

编译原理

2019 年 4 月 8 日

1 语法分析

1.1 自顶向下分析

1. 自顶向下分析

从分析树的顶部（根节点）向底部（叶节点）构造分析树
可以看成是从文法开始符号 S 推导出词串 w 是过程

The image shows a woman in a purple dress standing next to a large screen. She is pointing towards the screen with her right hand. The screen displays a presentation slide titled "自顶向下的分析 (Top-Down Parsing)". The slide contains the following text and diagrams:

- 从分析树的顶部（根节点）向底部（叶节点）方向构造分析树
- 可以看成是从文法开始符号 S 推导出词串 w 的过程
- 例 文法:
 - ① $E \rightarrow E + E$
 - ② $E \rightarrow E * E$
 - ③ $E \rightarrow (E)$
 - ④ $E \rightarrow id$
- 输入: $id + (id + id)$
- 分析树:

```
graph TD; E[E] --> E1[E]; E --> E2[E]; E1 --> id[id]; E1 --> P1[" "]; E2 --> E3[E]; E2 --> E4[E]; E3 --> id2[id]; E3 --> P2[" "]; E4 --> id3[id]; E4 --> P3[" "];
```
- 推导过程:

```
⇒ E ⇒ E + E
⇒ E + (E)
⇒ E + (E + E)
⇒ E + (id + E)
⇒ id + (id + E)
⇒ id + (id + id)
```

2. 最左推导

► 最左推导 (Left-most Derivation)

- 在最左推导中，总是选择每个句型的最左非终结符进行替换

例

文法
① $E \rightarrow E + E$
② $E \rightarrow E * E$
③ $E \rightarrow (E)$
④ $E \rightarrow id$
输入
$id + (id + id)$

推导过程

$$\begin{aligned}
 E &\Rightarrow_{lm} E + E \\
 &\Rightarrow_{lm} id + E \\
 &\Rightarrow_{lm} id + (E) \\
 &\Rightarrow_{lm} id + (E + E) \\
 &\Rightarrow_{lm} id + (id + E) \\
 &\Rightarrow_{lm} id + (id + id)
 \end{aligned}$$

3. 最右推导

► 最右推导 (Right-most Derivation)

- 在最右推导中，总是选择每个句型的最右非终结符进行替换

例

文法
① $E \rightarrow E + E$
② $E \rightarrow E * E$
③ $E \rightarrow (E)$
④ $E \rightarrow id$
输入
$id + (id + id)$

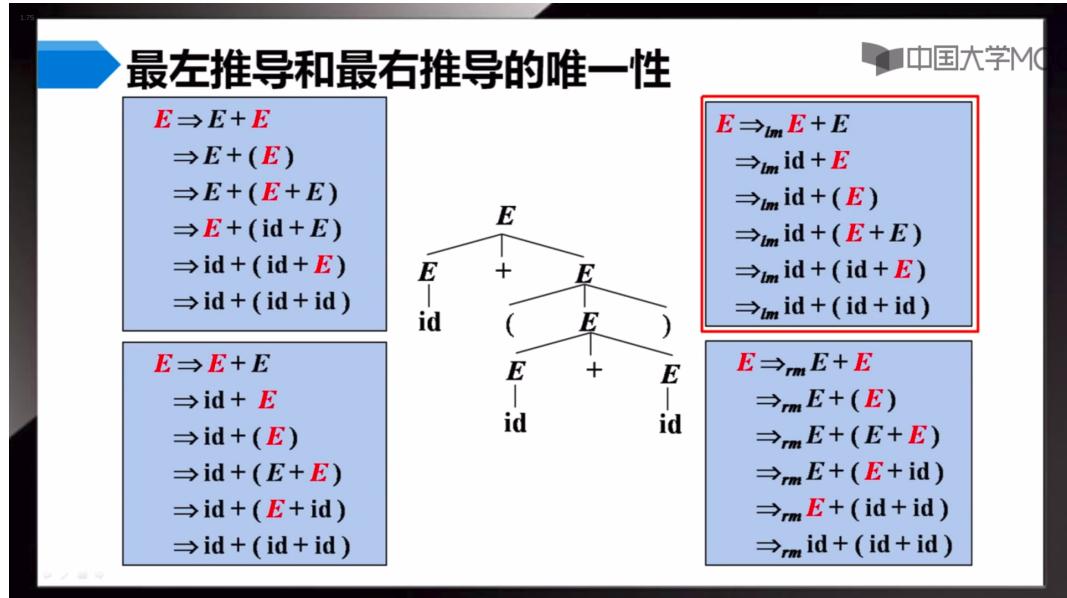
推导过程

$$\begin{aligned}
 E &\Rightarrow_{rm} E + E \\
 &\Rightarrow_{rm} E + (E) \\
 &\Rightarrow_{rm} E + (E + E) \\
 &\Rightarrow_{rm} E + (E + id) \\
 &\Rightarrow_{rm} E + (id + id) \\
 &\Rightarrow_{rm} id + (id + id)
 \end{aligned}$$

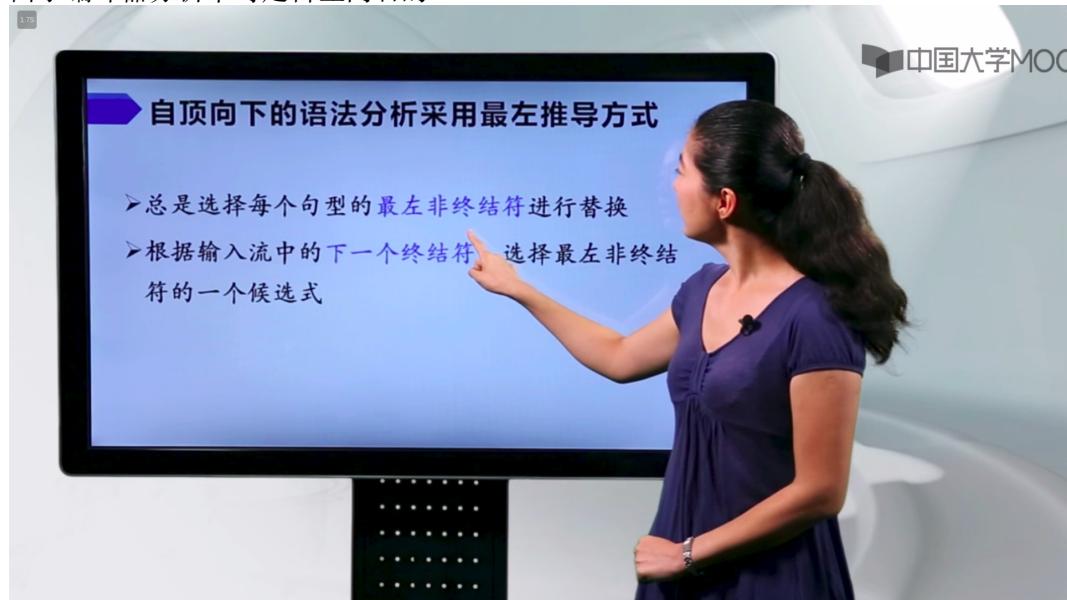
在自底向上的分析中，总是采用最左归约的方式，因此把最左归约称为规范归约，而最右推导相应地称为规范推导

4. 最左推导和最右推导的唯一性

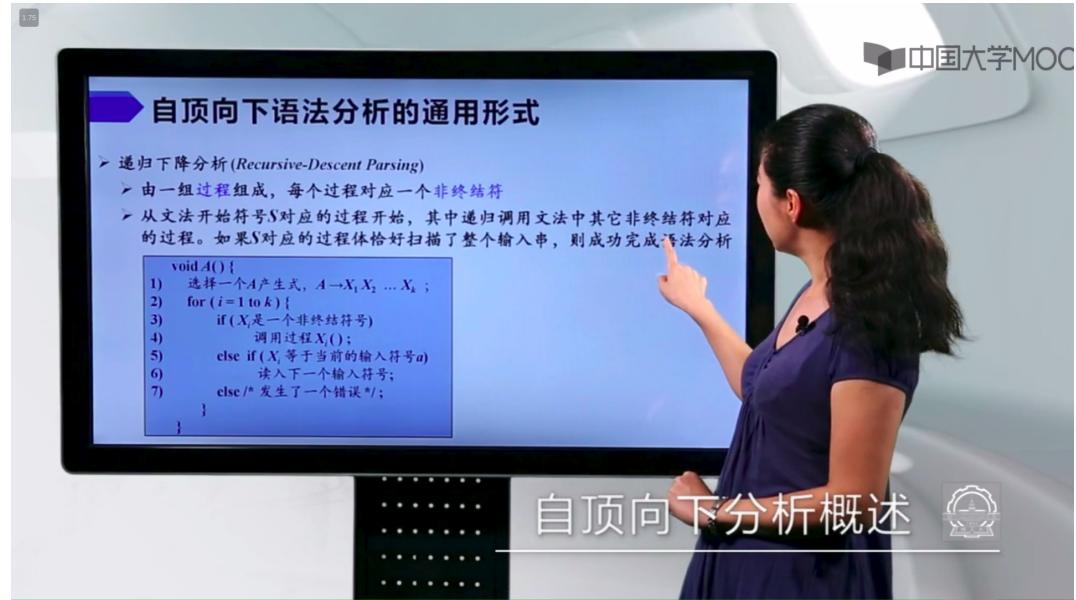
由于他们的每一步所用到的非终结符都是唯一的，所以他们推导出来的也是唯一的。



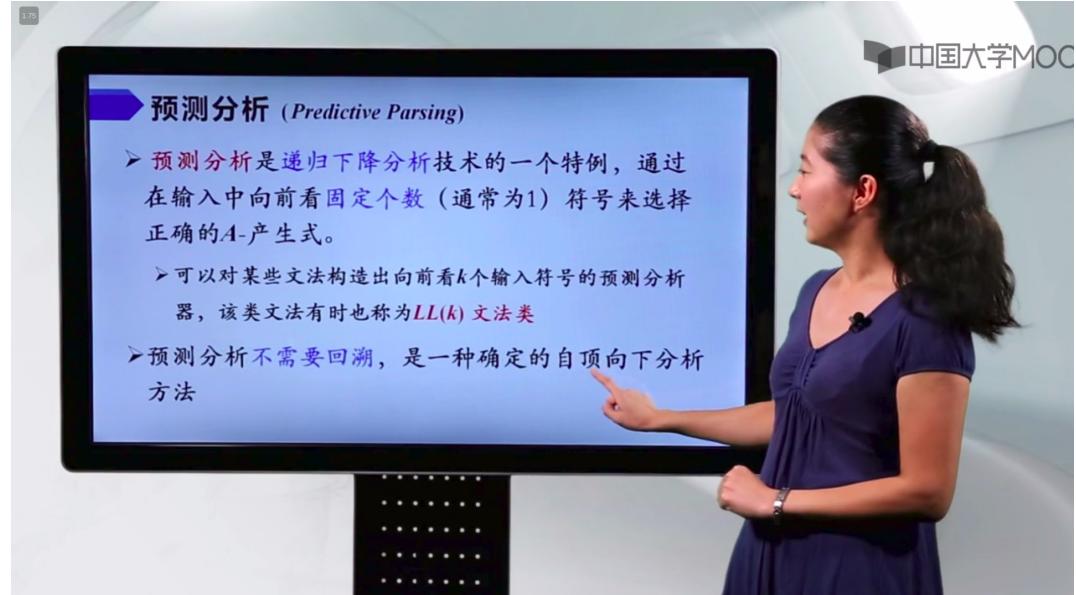
5. 自顶向下的语法分析采用最左推导方式
由于编译器分析串时是自左向右的。



6. 自顶向下语法分析的通用形式



7. 预测分析



1.2 文法转换

1. 出现的问题

The image shows a female teacher in a purple dress standing next to a large screen. The screen displays two slides from a MOOC lecture.

问题1

例 文法 G 同一非终结符的多个候选式存在
共同前缀，将导致回溯现象

$S \rightarrow aAd \mid aBe$
 $A \rightarrow c$
 $B \rightarrow b$

输入
 $a b c$
↑

问题2

例 文法 G 左递归文法会使递归下降分析器陷入无限循环

$E \rightarrow E + T \mid E - T \mid T$
 $T \rightarrow T * F \mid T / F \mid F$
 $F \rightarrow (E) \mid id$

$E \Rightarrow E + T$
 $\Rightarrow E + T + T$
 $\Rightarrow E + T + T + T$
 $\Rightarrow \dots$

输入
 $id + id * id$
↑

含有 $A \rightarrow A\alpha$ 形式产生式的文法称为是直接左递归的 (immediate left recursive)
如果一个文法中有一个非终结符 A 使得对某个串 α 在一个推导 $A \Rightarrow^* A\alpha$ ，那么这个文法就是左递归的
经过两步或两步以上推导产生的左递归称为是间接左递归的

2. 消除直接左递归

中国大学MOOC

消除直接左递归

$A \rightarrow A\alpha \mid \beta (\alpha \neq \varepsilon, \beta \text{ 不以 } A \text{ 开头})$

\downarrow

$A \rightarrow \beta A'$

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

事实上，这种消除过程就是
把左递归转换成了右递归

$A \Rightarrow A\alpha$
 $\Rightarrow A\alpha\alpha$
 $\Rightarrow A\alpha\alpha\alpha$
 \dots
 $\Rightarrow A\alpha \dots \alpha$
 $\Rightarrow \beta \alpha \dots \alpha$

$A' \Rightarrow \alpha A'$
 $\Rightarrow \alpha\alpha A'$
 $\Rightarrow \alpha\alpha\alpha A'$

$\Rightarrow \alpha \dots \alpha$

消除直接左递归的一般形式

$A \rightarrow A\alpha_1 \mid A\alpha_2 \mid \dots \mid A\alpha_n \mid \beta_1 \mid \beta_2 \mid \dots \mid \beta_m$

$(\alpha_i \neq \varepsilon, \beta_j \text{ 不以 } A \text{ 开头})$

\downarrow

$A \rightarrow \beta_1 A' \mid \beta_2 A' \mid \dots \mid \beta_m A'$

$A' \rightarrow \alpha_1 A' \mid \alpha_2 A' \mid \dots \mid \alpha_n A'$

3. 消除间接左递归

1.00

消除间接左递归

例 $S \rightarrow Aa \mid b$ $S \Rightarrow Aa$
 $A \rightarrow Ac \mid Sd \mid \varepsilon$ $\Rightarrow Sda$

将 S 的定义代入 A -产生式，得：
 $A \rightarrow Ac \mid Aa d \mid b d \mid \varepsilon$

消除 A -产生式的直接左递归，得：
 $A \rightarrow b d A' \mid A'$
 $A' \rightarrow c A' \mid a d A' \mid \varepsilon$



1.00

消除左递归算法

输入：不含循环推导（即形如 $A \Rightarrow^+ A$ 的推导）和 ε -产生式的文法 G
输出：等价的无左递归文法
方法：

```

1) 按照某个顺序将非终结符号排序为  $A_1, A_2, \dots, A_n$  .
2) for (从1到n的每个){
3)   for (从1到i-1的每个j) {
4)     将每个形如  $A_i \rightarrow A_j \gamma$  的产生式替换为产生式组  $A_i \rightarrow \delta_1 \gamma \mid \delta_2 \gamma \mid \dots \mid \delta_k \gamma$  ,
        其中  $A_j \rightarrow \delta_1 \mid \delta_2 \mid \dots \mid \delta_k$  , 是所有的  $A_j$  产生式
5)   }
6)   消除  $A_i$  产生式之间的直接左递归
7) }
```

4. 提取左公因子

1.00

提取左公因子 (Left Factoring)

例

- 文法 G
- $S \rightarrow aAd \mid aBe$
- $A \rightarrow c$
- $B \rightarrow b$

通过改写产生式来推迟决定，等读入了足够多的输入，获得足够信息后再做出正确的选择

文法 G'

- $S \rightarrow aS'$
- $S' \rightarrow Ad \mid Be$
- $A \rightarrow c$
- $B \rightarrow b$

1.00

提取左公因子算法

- 输入：文法 G
- 输出：等价的提取了左公因子的文法
- 方法：

对于每个非终结符 A ，找出它的两个或多个选项之间的最长公共前缀 α 。如果 $\alpha \neq \epsilon$ ，即存在一个非平凡的 (nontrivial) 公共前缀，那么将所有 A -产生式

$$A \rightarrow \alpha\beta_1 \mid \alpha\beta_2 \mid \dots \mid \alpha\beta_n \mid \gamma_1 \mid \gamma_2 \mid \dots \mid \gamma_m$$

替换为

$$A \rightarrow \alpha A' \mid \gamma_1 \mid \gamma_2 \mid \dots \mid \gamma_m$$

$$A' \rightarrow \beta_1 \mid \beta_2 \mid \dots \mid \beta_n$$

其中， γ_i 表示所有不以 α 开头的产生式体； A' 是一个新的非终结符。不断应用此转换，直到每个非终结符的任意两个产生式体都没有公共前缀为止。

5.