

编译原理与技术 H6

PB18111697 王章瀚

4.12

为文法

$$\begin{aligned} S &\rightarrow (L)|a \\ L &\rightarrow L, S|S \end{aligned}$$

(b) 写出自上而下分析的栈操作代码, 它打印每个 a 在句子中是第几个字符, 例如, 当句子是 $(a, (a, (a, a), (a)))$ 时, 打印的结果是 2 5 8 10 14

上一次作业中已经写出了消除了左递归的文法和翻译方案如下:

$Q \rightarrow S$	$Q \rightarrow \{S.first = 1;\}S$
$S \rightarrow (L)$	$S \rightarrow \{L.first = S.first;\}(L)\{S.last = L.last + 1;\}$
$S \rightarrow a$	$S \rightarrow a\{S.last = S.first; \text{print}(S.first);\}$
$L \rightarrow SR$	$L \rightarrow \{S.first = L.first\}S\{R.first = S.last + 1\}R\{L.last = R.last\}$
$R \rightarrow, SR_1$	$R \rightarrow, \{S.first = R.first + 1;\}S\{R_1.first = S.last + 1;\}R_1\{R.last = R_1.last\}$
$R \rightarrow \epsilon$	$R \rightarrow \{R.last = R.first\}$

这样, 对于一个LL(1)文法来说, 只需要系统地引入非终结符即可在LR分析期间完成属性计算. 引入的符号及引入后对栈操作代码如下:

翻译方案	语义规则	操作代码
$Q \rightarrow AS$	$S.first = A.last$	$val[top - 1] = val[top]$
$A \rightarrow \epsilon$	$A.last = 1$	$val[top] = 1$
$S \rightarrow (BL)$	$B.first = S.first + 1$ $L.first = B.last + 1$ $S.last = L.last + 1$	$val[top - 3] = val[top - 1] + 1$
$B \rightarrow \epsilon$	$B.last = B.first$	$val[top + 1] = val[top - 1] + 2$
$S \rightarrow a$	$S.last = S.first; print(S.first)$	$val[top] = val[top - 1] + 1$
$L \rightarrow SR$	$S.first = L.first + 1$ $R.first = S.last + 1$ $L.last = R.last$	$val[top - 1] = val[top]$
$R \rightarrow, DSR_1$	$D.first = R.first + 1$ $S.first = D.last + 1$ $R_1.first = S.last + 1$ $L.last = R_1.last$	$val[top - 3] = val[top]$
$D \rightarrow \epsilon$	$D.last = L.first$	$val[top + 1] = val[top - 1] + 1$
$R \rightarrow \epsilon$	$R.last = R.first$	$val[top + 1] = val[top]$

4.14

程序的文法如下:

$$\begin{aligned} P &\rightarrow D \\ D &\rightarrow D; D \mid id : T \mid proc\ id; D; S \end{aligned}$$

(a). 写一个语法制导定义, 打印该程序一共声明了多少个id.

即考虑文法(这里的 T 和 S 就当成终结符了)

$$\begin{aligned} P &\rightarrow D \\ D &\rightarrow D; D \\ D &\rightarrow id : T \\ D &\rightarrow proc\ id; D; S \end{aligned}$$

用 $count$ 表示直到当前符号包含的 id 个数. 则其语法制导定义如下:

产生式	语义规则
$P \rightarrow D$	$print(D.count)$
$D \rightarrow D_1; D_2$	$D.count = D_1.count + D_2.count$
$D \rightarrow id : T$	$D.count = 1$
$D \rightarrow proc\ id; D_1; S$	$D.count = 1 + D_1.count$

(b). 写一个翻译方案, 打印该程序每个变量id的嵌套深度. 例如, 当句型是 $a : T; proc\ b; ba : T; S$ 时, a 和 b 的嵌套深度是1, ba 的嵌套深度是2.

为了表示嵌套深度显然需要引入继承属性. 这里用 $depth$ 表示嵌套深度. 于是可以写出翻译方案如下:

$$\begin{aligned} P &\rightarrow \{D.depth = 0;\} D \\ D &\rightarrow \{D_1.depth = D.depth + 1;\} D_1; \{D_2.depth = D.depth + 1;\} D_2 \\ D &\rightarrow id : T \{print(D.depth);\} \\ D &\rightarrow proc\ id \{print(D.depth);\}; \{D_1.depth = D.depth + 1;\} D_1; S \end{aligned}$$