

## 各种语言成分的语法及其翻译方案(示例)

### 1. 普通声明语句的翻译

下面是声明语句的文法:

$$\begin{aligned} P &\rightarrow \text{prog id (input, output) } D ; S \\ D &\rightarrow D ; D \mid \text{List} : T \mid \text{proc id } D ; S \\ \text{List} &\rightarrow \text{List}_1, \text{id} \mid \text{id} \\ T &\rightarrow \text{integer} \mid \text{real} \mid \text{array } C \text{ of } T_1 \mid {}^{\uparrow}T_1 \mid \text{record } D \\ C &\rightarrow [\text{num}] C \mid \varepsilon \end{aligned}$$

声明语句的翻译模式:

$$\begin{aligned} P &\rightarrow \text{prog id (input, output) } \{ \text{offset} := 0 \} D ; S \\ D &\rightarrow D ; D \\ D &\rightarrow \text{id} : T \{ \text{enter (id.name, T.type, offset); offset} := \text{offset} + T.\text{width} \} \\ T &\rightarrow \text{integer} \{ T.\text{type} := \text{integer}; T.\text{width} := 4 \} \\ T &\rightarrow \text{real} \{ T.\text{type} := \text{real}; T.\text{width} := 8 \} \\ T &\rightarrow \text{array [num] of } T_1 \{ T.\text{type} := \text{array(num.val, } T_1.\text{type}); T.\text{width} := \text{num.val} \times T_1.\text{width} \} \\ T &\rightarrow {}^{\uparrow}T_1 \{ T.\text{type} := \text{pointer}(T_1.\text{type}); T.\text{width} := 4 \} \end{aligned}$$

### 2. 嵌套过程中声明语句的翻译

嵌套过程声明语句的产生式。

$$\begin{aligned} P &\rightarrow \text{prog id (input, output) } D ; S \\ D &\rightarrow D ; D \mid \text{id} : T \mid \text{proc id} ; D ; S \end{aligned} \quad (7.1)$$

嵌套过程声明语句的翻译模式:

$$\begin{aligned} P &\rightarrow \text{prog id (input, output) } MD; S \{ \text{addwidth(top(tblptr), top(offset));} \\ &\quad \text{pop(tblptr); pop(offset)} \} \\ M &\rightarrow \varepsilon \{ t := \text{mktable(nil); push(t, tblptr); push(0, offset)} \} \\ D &\rightarrow D_1; D_2 \\ D &\rightarrow \text{proc id; } N D_1 ; S \{ t := \text{top(tblptr); addwidth(t, top(offset)); pop(tblptr);} \\ &\quad \text{pop(offset); enterproc(top(tblptr), id.name, t)} \} \\ D &\rightarrow \text{id} : T \{ \text{enter(top(tblptr), id.name, T.type, top(offset));} \\ &\quad \text{top(offset) := top(offset) + T.width} \} \\ N &\rightarrow \varepsilon \{ t := \text{mktable(top(tblptr)); push(t, tblptr); push(0, offset)} \} \end{aligned}$$

### 3. 记录的翻译

下面是生成记录类型的产生式:

$$T \rightarrow \text{record } D \text{ end}$$

生成记录类型的翻译模式:

$$\begin{aligned} T &\rightarrow \text{record } L D \text{ end} \{ T.\text{type} := \text{record(top(tblptr));} \\ &\quad T.\text{width} := \text{top(offset);} \\ &\quad \text{pop(tblptr); pop(offset)} \} \\ L &\rightarrow \varepsilon \{ t := \text{mktable(nil); push(t, tblptr); push(0, offset)} \} \end{aligned}$$

### 4. 赋值语句的翻译

下面是典型的赋值语句文法:

$$\begin{aligned} S &\rightarrow \text{Left} := E \\ E &\rightarrow E_1 + E_2 \mid E_1 * E_2 \mid - E_1 \mid (E_1) \mid \text{Left} \\ \text{Left} &\rightarrow E \text{list} \mid \text{id} \end{aligned}$$

$$Elist \rightarrow Elist, E \mid \text{id}[E] \quad (7.2)$$

赋值语句的翻译模式:

- (1)  $S \rightarrow Left := E \{ \text{if } Left.offset = \text{null} \text{ then } /*Left \text{ 是简单变量 } id*/$   
 $\quad \quad \quad gencode(Left.addr := E.addr);$   
 $\quad \quad \quad \text{else}$   
 $\quad \quad \quad \quad \quad gencode(Left.addr '[' Left.offset ']' := E.addr) /*Left \text{ 是数组元素}*/$
- (2)  $E \rightarrow E_1 + E_2 \{ E.addr := \text{newtemp}; gencode(E.addr := E_1.addr + E_2.addr) \}$
- (3)  $E \rightarrow (E_1) \{ E.addr := E_1.addr \}$
- (4)  $E \rightarrow Left \{ \text{if } Left.offset = \text{null} \text{ then } /*Left \text{ 是简单 } id*/$   
 $\quad \quad \quad E.addr := Left.addr$   
 $\quad \quad \quad \text{else begin } /*Left \text{ 是数组元素}*/$   
 $\quad \quad \quad \quad \quad E.addr := \text{newtemp};$   
 $\quad \quad \quad \quad \quad gencode(E.addr := Left.addr '[' Left.offset ']);$   
 $\quad \quad \quad \text{end} \}$
- (5)  $Left \rightarrow Elist \{ Left.addr := \text{newtemp}; /*Left \text{ 是数组元素, 因此存放基址和位移}*/$   
 $\quad \quad \quad Left.offset := \text{newtemp};$   
 $\quad \quad \quad gencode(Left.addr := c(Elist.array));$   
 $\quad \quad \quad gencode(Left.offset := Elist.addr * width(Elist.array)) \}$
- (6)  $Left \rightarrow \text{id} \{ Left.addr := \text{id.addr}; Left.offset := \text{null} \}$
- (7)  $Elist \rightarrow Elist_1, E \{ t := \text{newtemp}; m := Elist_1.ndim + 1;$   
 $\quad \quad \quad gencode(t := Elist_1.addr * limit(Elist_1.array, m)); /*计算 } e_{m-1} \times n_m */$   
 $\quad \quad \quad gencode(t := t + E.addr); /* 计算 + i_m */$   
 $\quad \quad \quad Elist.array := Elist_1.array;$   
 $\quad \quad \quad Elist.addr := t;$   
 $\quad \quad \quad Elist.ndim := m \}$
- (8)  $Elist \rightarrow \text{id}[E] \{ Elist.array := \text{id.addr}; Elist.addr := E.addr; Elist.ndim := 1 \}$

## 5. 各种控制结构的翻译

### 5.1 布尔表达式的翻译

布尔表达式的文法为:

- (1)  $B \rightarrow B_1 \text{ or } M B_2$
- (2)  $B \rightarrow B_1 \text{ and } M B_2$
- (3)  $B \rightarrow \text{not } B_1$
- (4)  $B \rightarrow (B_1)$
- (5)  $B \rightarrow E_1 \text{ relop } E_2$
- (6)  $B \rightarrow \text{true}$
- (7)  $B \rightarrow \text{false}$
- (8)  $M \rightarrow \epsilon$

布尔表达式的翻译模式如下所示:

- (1)  $B \rightarrow B_1 \text{ or } M B_2 \{ \text{backpatch}(B_1.\text{falselist}, M.\text{quad});$   
 $\quad \quad \quad B.\text{truelist} := \text{merge}(B_1.\text{truelist}, B_2.\text{truelist});$   
 $\quad \quad \quad B.\text{falselist} := B_2.\text{falselist} \}$
- (2)  $B \rightarrow B_1 \text{ and } M B_2 \{ \text{backpatch}(B_1.\text{truelist}, M.\text{quad});$

- $$\begin{aligned}
& B.\text{truelist} := B_2.\text{truelist}; \\
& B.\text{falselist} := \text{merge}(B_1.\text{falselist}, B_2.\text{falselist})\} \\
(3) & B \rightarrow \text{not } B_1 \{B.\text{truelist} := B_1.\text{falselist}; B.\text{falselist} := B_1.\text{truelist}\} \\
(4) & B \rightarrow (B_1) \{B.\text{truelist} := B_1.\text{truelist}; B.\text{falselist} := B_1.\text{falselist}\} \\
(5) & B \rightarrow E_1 \text{ relop } E_2 \{B.\text{truelist} := \text{makelist}(\text{nextquad}); \\
& \quad B.\text{falselist} := \text{makelist}(\text{nextquad}+1); \\
& \quad \text{gencode}(\text{'if' } E_1.\text{addr relop.op } E_2.\text{addr 'goto -'}); \\
& \quad \text{gencode}(\text{'goto -'})\} \\
(6) & B \rightarrow \text{true} \{B.\text{truelist} := \text{makelist}(\text{nextquad}); \text{gencode}(\text{'goto -'})\} \\
(7) & B \rightarrow \text{false} \{B.\text{falselist} := \text{makelist}(\text{nextquad}); \text{gencode}(\text{'goto -'})\} \\
(8) & M \rightarrow \epsilon \{M.\text{quad} := \text{nextquad}\}
\end{aligned}$$

## 5.2 常用控制流语句的翻译

控制流语句 **if-then**, **if-then-else** 和 **while-do** 的文法为:

- $$\begin{aligned}
(1) & S \rightarrow \text{if } B \text{ then } S_1 \\
(2) & S \rightarrow \text{if } B \text{ then } S_1 \text{ else } S_2 \\
(3) & S \rightarrow \text{while } B \text{ do } S_1 \\
(4) & S \rightarrow \text{begin } L \text{ end} \\
(5) & S \rightarrow A \\
(6) & L \rightarrow L_1; S \\
(7) & L \rightarrow S \tag{7.9}
\end{aligned}$$

**if-then**, **if-then-else** 和 **while-do** 语句的翻译模式:

- $$\begin{aligned}
(1) & S \rightarrow \text{if } B \text{ then } M_1 S_1 N \text{ else } M_2 S_2 \{ \text{backpatch}(B.\text{truelist}, M_1.\text{quad}); \\
& \quad \text{backpatch}(B.\text{falselist}, M_2.\text{quad}); \\
& \quad S.\text{nextlist} := \text{merge}(S_1.\text{nextlist}, \text{merge}(N.\text{nextlist}, S_2.\text{nextlist})) \} \\
(2) & N \rightarrow \epsilon \{N.\text{nextlist} := \text{makelist}(\text{nextquad}); \text{gencode}(\text{'goto -'})\} \\
(3) & M \rightarrow \epsilon \{M.\text{quad} := \text{nextquad}\} \\
(4) & S \rightarrow \text{if } B \text{ then } M S_1 \{ \text{backpatch}(B.\text{truelist}, M.\text{quad}); \\
& \quad S.\text{nextlist} := \text{merge}(B.\text{falselist}, S_1.\text{nextlist}) \} \\
(5) & S \rightarrow \text{while } M_1 B \text{ do } M_2 S_1 \{ \text{backpatch}(S_1.\text{nextlist}, M_1.\text{quad}); \\
& \quad \text{backpatch}(B.\text{truelist}, M_2.\text{quad}); S.\text{nextlist} := B.\text{falselist}; \text{gencode}(\text{'goto' } M_1.\text{quad}) \} \\
(6) & S \rightarrow \text{begin } L \text{ end} \{S.\text{nextlist} := L.\text{nextlist}\} \\
(7) & S \rightarrow A \{S.\text{nextlist} := \text{nil}\} \\
(8) & L \rightarrow L_1; MS \{ \text{backpatch}(L_1.\text{nextlist}, M.\text{quad}); L.\text{nextlist} := S.\text{nextlist} \} \\
(9) & L \rightarrow S \{L.\text{nextlist} := S.\text{nextlist}\}
\end{aligned}$$

## 5.3 for 循环语句的翻译

**for** 循环语句的文法如下所示:

$$S \rightarrow \text{for id} := E_1 \text{ to } E_2 \text{ step } E_3 \text{ do } S_1$$

**for** 循环语句的翻译模式如下所示:

$$\begin{aligned}
S & \rightarrow \text{for id} := E_1 \text{ to } E_2 \text{ step } E_3 \text{ do } M S_1 \{ \text{backpatch}(S_1.\text{nextlist}, M.\text{again}); \\
& \quad \text{gencode}(\text{'goto' }, -, -, M.\text{again}); S.\text{nextlist} := M.\text{again}; \} \\
M & \rightarrow \epsilon \{ M.\text{addr} := \text{entry}(\text{id}); \text{gencode}(\text{'='}, E_1.\text{addr}, -, M.\text{addr}); T_1 := \text{newtemp}; \\
& \quad \text{gencode}(\text{'='}, E_2.\text{addr}, -, T_1); T_2 := \text{newtemp}; \text{gencode}(\text{'='}, E_3.\text{addr}, -, T_2); q := \text{nextquad};
\end{aligned}$$

$gencode('goto', -, -, q+2); M.again:=q+1; gencode('+', M.addr, T_2, M.addr);$   
 $M.nextlist:=nextquad; gencode('if' M.addr '>' T_1 'goto -');\}$

#### 5.4 repeat 语句的翻译

**repeat** 语句的文法如下所示:

$S \rightarrow \text{repeat } S_1 \text{ until } B$

**Repeat** 语句的翻译模式如下所示:

$S \rightarrow \text{repeat } M S_1 \text{ until } N B \{ \text{backpatch}(B.\text{falselist}, M.\text{quad});$

$S.\text{nextlist}:=B.\text{truelist} \}$

$M \rightarrow \varepsilon \{ M.\text{quad} := \text{nextquad} \}$

$N \rightarrow \varepsilon \{ \text{backpatch}(S_1.\text{nextlist}, \text{nextquad}) \}$

#### 6. switch 语句的语法制导翻译

**switch** 语句的文法为:

$S \rightarrow \text{switch } (E) \text{ Clist}$

$\text{Clist} \rightarrow \text{case } V : S \text{ Clist} \mid \text{default} : S$

**switch** 语句的翻译模式如下所示:

(1)  $S \rightarrow \text{switch } (E) \{ i:=0; S_i.\text{nextlist}:=0; \text{push } S_i.\text{nextlist}; \text{push } E.\text{addr}; \text{push } i; q:=0; \text{push } q \}$

$\text{Clist} \{ \text{pop } q; \text{pop } i; \text{pop } E.\text{addr}; \text{pop } S_i.\text{nextlist}; S.\text{nextlist}:=\text{merge}(S_i.\text{nextlist}, q); \text{push } S.\text{nextlist} \}$

(2)  $\text{Clist} \rightarrow \text{case } V : \{ \text{pop } q; \text{pop } i; i:=i+1; \text{pop } E.\text{addr};$

**if**  $\text{nextquad} \neq 0$  **then**  $\text{backpatch}(q, \text{nextquad});$

$q:=\text{nextquad};$

$gencode('if' E.\text{addr} \neq V_i 'goto' L_i);$

$\text{push } E.\text{addr}; \text{push } i;$

$\text{push } q \} S \{ \text{pop } q; \text{pop } i; \text{pop } E.\text{addr}; \text{pop } S_{i-1}.\text{nextlist};$

$p:=\text{nextquad};$

$gencode('goto -'); gencode(L_i ':');$

$S_i.\text{nextlist}:=\text{merge}(S_i.\text{nextlist}, p);$

$S_i.\text{nextlist}:=\text{merge}(S_i.\text{nextlist}, S_{i-1}.\text{nextlist});$

$\text{push } S_i.\text{nextlist}; \text{push } E.\text{addr}; \text{push } i; \text{push } q \} \text{Clist}$

(3)  $\text{Clist} \rightarrow \text{default} : \{ \text{pop } q; \text{pop } i; i:=i+1; \text{pop } E.\text{addr};$

**if**  $\text{nextquad} \neq 0$  **then**  $\text{backpatch}(q, \text{nextquad});$

$q:=\text{nextquad};$

$gencode('if' E.\text{addr} \neq V_i 'goto' V_{i+1});$

$\text{push } E.\text{addr}; \text{push } i;$

$\text{push } q \} S \{ \text{pop } q; \text{pop } i; \text{pop } E.\text{addr}; \text{pop } S_{i-1}.\text{nextlist};$

$p:=\text{nextquad};$

$gencode('goto -'); gencode(L_i ':');$

$S_i.\text{nextlist}:=\text{merge}(S_i.\text{nextlist}, p);$

$S_i.\text{nextlist}:=\text{merge}(S_i.\text{nextlist}, S_{i-1}.\text{nextlist});$

$\text{push } S_i.\text{nextlist}; \text{push } E.\text{addr}; \text{push } i; \text{push } q \}$

#### 7. 过程调用和返回语句的翻译

过程调用和返回语句的文法如下所示:

$$S \rightarrow \text{call id}(Elist)$$

$$Elist \rightarrow Elist, E \mid E$$

$$S \rightarrow \text{return } E$$

过程调用语句的翻译模式如下所示:

(1)  $S \rightarrow \text{call id}(Elist) \{n := 0;$

**repeat**

$n := n + 1;$

从 *queue* 的队首取出一个实参地址 *p*;

$\text{gencode}(\text{'param'}, -, -, p);$

**until** *queue* 为空;

$\text{gencode}(\text{'call'}, \text{id.addr}, n, -);$

(2)  $Elist \rightarrow Elist, E \{$ 将 *E.addr* 添加到 *queue* 的队尾}

(3)  $Elist \rightarrow E \{$ 初始化 *queue*, 然后将 *E.addr* 加入到 *queue* 的队尾。 $\}$

过程返回语句的翻译模式为:

$S \rightarrow \text{return } E \{$ 需要返回结果 **then**  $\text{gencode}(\text{'='}, E.addr, -, F);$

$\text{gencode}(\text{'ret'}, -, -, -);$

其中, *F* 是存放结果的指定单元, 四元式('ret', -, -, -)执行如下操作:

(1) 恢复主调程序的寄存器内容;

(2) 释放过程运行时所占用的数据区;

(3) 按返回地址返回到主调程序。

## 8. 输入输出语句的翻译

带 I/O 参数的程序语句和输入输出语句的文法如下所示:

$$P \rightarrow \text{prog id}(\text{input}, \text{output}) D; S$$

$$S \rightarrow \text{read}(List)$$

|  $\text{readln}(List)$

$$S \rightarrow \text{write}(Elist)$$

|  $\text{writeln}(Elist)$

带 I/O 参数的程序语句和输入输出语句的翻译方案如下所示:

$$P \rightarrow \text{prog id}(\text{Parlist}) M D; S$$

$$\text{Parlist} \rightarrow \text{input}(\varepsilon \mid -, \text{output})$$

$$S \rightarrow (\text{read} \mid \text{readln})(N List); \{n := 0;$$

**repeat**

$\text{move}(\text{Queue}, i_n);$

$\text{gencode}(\text{'par'}, \text{'in'}, -, -);$

$n := n + 1;$

**until** *Queue* 为空;

$\text{gencode}(\text{'call'}, \text{'SYSIN'}, n-1, -);$

$$List \rightarrow \text{id}, L(\varepsilon \mid List)$$

$$S \rightarrow (\text{write} \mid \text{writeln})(Elist); \{n := 0;$$

**repeat**

$\text{move}(\text{Queue}, i_n);$

$\text{gencode}(\text{'par'}, \text{'out'}, -, -);$

$n := n + 1;$

```

        until Queue 为空;
        gencode('call', 'SYSOUT', n, 'w')}
/*n 为输出参数个数, w 是输出操作类型*/

EList  $\rightarrow$  E, K ( $\epsilon$ |EList)
M  $\rightarrow$   $\epsilon$  {gencode('prog', id, y, -)} /*y 的值表示 input, output 或两者皆有*/
N  $\rightarrow$   $\epsilon$  {设置一个语义队列 Queue}
L  $\rightarrow$   $\epsilon$  {T:=entry(id); add(Queue, T)}
K  $\rightarrow$   $\epsilon$  {T:= E.addr; add(Queue, T)}

```