation”
  - k-means “predict based on k tokens of lookahead”
  - In practice, LL(1) is used

LL(1) vs. 递归下降

- 每一步，递归下降有若干产生式需要尝试，而LL(1)仅有一个产生式可以选择
- 递归下降有回溯，而LL(1)没有
- 因此，LL(1)文法不可能是左递归和二义的

LL(1)文法

- 对文法加什么样的限制可以保证没有回溯?

- 先定义两个和文法有关的函数
  - FIRST(α) = $\{a \mid \alpha \Rightarrow^* a\ldots, a \in V_T\}$
    特别是，$\alpha \Rightarrow^* \varepsilon$时，规定$\varepsilon \in$ FIRST(α)
  - FOLLOW(A) = $\{a \mid S \Rightarrow^* \ldots Aa\ldots, a \in V_T\}$
    如果A是某个句型的最右符号，那么\$属于FOLLOW(A)

LL(1)文法：FIRST(X)

- 计算FIRST(X), $X \in V_T \cup V_N$
  - $X \in V_T$, FIRST(X) = {X}
  - $X \in V_N$且 $X \to \varepsilon$
    则将 ε加入到FIRST(X)
  - $X \in V_N$ 且 $X \to Y_1 Y_2 \ldots Y_k$
    - 如果 $a \in$ FIRST(Y_i)且ε在FIRST(Y_1), …, FIRST(Y_{i-1})中，则将 a加入到FIRST(X)
    - 如果 ε 在FIRST(Y_1), …, FIRST(Y_k)中，则将ε加入到FIRST(X)

LL(1)文法：FOLLOW(A)

- 计算FOLLOW(A), $A \in V_N$
  - $加入到FOLLOW($A$)，当A是开始符号
  - 如果$A \to \alpha B\beta$，则FIRST(β)-{ε}加入到FOLLOW(B)
  - 如果$A \to \alpha B$或$A \to \alpha B\beta$且$\varepsilon \in$ FIRST(β)，则FOLLOW(A)加入到FOLLOW(B)

LL(1)文法

❑LL(1)文法的定义

任何两个产生式$A \to \alpha \mid \beta$都满足下列条件：

- ❖ $\mathbf{FIRST}(\alpha) \cap \mathbf{FIRST}(\beta) = \varnothing$
- ❖若$\beta \Rightarrow^* \varepsilon$，那么$\mathbf{FIRST}(\alpha) \cap \mathbf{FOLLOW}(A) = \varnothing$

❑例如，考虑下面文法

面临a…时，第2步推导不知用哪个产生式

$S \to A\ B$

$A \to a\ b \mid \varepsilon$　　$a \in \mathbf{FIRST}(ab) \cap \mathbf{FOLLOW}(A)$

$B \to a\ C$

$C \to \ldots$

LL(1)文法

LL(1)文法的定义

任何两个产生式$A \to \alpha \mid \beta$都满足下列条件：

- $\mathbf{FIRST}(\alpha) \cap \mathbf{FIRST}(\beta) = \varnothing$
- 若$\beta \Rightarrow^* \varepsilon$，那么$\mathbf{FIRST}(\alpha) \cap \mathbf{FOLLOW}(A) = \varnothing$

LL(1)文法有一些明显的性质

- 没有公共左因子
- 不是二义的
- 不含左递归

表达式文法：无左递归的

❑例 $E \to TE'$

$E' \to +TE' \mid \varepsilon$

$T \to FT'$

$T' \to *FT' \mid \varepsilon$

$F \to (E) \mid \mathrm{id}$

FIRST(E) = FIRST(T) = FIRST(F) = { (, id }

FIRST(E') = {+, ε}

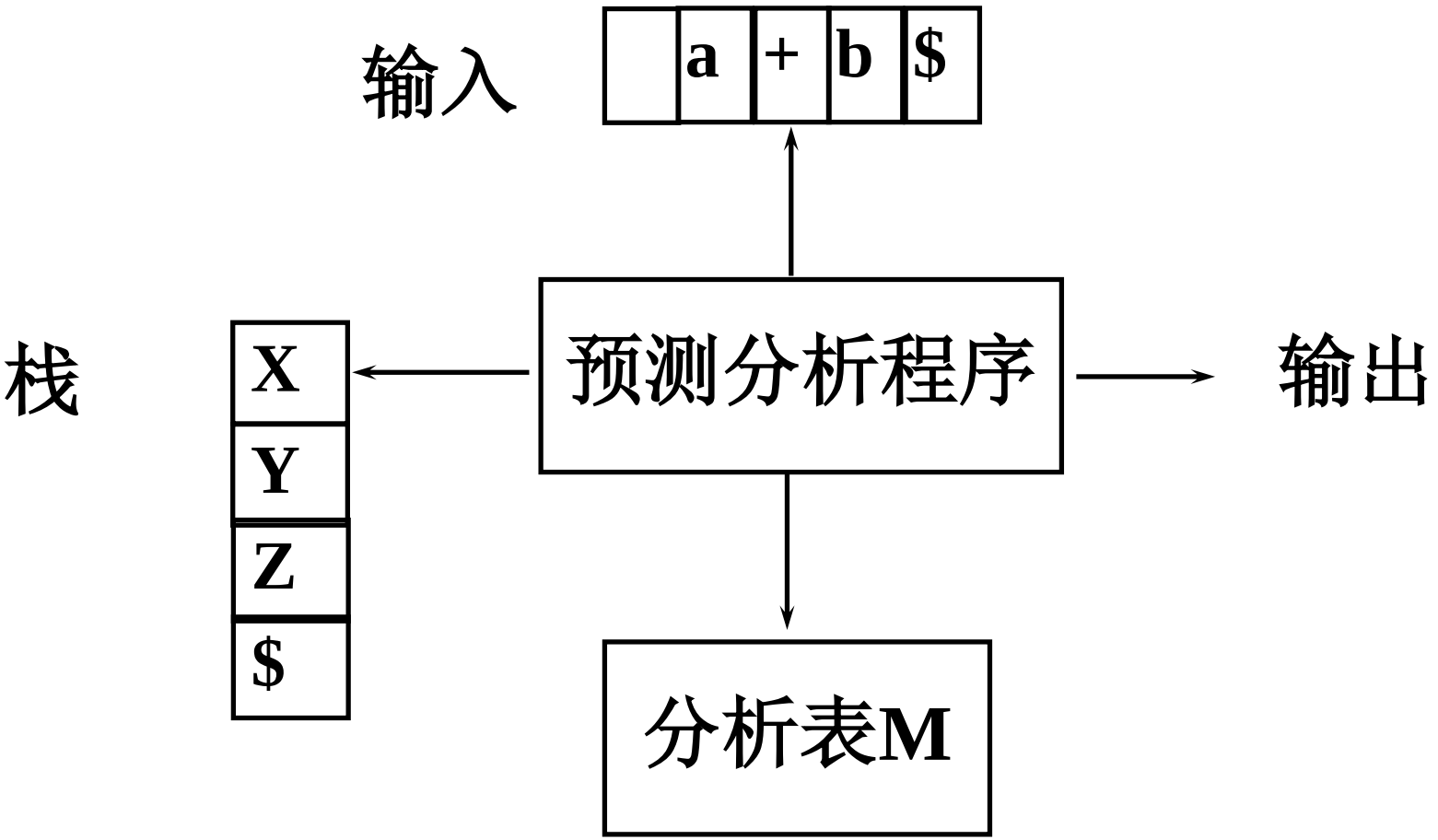
FRIST(T') = {*, ε}

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

FOLLOW(T) = FOLLOW (T') = {+,), $}

FOLLOW(F) = {+, *,), $}

非递归的预测分析



预测分析表M的构造

- 对文法的每个产生式$A \to \alpha$，执行(1)和(2)
  - （1）对FIRST(α)的每个终结符a，把$A \to \alpha$加入$M[A, a]$
  - （2）如果ε在FIRST(α)中，对FOLLOW(A)的每个终结符b（包括\$），把$A \to \alpha$加入$M[A, b]$

M中其它没有定义的条目都是error

预测分析表M的构造

❑行：非终结符；列：终结符 或$ ； 单元：产生式

非终结符	输 入 符 号					
	id	+	*	(	)	$
E	$E \rightarrow TE'$			$E \rightarrow TE'$		
E'		$E' \rightarrow +TE'$			$E' \rightarrow \varepsilon$	$E' \rightarrow \varepsilon$
T	$T \rightarrow FT'$			$T \rightarrow FT'$		
T'		$T' \rightarrow \varepsilon$	$T' \rightarrow *FT'$		$T' \rightarrow \varepsilon$	$T' \rightarrow \varepsilon$
F	$F \rightarrow$id			$F \rightarrow (E)$		

预测分析举例

预测分析器接受输入id * id + id的前一部分动作

栈	输　入	输　出
$*E*	id * id + id$	

预测分析举例

预测分析器接受输入id * id + id的前一部分动作

栈	输　入	输　出
$$E$	id * id + id$	
$$E'T$	id * id + id$	$E \rightarrow TE'$

预测分析举例

预测分析器接受输入id * id + id的前一部分动作

栈	输　入	输　出
$$E$	id * id + id$	
$$E\,'T$	id * id + id$	$E \rightarrow TE'$
$$E\,'T\,'F$	id * id + id$	$T \rightarrow FT'$

预测分析举例

预测分析器接受输入id * id + id的前一部分动作

栈	输　入	输　出
$$E	id * id + id$	
$$E'T	id * id + id$	$E \rightarrow TE'$
$$E'T'F	id * id + id$	$T \rightarrow FT'$
$$E'T' id	id * id + id$	$F \rightarrow$ id

预测分析举例

预测分析器接受输入id * id + id的前一部分动作

栈	输　入	输　出
$$E	id * id + id$	
$$E'T	id * id + id$	$E \rightarrow TE'$
$$E'T'F	id * id + id$	$T \rightarrow FT'$
$$E'T' id	id * id + id$	$F \rightarrow$ id
$$E'T'	* id + id$	

预测分析举例

预测分析器接受输入id * id + id的前一部分动作

栈	输　入	输　出
$$E$	id * id + id$	
$$E'T$	id * id + id$	$E \rightarrow TE'$
$$E'T'F$	id * id + id$	$T \rightarrow FT'$
$$E'T'$ id	id * id + id$	$F \rightarrow$ id
$$E'T'$	* id + id$	
$$E'T'F$ *	* id + id$	$T' \rightarrow *FT'$

预测分析举例

预测分析器接受输入id * id + id的前一部分动作

栈	输　入	输　出
$$E$	id * id + id$	
$$E'T$	id * id + id$	$E \rightarrow TE'$
$$E'T'F$	id * id + id$	$T \rightarrow FT'$
$$E'T'$ id	id * id + id$	$F \rightarrow$ id
$$E'T'$	* id + id$	
$$E'T'F$ *	* id + id$	$T' \rightarrow *FT'$
$$E'T'F$	id + id$	

预测分析举例

预测分析器接受输入id * id + id的前一部分动作

栈	输　入	输　出
$E	id * id + id$	
$E ′T	id * id + id$	$E \rightarrow TE'$
$E ′T ′F	id * id + id$	$T \rightarrow FT'$
$E ′T ′ id	id * id + id$	$F \rightarrow$ id
$E ′T ′	* id + id$	
$E ′T ′F *	* id + id$	$T' \rightarrow *FT'$
$E ′T ′F	id + id$	
$E ′T ′ id	id + id$	$F \rightarrow$ id

例：*stmt* → if *expr* then *stmt e_part* | other

e_part → else *stmt*| ε　　*expr* → *b*

非终结符	输入符号			
	other	*b*	else	. . .
stmt	*stmt* → other			
e_part			*e_part*→ else *stmt* *e_part* → ε	
expr		*expr* → *b*		

多重定义条目意味着文法左递归或者是二义的

多重定义的消除

例：删去*e_part* → ε，这正好满足else和近的then配对
LL(1)文法：预测分析表无多重定义的条目

非终结符	输入符号			
	other	*b*	else	. . .
stmt	*stmt* → other			
e_part			*e_part*→ else *stmt* ~~*e_part* → ε~~	
expr		*expr* → *b*		

预测分析的错误恢复

❑编译器的错误

- ❖ 词法错误，如标识符、关键字或算符的拼写错
- ❖语法错误，如算术表达式的括号不配对
- ❖语义错误，如算符作用于不相容的运算对象
- ❖逻辑错误，如无穷的递归调用

中国科学技术大学
University of Science and Technology of China
USTC, CHINA
ADSLAB
先进数据系统实验室

《编译原理与技术》
语法分析 II

任何伟大的科学发现都是不断平凡的积累！
—— 钱学森