

编译原理与技术 H5

PB18111697 王章瀚

4.12

为文法

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

(b)

分别写出相应的语法制导定义, 翻译方案以及预测翻译器, 它打印出每个 a 在句子中是第几个字符. 例如, 当句子是 $(a, (a, (a, a), (a)))$ 时, 打印结果是 2 5 8 10 14

消除了左递归的1,2问答案在第3问都有描述

1. 语法制导定义

分析:

- 字符 a 的位置能够通过其左边部分判断, 因此使用 L 属性即可解决
- 字符 a 的位置不可能由字符 a 本身就判定, 因此需要继承属性

综上: 用 $first$ (继承属性)和 $last$ (综合属性)表示一个文法符号的第一个字符和最后一个字符的位置, 并加入一个 $S' \rightarrow S$ 以作为初始情况.

产生式	语义规则
$S' \rightarrow S$	$L.first = 1$
$S \rightarrow (L)$	$L.first = S.first + 1$ $S.last = L.last + 1$
$S \rightarrow a$	$S.last = S.first$ $print(S.first)$
$L \rightarrow L_1, S$	$L_1.first = L.first$ $S.first = L_1.last + 2$ $L.last = S.last$
$L \rightarrow S$	$S.first = L.first$ $L.last = S.last$

2. 翻译方案

根据 $first$ 和 $last$ 的综合和继承性, 很容易写出以下翻译方案:

$$\begin{aligned}
 S' &\rightarrow \{S'.first = 1;\}S \\
 S &\rightarrow \{L.first = S.first;\}(L)\{S.last = L.last + 1;\} \\
 S &\rightarrow a\{S.last = S.first; \text{print}(S.first);\} \\
 L &\rightarrow \{L_1.first = L.first\}L_1, \{S.first = L_1.last + 2;\}S\{L.last = S.last;\} \\
 L &\rightarrow \{S.first = L.first;\}S\{L.last = S.last;\}
 \end{aligned}$$

3. 预测翻译器

为了设计预测翻译器, 首先需要消除左递归, 使之成为LL(1)文法. 改进后文法如下:

$$\begin{aligned}
 Q &\rightarrow S \\
 S &\rightarrow (L) \\
 S &\rightarrow a \\
 L &\rightarrow SR \\
 R &\rightarrow, SR_1 \\
 R &\rightarrow \epsilon
 \end{aligned}$$

相应语法制导定义为:

产生式	语义规则
$Q \rightarrow S$	S.first=1
$S \rightarrow (L)$	L.first=S.first S.last=L.last+1
$S \rightarrow a$	S.last=S.first print(S.first)
$L \rightarrow SR$	S.first=L.first R.first=S.last+1 L.last=R.last
$R \rightarrow, SR_1$	$S.first = R.first + 1$ $R_1.first = S.last + 1$ $R.last = R_1.last$
$R \rightarrow \epsilon$	$R.last = R.fisrt$

从而得到翻译方案:

$$\begin{aligned}
 Q &\rightarrow \{S.first = 1;\}S \\
 S &\rightarrow \{L.first = S.first;\}(L)\{S.last = L.last + 1;\} \\
 S &\rightarrow a\{S.last = S.first; \text{print}(S.first);\} \\
 L &\rightarrow \{S.first = L.first\}S\{R.first = S.last + 1\}R\{L.last = R.last\} \\
 R &\rightarrow, \{S.first = R.first + 1;\}S\{R_1.first = S.last + 1;\}R_1\{R.last = R_1.last\} \\
 R &\rightarrow \{R.last = R.fisrt\}
 \end{aligned}$$

进而能够写出其预测翻译器代码如下(见下页), 为了代码言简意赅, 这里没有加入错误处理:

```

1 // 分析 Q 的函数
2 void Q() {
3     int s_first = 1;
4     S(s_first);
5 }
6
7 // 分析 S 的函数
8 int S(int s_first) {
9     int s_last;
10
11     if(lookahead == 'a') { // 产生式 S->a
12         match('a');
13         s_last = s_first;
14     }
15     else { // 产生式 S->(L)
16         int l_first, l_last;
17         l_first = s_first;
18         match('(');
19         l_last = L(l_first);
20         match(')');
21         s_last = l_last + 1;
22     }
23     return s_last;
24 }
25
26 // 分析的函数L
27 int L(int l_first) { //产生式 L->SR
28     int l_last, s_first, s_last, r_first, r_last;
29
30     s_first = l_first;
31     s_last = S(s_first);
32     r_first = s_last + 1;
33     r_last = R(r_first);
34     l_last = r_last;
35     return l_last;
36 }
37
38 // 分析的函数R
39 int R(int r_first) {
40     int r_last;
41
42     if(lookahead == ',') { // 产生式 R->,SR1
43         int s_first, s_last, r1_first, r1_last;
44         s_first = r_first + 1;
45         s_last = S(s_first);
46         r1_first = s_last + 1;
47         r1_last = R(r1_first);
48         r_last = r1_last;
49     }
50     else {
51         r_last = r_first;
52     }
53     return r_last;
54 }

```