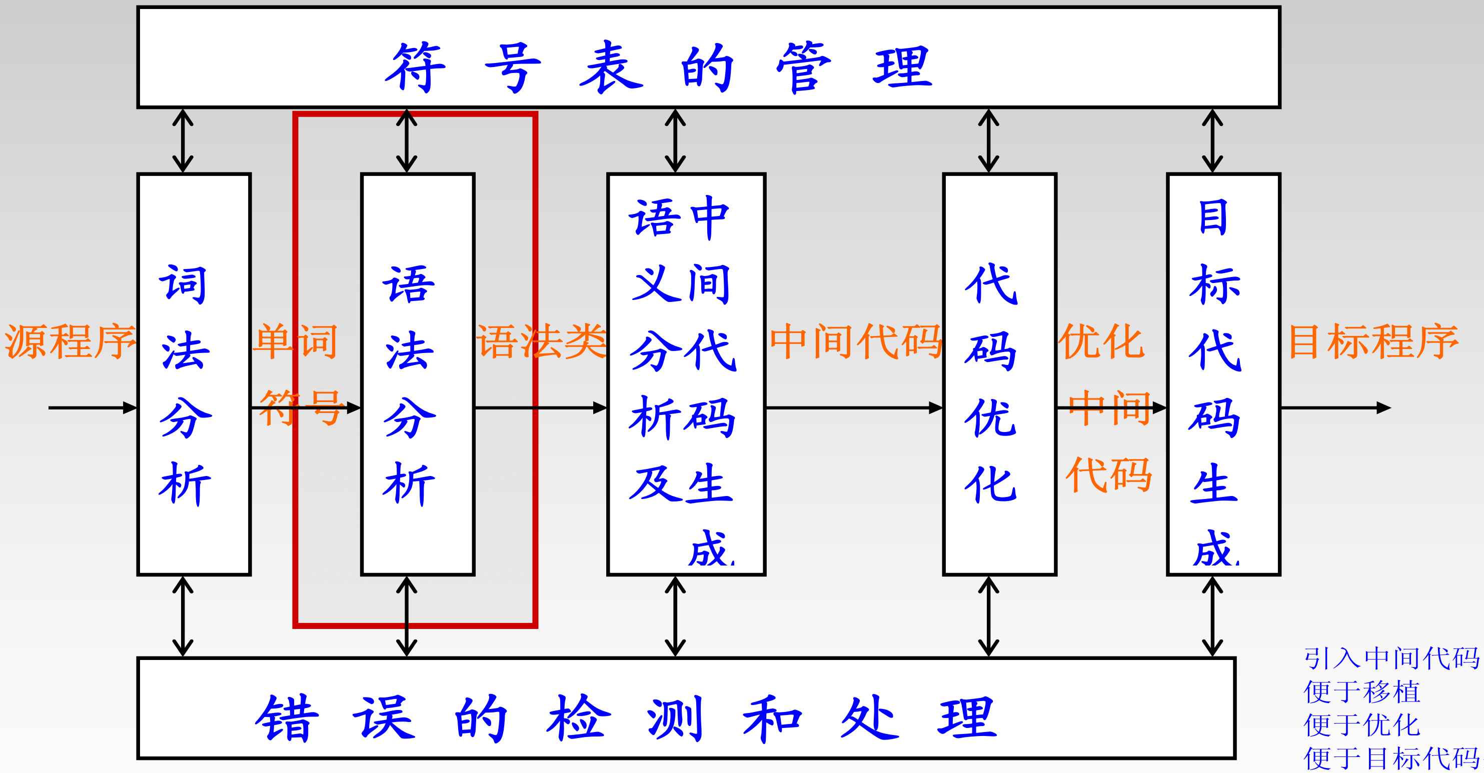
第四章　语法分析

典型编译程序的组成



引入中间代码的目的：
便于移植
便于优化
便于目标代码生成

语法分析的方法

❑自顶向下分析

- 递归下降分析
- LL（1）分析

适用于LL(1)文法

❑自底向上分析

- 简单优先分析 → 适用于简单优先文法
- 算符优先分析 → 适用于算符优先文法
- LR分析法 → 适用于LR类文法

- 效率低、代价高
  - ①虚假匹配：如上例第一次a与a匹配是虚假的；
  - ②影响语义分析：有些语法制导方式中，语义分析也得重来。

不能处理含左递归规则（直接或间接）的文法

例：$U \rightarrow Ua \mid b$

2、回朔问题

当规则中含有多个后选式时需一次次试探。

例：$S \rightarrow cAd$

$A \rightarrow ab \mid a$

匹配串cad

二、问题解决方法

1. 消除左递归（直接和间接）

(1)直接左递归的消除：

用扩充的BNF表示{ },[ ],

引入新的非终结符号（提取公因子）

(2) 间接左递归的消除

消除左递归的算法（直接和间接均消除）

（1）直接左递归的消除

例：$U \rightarrow U\alpha \mid \beta$

方法一：扩充的BNF表示

$$U \rightarrow \beta\{\alpha\}$$

方法二：引入新的非终结符

即 $U \rightarrow \beta U'$

$$U' \rightarrow \alpha U' \mid \varepsilon$$

提取公因子——使文法至多含有一个直接左递归的右部

$U \rightarrow UV_1|UV_2|\ldots|UV_n|x|y|\ldots|z$

$U \rightarrow U\underline{(V_1|V_2|\ldots|V_n)}\ \underline{|x|y|\ldots|z}$

设 $\alpha = (V_1|V_2|\ldots|V_n)$

$\beta = x|y|\ldots|z$,

即简写为 $U \rightarrow U\alpha \mid \beta$

则： $U \rightarrow U'\{V_1|V_2|\ldots|V_n\}$

$U' \rightarrow x|y|\ldots|z$

或则： $U \rightarrow xU' \mid yU' \mid\ldots\mid zU'$

$U' \rightarrow V_1U'|V_2U'|\ldots|V_nU' \mid \varepsilon$

- 例1：$T \to T*F|T/F|F$

  变成：$T \to T(*F|/F)\ |F$

  消除左递归后得：$T \to F\{*F|/F\}$

- 或：

  $T' \to (*F|/F)T'|\ \varepsilon$

  $T \to FT'$

- 即：

  $T \to FT'$

  $T' \to *FT'|/FT'|\ \varepsilon$

(2)间接左递归的消除

——消除左递归的算法(直接和间接均消除)

要求：文法不含$A \rightarrow A$和$A \rightarrow \varepsilon$

原理：$S \rightarrow A\beta \mid \gamma$

$A \rightarrow S\alpha$

转换成 $S \rightarrow S\alpha\beta \mid \gamma$

间接左递归替换成直接左递归再消除

消除左递归的算法

①把G的非终结符号排序$A_1, A_2, \ldots, A_n$;

② for i:=1 to n do

Begin

for j:=1 to i-1 do

把每个形如$A_i \rightarrow A_j \gamma$ 的规则替换成

$A_i \rightarrow \delta_1 \gamma \mid \delta_2 \gamma \mid \ldots \mid \delta_m \gamma$,

其中 $A_j \rightarrow \delta_1 \mid \delta_2 \mid \ldots \mid \delta_m$是当前$A_j$的全部规则;

消除A_i规则中的直接递归 ;

End.

③化简由②得的文法,即去掉多余规则;

例： G[S]: $S \rightarrow Qc|c$

$Q \rightarrow Rb|b$

$R \rightarrow Sa|a$

消除左递归

解: (1) $A_1=R, A_2=Q, A_3=S$

(2)①$i=1, A_i=A_1=R$ 不含$R \rightarrow Rr$规则

②$i=2, A_i=A_2=Q$

$j=1, A_j=A_1=R$ 存在$Q \rightarrow Rb$

改写成$Q \rightarrow Sab|ab$(因$R \rightarrow Sa|a$) 无直接左递归

G中:$Q \rightarrow Sab|ab|b$ —— 原规则中的

例： G[S]: S→Qc|c

Q→Rb|b

R→Sa|a

③ i=3， $A_i=A_3=S$

$j=1, A_j=A_1=R$ 无S→Rc规则

$j=2, A_j=A_2=Q$ 含S→Qc规则

S→Sabc|abc|bc

G[S]中:S→Sabc|abc|bc|c 原规则中的

消除直接递归:S→(abc|bc|c){abc}

例： G[S]: S→Qc|c

Q→Rb|b

R→Sa|a

G[S]:S→(abc|bc|c){abc}

Q→Sab|ab|b 多余

R→Sa|a 多余

G[S]:S→S’{abc}

S’→abc|bc|c

或 S→abcS’|bcS’|cS’

S’ →abcS’| ε

- 说明:非终结符号的排序不同，所得文法形式就不同，但语言等价。

消除左递归的矩阵表示方法

- 设V_N中有n个非终结符号$X_1, X_2, \ldots, X_n$

将$X_i \rightarrow \gamma_1 | \gamma_2 | \ldots | \gamma_m$表示成

$X_i = \gamma_1 + \gamma_2 + \ldots + \gamma_m$ 将γ_i分成两类

$$\begin{cases} 首符号 \in V_N \\ 首符号 \in V_T \end{cases}$$

$X_i = X_1 \alpha_{1i} + X_2 \alpha_{2i} + \ldots + X_n \alpha_{ni} + \beta_i$ α_{ji}可能为∅

β_i是以终结符号开头的候选式的“和”

求解X_i

$$(X_1,X_2,\ldots,X_n)=(X_1,X_2,\ldots,X_n)\cdot\begin{vmatrix}\alpha_{11} & \alpha_{12} & \cdots & \alpha_{1n}\\ \alpha_{21} & \alpha_{22} & \cdots & \alpha_{2n}\\ \vdots & & \vdots & \\ \alpha_{n1} & \alpha_{n2} & \cdots & \alpha_{nn}\end{vmatrix}$$

$$+(\beta_{1},\beta_{2},\ldots\beta_{n})$$

或$X=XA+B$其解为$X=BA^*$　　A^*?

$$A^*=I_n+A+A^2+\ldots=I_n+AA^*$$

- 解出A^*很复杂，但我们的目的只是消除左递归，可绕开该问题。

$$I_n = \begin{pmatrix} \varepsilon & \Phi & \Phi & \cdots & \Phi \\ \Phi & \varepsilon & \Phi & \cdots & \Phi \\ \Phi & \Phi & \varepsilon & \cdots & \Phi \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \Phi & \Phi & \Phi & \cdots & \varepsilon \end{pmatrix} \qquad A^* = \begin{pmatrix} Z_{11} & Z_{12} & Z_{13} & \cdots & Z_{1n} \\ Z_{21} & Z_{22} & Z_{23} & \cdots & Z_{2n} \\ Z_{31} & Z_{32} & Z_{33} & \cdots & Z_{3n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ Z_{n1} & Z_{n2} & Z_{n3} & \cdots & Z_{nn} \end{pmatrix} = Z$$

即： X=BZ

$Z=I_n+AZ$

将矩阵展开，得新文法，不含左递归 ！

举　例

上例:S=Qc+c

Q=Rb+b

R=Sa+a

矩阵：

$$(\mathrm{S},\mathrm{Q},\mathrm{R}) = (\mathrm{S},\mathrm{Q},\mathrm{R})\underbrace{\begin{pmatrix} \Phi & \Phi & a \\ \mathrm{c} & \Phi & \Phi \\ \Phi & b & \Phi \end{pmatrix}}_{A} + \underbrace{(c,\mathrm{b},\mathrm{a})}_{B}$$

矩阵展开

$$(S,Q,R)=BZ=(c,b,a)\begin{pmatrix} Z_{11} & Z_{12} & Z_{13} \\ Z_{21} & Z_{22} & Z_{23} \\ Z_{31} & Z_{32} & Z_{33} \end{pmatrix}$$

$$Z=I_n+AZ=\begin{pmatrix} \varepsilon & \Phi & \Phi \\ \Phi & \varepsilon & \Phi \\ \Phi & \Phi & \varepsilon \end{pmatrix}+\begin{pmatrix} \Phi & \Phi & a \\ c & \Phi & \Phi \\ \Phi & b & \Phi \end{pmatrix}\begin{pmatrix} Z_{11} & Z_{12} & Z_{13} \\ Z_{21} & Z_{22} & Z_{23} \\ Z_{31} & Z_{32} & Z_{33} \end{pmatrix}$$

等价文法

S→ $cZ_{11}|bZ_{21}|aZ_{31}$

Q→ $cZ_{12}|bZ_{22}|aZ_{32}$

R→ $cZ_{13}|bZ_{23}|aZ_{33}$

$$(S,Q,R)=BZ=(c,b,a)\begin{pmatrix} Z_{11} & Z_{12} & Z_{13} \\ Z_{21} & Z_{22} & Z_{23} \\ Z_{31} & Z_{32} & Z_{33} \end{pmatrix}$$

$$Z=I_n+AZ=\begin{pmatrix} \varepsilon & \Phi & \Phi \\ \Phi & \varepsilon & \Phi \\ \Phi & \Phi & \varepsilon \end{pmatrix}+\begin{pmatrix} \Phi & \Phi & a \\ c & \Phi & \Phi \\ \Phi & b & \Phi \end{pmatrix}\begin{pmatrix} Z_{11} & Z_{12} & Z_{13} \\ Z_{21} & Z_{22} & Z_{23} \\ Z_{31} & Z_{32} & Z_{33} \end{pmatrix}$$

$Z_{11}\to aZ_{31}|\ \varepsilon$ $\quad$ $Z_{21}\to cZ_{11}$ $\quad$ $Z_{31}\to bZ_{21}$

$Z_{12}\to aZ_{32}$ $\quad$ $Z_{22}\to cZ_{12}\ |\ \varepsilon$ $\quad$ $Z_{32}\to bZ_{22}$

$Z_{13}\to aZ_{33}$ $\quad$ $Z_{23}\to cZ_{13}$ $\quad$ $Z_{33}\to bZ_{23}\ |\ \varepsilon$

2、回溯的消除

问题：不能准确地选定候选式；

解决：对文法的任何非终结符号，当要匹配输入串时，它能根据所面临的输入符号准确地指派一候选式。

需要对每一$A_i \rightarrow \alpha_1 | \alpha_2 | \ldots | \alpha_n$，$\alpha_i$的首字母（终结符号）要求不同即可。

如：$A \rightarrow aB|bC|c$

若有ε出现，则需考虑A_i后面的符号

如：$A \rightarrow aB|bC|\varepsilon$

推导到$S \overset{*}{\Rightarrow} \alpha A a \beta$，仍然不知选用哪条候选式

注意：事实上并非消除，而是设计文法避免之。

(1) 定义两个集合：First集和Follow集

$First(\alpha_i)=\{a \mid \alpha_i \stackrel{*}{\Rightarrow} a\delta$，且$a \in V_T$, α_i， $\delta \in V^*\}$

当 $\alpha_i \stackrel{*}{\Rightarrow} \varepsilon$ 时，则 $\varepsilon \in First(\alpha_i)$

$Follow(A)=\{a \mid S\# \stackrel{*}{\Rightarrow} \alpha A a \delta$，且$a \in V_T$, $\alpha, \delta \in V^*\}$

若$S \stackrel{*}{\Rightarrow} \alpha A$ 则 $\# \in Follow(A)$。

若S 为文法的开始符号，则$\# \in Follow(S)$？ √

（2）无回溯的条件

对于G中的每一个$A \in V_N$， $A \rightarrow \alpha_1 | \alpha_2 | \ldots | \alpha_n$

① $First(\alpha_i) \cap First(\alpha_j) = \Phi$ （$i \neq j$时）

② α_i和α_j至多有一个能推出ε

③若有α_i能推导出ε， 则

$First(\alpha_j) \cap Follow(A) = \Phi$ （$i \neq j$时）

$First(A) \cap Follow(A) = \Phi$