编译原理

第七章 语义分析和中间代码产生

第七章 语义分析和中间代码产生

- 中间语言
- 赋值语句的翻译
- 布尔表达式的翻译
- 控制语句的翻译
- 过程调用的处理

第七章 语义分析和中间代码产生

- 中间语言
- 赋值语句的翻译
- 布尔表达式的翻译
- 控制语句的翻译
- 过程调用的处理

控制语句的翻译

- 控制语句
  - $S \rightarrow$ if E then S_1

    | if E then S_1 else S_2

    | while E do S_1
  - 标号与 goto 语句
  - CASE 语句

7.5.2 标号与 goto 语句

- 标号定义形式

  L: S ;

- 标号引用

  goto L ;

- 示例:

- 向后转移:

```
L1:   ……
      ……
      goto L1;
```

- 向前转移:

```
      goto L1;
      ……
L1:   ……
```

符号表信息

名字	类型	…	定义否	地址
…	…	…	…	…

- 向后转移

L1: ……

……

goto L1;

符号表信息

名字	类型	…	定义否	地址
…	…	…	…	…
L1	标号		已	p

- 向后转移

L1: ……

……

goto L1;

(p) (…, …, …, …)

符号表信息

名字	类型	…	定义否	地址
…	…	…	…	…
L1	标号		已	p

- 向后转移

L1: ……

……

goto L1;

(p) (…, …, …, …)

(n) (j, -, -, p)

符号表信息

- 向前转移

```
        goto L1;
        ……
L1:     ……
```

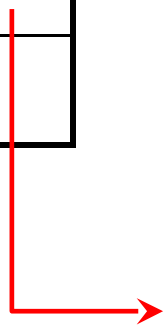
名字	类型	…	定义否	地址
…	…	…	…	…

符号表信息

- 向前转移

```
        goto L1;
        ……
L1:     ……
```

名字	类型	…	定义否	地址
…	…	…	…	…
L1	标号		未	p



(p) (j, -, -, 0)

符号表信息

- 向前转移

```
        goto L1;
        ……
        goto L1;
        ……
L1:     ……
```

名字	类型	…	定义否	地址
…	…	…	…	…
L1	标号		未	q

(p) (j, -, -, 0)

…

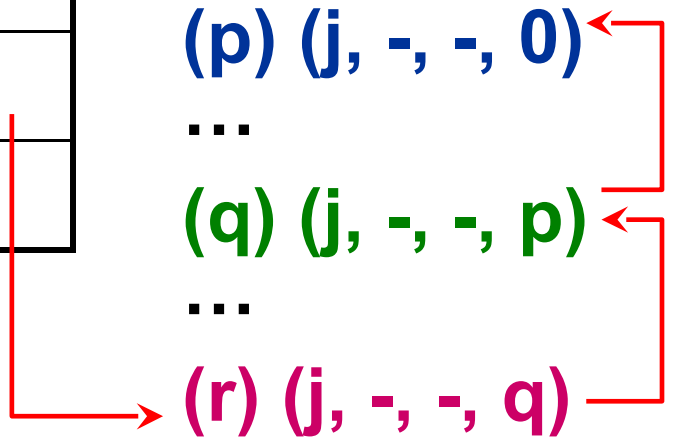
(q) (j, -, -, p)

符号表信息

名字	类型	…	定义否	地址
…	…	…	…	…
L1	标号		未	r

- 向前转移

```
        goto L1;
        ……
        goto L1;
        ……
        goto L1;
        ……
L1:     ……
```

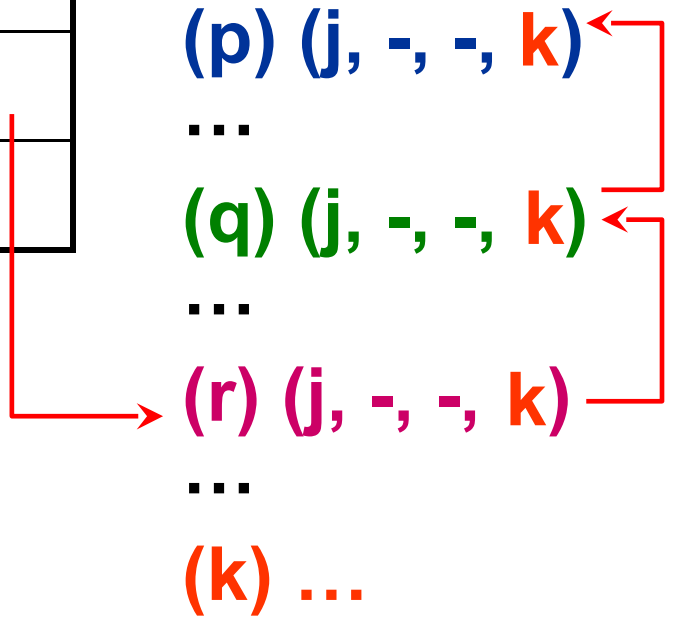


符号表信息

名字	类型	…	定义否	地址
…	…	…	…	…
L1	标号		已	k

- 向前转移

```
        goto L1;
        ……
        goto L1;
        ……
        goto L1;
        ……
L1:     ……
```



产生式 S′→goto L 的语义动作：

```
{   查找符号表;
    IF  L 在符号表中且 " 定义否 " 栏为 " 已 "
       THEN GEN(J ， - ， - ， P)
       ELSE IF  L 不在符号表中
            THEN BEGIN
                 把 L 填入表中;
                 置“定义否”为“未”，“地址”栏为 nextquad ;
                 GEN(J ， - ， - ， 0)
            END
            ELSE BEGIN
                 Q:=L 的地址栏中的编号;
                 置地址栏编号为 nextquad ;
                 GEN(J ， - ， - ， Q)
            END
}
```

- 带标号语句的产生式：

  S→label S　　label → i:

- label → i:　对应的语义动作：

1. 若 i 所指的标识符 (假定为 L) 不在符号表中，则把它填入，置 " 类型 " 为 " 标号 "，定义否为 " 已 "，" 地址 " 为 nextquad ；
2. 若 L 已在符号表中但 " 类型 " 不为标号或 " 定义否 " 为 " 已 "，则报告出错；
3. 若 L 已在符号表中，则把标号 " 未 " 改为 " 已 "，然后，把地址栏中的链头 (记为 q) 取出，同时把 nextquad 填在其中，最后，执行 BACKPATCH(q，nextquad)。

7.5.3 CASE 语句的翻译

- 语句结构

case E of

C_1: S_1;

C_2: S_2;

…

C_{n-1}: S_{n-1};

otherwise: S_n

end

```
case  E  of
    C1: S1;
    C2: S2;
     …
    Cn-1: Sn-1;
    otherwise: Sn
end
```

- 翻译法（一）：

```
        T:=E
L1:     if T≠C1 goto L2
        S1 的代码
        goto next
L2:     if T≠C2 goto L3
        S2 的代码
        goto next
L3:
        …
Ln-1:   if T≠Cn-1 goto Ln
        Sn-1 的代码
        goto next
Ln:     Sn 的代码
next:
```

- 改进

C_1	S_1 的地址
C_2	S_2 的地址
⋮	⋮
E	S_n 的地址

- 翻译法 (二):

计算 E 并放入 T 中
goto test

L_1: 关于 S_1 的中间码
goto next
…
L_{n-1}: 关于 S_{n-1} 的中间码
goto next
L_n: 关于 S_n 的中间码
goto next
test: if T=C_1 goto L_1
if T=C_2 goto L_2
…
if T=C_{n-1} goto L_{n-1}
goto L_n
next:

```
case E of
    C1: S1;
    C2: S2;
      …
    Cn-1: Sn-1;
    otherwise: Sn
end
```

L_1	S_1 □□□□□□□
L_2	S_2 □□□□□□□
⋮	⋮
L_{n-1}	S_{n-1} □□□□□□□
L_n	S_n □□□□□□□

C_1	P_1
C_2	P_2
⋮	⋮
C_{n-1}	P_{n-1}

(case, C_1, P_1)
(case, C_2, P_2)
…
(case, C_{n-1}, P_{n-1})
(case, T, P_n)
(label, NEXT, -, -)

P_i 是 L_i 在符号表中的位置

第七章 语义分析和中间代码产生

- 中间语言
- 赋值语句的翻译
- 布尔表达式的翻译
- 控制语句的翻译
- 过程调用的处理

7.6 过程调用的处理

- 过程调用主要完成两项工作
  - 传递参数
  - 转子
- 传地址
  - 把实在参数的地址传递给相应的形式参数
  - 调用段预先把实在参数的地址传递到被调用段可以拿到的地方
  - 程序控制转入被调用段之后，被调用段首先把实在参数的地址抄进自己相应的形式单元中
  - 过程体对形式参数的引用域赋值被处理成对形式单元的间接访问

过程调用的翻译

- 翻译方法：把实参的地址逐一放在转子指令的前面 .

  例如， CALL S(A ， X+Y) 翻译为：

```
计算 X+Y 置于 T 中的代码
par A     /* 第一个参数的地址 */
par T     /* 第二个参数的地址 */
call S    /* 转子 */
```

过程调用的翻译

- 过程调用文法：
  (1) S → call id (Elist)
  (2) Elist → Elist, E
  (3) Elist → E
- 参数的地址存放在一个队列中
- 最后对队列中的每一项生成一条 par 语句

```
CALL S(A ， X+Y) 翻译为
计算 X+Y 置于 T 中的代码
par A     /* 第一个参数的地址 */
par T     /* 第二个参数的地址 */
call S    /* 转子 */
```

过程调用的翻译

CALL S(A， X+Y) 翻译为
计算 X+Y 置于 T 中的代码
par A　/* 第一个参数的地址 */
par T　/* 第二个参数的地址 */
call S　/* 转子 */

- 翻译模式

3. Elist→E
　　{ 初始化 queue 仅包含 E.place }
2. Elist→Elist, E
　　{ 将 E.place 加入到 queue 的队尾 }
1. S→call id (Elist)
　{　for 队列 queue 中的每一项 p do
　　　emit(‘param’ p);
　　emit(‘call’ id.place) }

小结

- 标号与 goto 语句
- CASE 语句的翻译
- 过程调用的处理

本章小结

- 中间语言
- 表达式和赋值语句的翻译
- 布尔表达式的翻译
- 控制语句的翻译
- 过程调用的处理