中间代码生成
(2. 回填技术)

魏恒峰

hfwei@nju.edu.cn

2021 年 12 月 21 日

$S \rightarrow \text{if (} B \text{) } S_1$
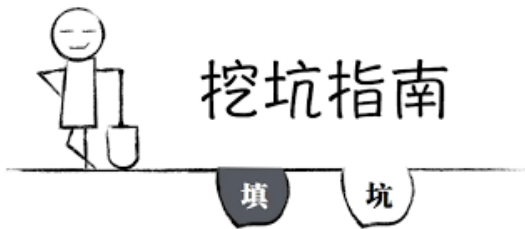
$$\begin{array}{l} B.true \ = \ newlabel() \\ B.false \ = \ S_1.next \ = \ S.next \\ S.code \ = \ B.code \ || \ label(B.true) \ || \ S_1.code \end{array}$$

B 还不知道 $S.next$ 的指令地址, 如何跳转?

再扫描一遍中间代码, 将标号替换成指令 (相对) 地址

可否在生成中间代码的时候就填入指令地址?

回填 (Backpatching) 技术

子节点挖坑、祖先节点填坑

子节点暂时不指定跳转指令的目标

待祖先节点能够确定正确的目标地址时回头填充

父节点通过**综合属性**收集子节点中具有相同目标的跳转指令

在自底向上的分析过程中

为左部非终结符 B 计算综合属性 $B.\text{truelist}$ 与 $B.\text{falselist}$

为左部非终结符 S/L 计算综合属性 $S/L.\text{nextlist}$

并为已能确定目标地址的跳转指令进行回填 (考虑每个综合属性)

针对布尔表达式的回填技术

1) $B \rightarrow B_1 \ || \ \boxed{M} \ B_2$ { $backpatch(B_1.falselist, M.instr)$;
$B.truelist = merge(B_1.truelist, B_2.truelist)$;
$B.falselist = B_2.falselist$; }

2) $B \rightarrow B_1 \ \&\& \ \boxed{M} \ B_2$ { $backpatch(B_1.truelist, M.instr)$;
$B.truelist = B_2.truelist$;
$B.falselist = merge(B_1.falselist, B_2.falselist)$; }

3) $B \rightarrow \ ! \ B_1$ { $B.truelist = B_1.falselist$;
$B.falselist = B_1.truelist$; }

4) $B \rightarrow (\ B_1 \ )$ { $B.truelist = B_1.truelist$;
$B.falselist \ = \ B_1.falselist$; }

5) $B \rightarrow E_1 \ \mathbf{rel} \ E_2$ { $B.truelist = makelist(nextinstr)$;
$B.falselist = makelist(nextinstr + 1)$;
$gen('\texttt{if}' \ E_1.addr \ \mathbf{rel}.op \ E_2.addr \ '\texttt{goto} \ \_')$;
$gen('\texttt{goto} \ \_')$; }

6) $B \rightarrow \mathbf{true}$ { $B.truelist = makelist(nextinstr)$;
$gen('\texttt{goto} \ \_')$; }

7) $B \rightarrow \mathbf{false}$ { $B.falselist = makelist(nextinstr)$;
$gen('\texttt{goto} \ \_')$; }

8) $\boxed{M \rightarrow \epsilon}$ { $M.instr = nextinstr$; }

综合属性 $B.truelist$ 保存 需要跳转到 $B.true$ 的指令地址

6) $B \rightarrow \textbf{true}$ 　　　　 $\{ \boxed{B.truelist} = makelist(nextinstr);$
　　　　　　　　　　　　 $gen('goto \_\,'); \}$

7) $B \rightarrow \textbf{false}$ 　　　 $\{ \boxed{B.falselist} = \boxed{makelist}\, nextinstr);$
　　　　　　　　　　　　 $gen('goto \_\,'); \}$

综合属性 $B.falselist$ 保存 需要跳转到 $B.false$ 的指令地址

$B \rightarrow \textbf{true}$	$B.code = gen('goto'\ \boxed{B.true})$
$B \rightarrow \textbf{false}$	$B.code = gen('goto'\ \boxed{B.false})$

5) $B \to E_1 \ \mathbf{rel} \ E_2$ { $B.truelist = makelist(nextinstr);$
                                          $B.falsclist = makelist(nextinstr + 1);$
                                          $gen('\mathtt{if}' \ E_1.addr \ \mathbf{rel}.op \ E_2.addr \ '\mathtt{goto}' \ \_);$
                                          $gen('\mathtt{goto}' \ \_); \}$

$B \ \to \ E_1 \ \mathbf{rel} \ E_2$ $\Big|$ $B.code = E_1.code \parallel E_2.code$
                                          $\parallel gen('\mathtt{if}' \ E_1.addr \ \mathbf{rel}.op \ E_2.addr \ '\mathtt{goto}' \ B.true$
                                          $\parallel gen('\mathtt{goto}' \ B.false)$

$3)$ $B \rightarrow \ !\ B_1$

$\{\ \boxed{B.truelist} = B_1.falselist;$
$\boxed{B.falselist} = B_1.truelist; \ \}$

$4)$ $B \rightarrow (\ B_1\ )$

$\{\ \boxed{B.truelist} = B_1.truelist;$
$\boxed{B.falselist} = B_1.falselist; \ \}$

$B \rightarrow \ !\ B_1$

$B_1.true = B.false$
$B_1.false = B.true$
$B.code = B_1.code$

2) $B \rightarrow B_1 \ \&\& \ M \ B_2$ { $\boxed{backpatch}(B_1.truelist, M.instr);$
$\boxed{B.truelist} = B_2.truelist;$
$\boxed{B.falselist} = merge(B_1.falselist, B_2.falselist);$ }

8) $\boxed{M \rightarrow \epsilon}$ { $M.instr = nextinstr;$ }

$B \rightarrow B_1 \ \&\& \ B_2$ | $\boxed{B_1.true = newlabel()}$
$\boxed{B_1.false} = B.false$
$B_2.true = B.true$
$\boxed{B_2.false} = B.false$
$B.code = B_1.code \ || \ label(B_1.true) \ || \ B_2.code$

1) $B \rightarrow B_1 \ || \ M \ B_2$ { $backpatch(B_1.falselist, M.instr);$
                                            $B.truelist = merge(B_1.truelist, B_2.truelist);$
                                            $B.falselist = B_2.falselist;$ }

8) $M \rightarrow \epsilon$ { $M.instr = nextinstr;$ }

$B \rightarrow B_1 \ || \ B_2$ | $B_1.true = B.true$
                                 | $B_1.false = newlabel()$
                                 | $B_2.true = B.true$
                                 | $B_2.false = B.false$
                                 | $B.code = B_1.code \ || \ label(B_1.false) \ || \ B_2.code$

x < 100 || x > 200 && x != y
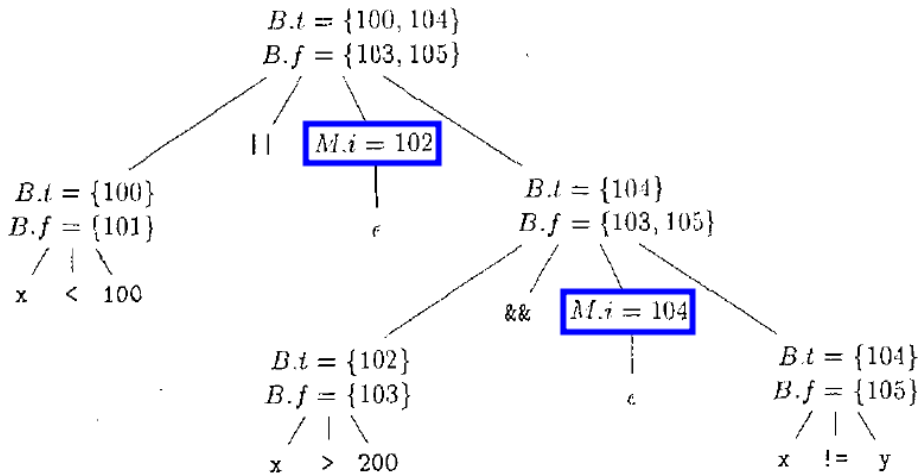
```
100:   if x < 100 goto _
101:   goto _
102:   if x > 200 goto 104
103:   goto _
104:   if x != y goto _
105:   goto _
```

a) 将 104 回填到指令 102 中之后

```
100:   if x < 100 goto _
101:   goto 102
102:   if x > 200 goto 104
103:   goto _
104:   if x != y goto _
105:   goto _
```

b) 将 102 回填到指令 101 中之后

$S \rightarrow \textbf{if}\,(\,B\,)\,S \mid \textbf{if}\,(\,B\,)\,S\,\text{else}\,S \mid \textbf{while}\,(\,B\,)\,S \mid \boxed{\{\,L\,\}} \mid A \; ;$

$L \rightarrow L\,S \mid S$

1) $S \rightarrow \textbf{if}\,(\,B\,)\,M\,S_1$ { $backpatch(B.truelist,\ M.instr);$
$\qquad\qquad\qquad\qquad\qquad S.nextlist\ =\ merge(B.falselist,\ S_1.nextlist);$ }

2) $S \rightarrow \quad \textbf{if}\,(\,B\,)\,M_1\,S_1\,N\,\textbf{else}\,M_2\,S_2$
$\qquad\qquad\qquad\qquad\quad$ { $backpatch(B.truelist,\ M_1.instr);$
$\qquad\qquad\qquad\qquad\qquad backpatch(B.falselist,\ M_2.instr);$
$\qquad\qquad\qquad\qquad\qquad temp\ =\ merge(S_1.nextlist,\ N.nextlist);$
$\qquad\qquad\qquad\qquad\qquad S.nextlist\ =\ merge(temp,\ S_2.nextlist);$ }

3) $S \rightarrow \quad \textbf{while}\,M_1\,(\,B\,)\,M_2\,S_1$
$\qquad\qquad\qquad\qquad\quad$ { $backpatch(S_1.nextlist,\ M_1.instr);$
$\qquad\qquad\qquad\qquad\qquad backpatch(B.truelist,\ M_2.instr);$
$\qquad\qquad\qquad\qquad\qquad S.nextlist\ =\ B.falselist;$
$\qquad\qquad\qquad\qquad\qquad gen('\text{goto}'\ M_1.instr);$ }

4) $S \rightarrow \{\,L\,\}$ $\qquad$ { $S.nextlist\ =\ L.nextlist;$ }

5) $S \rightarrow A\,;$ $\qquad\qquad$ { $S.nextlist\ =\ \textbf{null};$ }

6) $M \rightarrow \epsilon$ $\qquad\qquad$ { $M.instr\ =\ nextinstr;$ }

7) $N \rightarrow \epsilon$ $\qquad\qquad$ { $N.nextlist\ =\ makelist(nextinstr);$
$\qquad\qquad\qquad\qquad\qquad gen('\text{goto}\ \_');$ }

8) $L \rightarrow L_1\,M\,S$ $\qquad$ { $backpatch(L_1.nextlist,\ M.instr);$
$\qquad\qquad\qquad\qquad\qquad L.nextlist\ =\ S.nextlist;$ }

9) $L \rightarrow S$ $\qquad\qquad$ { $L.nextlist\ =\ S.nextlist;$ }

1) $S \rightarrow \text{if } (B) M \ S_1 \ \{ \ \boxed{backpatch}(B.truelist, \ M.instr);$
$\boxed{S.nextlist} \ = \ merge(B.falselist, \ S_1.nextlist); \ \}$

6) $M \rightarrow \epsilon$ $\{ \ M.instr \ = \ nextinstr; \ \}$

$S \rightarrow \text{if } (B) S_1$

$\begin{vmatrix} \boxed{B.true} = newlabel() \\ B.false = S_1.next = \boxed{S.next} \\ S.code = B.code \ || \ \boxed{label(B.true)} \ || \ S_1.code \end{vmatrix}$

$$S \rightarrow \textbf{if} \ (\ B \ ) \ M_1 \ S_1 \ N \ \textbf{else} \ M_2 \ S_2$$
$$\{ \ \boxed{backpatch}(B.truelist, \ M_1.instr);$$
$$\boxed{backpatch}(B.falselist, \ M_2.instr);$$
$$temp \ = \ merge(S_1.nextlist, \ N.nextlist);$$
$$\boxed{S.nextlist} \ = \ merge(temp, \ S_2.nextlist); \ \}$$

$$6) \ M \rightarrow \epsilon \qquad \{ \ M.instr \ = \ nextinstr; \ \}$$

$$7) \ N \rightarrow \epsilon \qquad \{ \ N.nextlist \ = \ makelist(nextinstr);$$
$$\boxed{gen('\texttt{goto \_}');} \ \}$$

$$S \rightarrow \textbf{if} \ (\ B \ ) \ S_1 \ \textbf{else} \ S_2 \ \Big| \ \begin{array}{l} B.true \ = \ newlabel() \\ B.false \ = \ newlabel() \\ S_1.next \ = \ S_2.next \ = \ \boxed{S.next} \\ S.code \ = \ B.code \\ \qquad \| \ label(\boxed{B.true}) \ \| \ S_1.code \\ \qquad \| \ gen('\texttt{goto}' \ \boxed{S.next}) \\ \qquad \| \ label(\boxed{B.false}) \ \| \ S_2.code \end{array}$$

3) $S \rightarrow$ **while** M_1 (B) M_2 S_1

$$\{ \boxed{backpatch}(S_1.nextlist, \ M_1.instr);$$
$$\boxed{backpatch}(B.truelist, \ M_2.instr);$$
$$\boxed{S.nextlist} = \ B.falselist;$$
$$gen(\boxed{'goto' \ M_1.instr);} \}$$

6) $M \rightarrow \epsilon$ $\qquad \{ \ M.instr \ = \ nextinstr; \ \}$

$S \rightarrow$ **while** (B) S_1

$$begin = \ newlabel()$$
$$B.true = \ newlabel()$$
$$B.false = \ \boxed{S.next}$$
$$S_1.next = \ begin$$
$$S.code = \ label(\boxed{begin}) \ || \ B.code$$
$$|| \ label(\boxed{B.true}) \ || \ S_1.code$$
$$|| \ gen(\boxed{'goto' \ begin})$$

8) $L \rightarrow L_1 \ M \ S$ $\{$ $backpatch$($L_1.nextlist$, $M.instr$);
$L.nextlist$ = $S.nextlist$; $\}$

9) $L \rightarrow S$ $\{$ $L.nextlist$ = $S.nextlist$; $\}$

$4)$ $\quad S \to \{\ L\ \}$ $\qquad\qquad \{\ \boxed{S.nextlist}\ =\ L.nextlist;\ \}$

$5)$ $\quad S \to A\ ;$ $\qquad\qquad \{\ \boxed{S.nextlist}\ =\ \textbf{null};\ \}$

1) $S \rightarrow \textbf{if } (B) \ M \ S_1$ $\{ \ backpatch(B.truelist, \ M.instr);$
                                  $S.nextlist \ = \ merge(B.falselist, \ S_1.nextlist); \ \}$

2) $S \rightarrow \textbf{if } (B) \ M_1 \ S_1 \ N \ \textbf{else} \ M_2 \ S_2$
                             $\{ \ backpatch(B.truelist, \ M_1.instr);$
                                $backpatch(B.falselist, \ M_2.instr);$
                                $temp \ = \ merge(S_1.nextlist, \ N.nextlist);$
                                $S.nextlist \ = \ merge(temp, \ S_2.nextlist); \ \}$

3) $S \rightarrow \textbf{while } M_1 \ (B) \ M_2 \ S_1$
                              $\{ \ backpatch(S_1.nextlist, \ M_1.instr);$
                                $backpatch(B.truelist, \ M_2.instr);$
                                $S.nextlist \ = \ B.falselist;$
                                $\boxed{gen('\text{goto}' \ M_1.instr);} \ \}$

4) $S \rightarrow \{ \ L \ \}$ $\{ \ S.nextlist \ = \ L.nextlist; \ \}$

5) $S \rightarrow A \ ;$ $\{ \ S.nextlist \ = \ \textbf{null}; \ \}$

6) $M \rightarrow \epsilon$ $\{ \ M.instr \ = \ nextinstr; \ \}$

7) $N \rightarrow \epsilon$ $\{ \ N.nextlist \ = \ makelist(nextinstr);$
                              $\boxed{gen('\text{goto \_}');} \ \}$

8) $L \rightarrow L_1 \ M \ S$ $\{ \ backpatch(L_1.nextlist, \ M.instr);$
                                $L.nextlist \ = \ S.nextlist; \ \}$

9) $L \rightarrow S$ $\{ \ L.nextlist \ = \ S.nextlist; \ \}$

只有 (3) 与 (7) 生成了新的代码, 控制流语句的主要目的是"控制" 流。

```
1:  procedure AREYOUOK(score)
2:      if score ≥ 60 then
3:          while true do
4:              print "Happy New Year"
5:      else
6:          print "Sad"
```

2) $S \rightarrow$ **if** $(B) M_1 S_1 N$ **else** $M_2 S_2$
$\{ backpatch(B.truelist, M_1.instr);$
$backpatch(B.falselist, M_2.instr);$
$temp = merge(S_1.nextlist, N.nextlist);$
$S.nextlist = merge(temp, S_2.nextlist); \}$

3) $S \rightarrow$ **while** $M_1 (B) M_2 S_1$
$\{ backpatch(S_1.nextlist, M_1.instr);$
$backpatch(B.truelist, M_2.instr);$
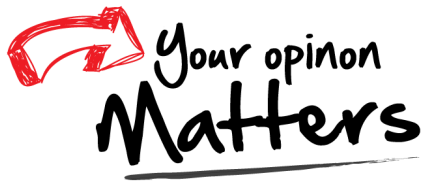$S.nextlist = B.falselist;$
$gen('goto' M_1.instr); \}$

6) $M \rightarrow \epsilon$ $\{ M.instr = nextinstr; \}$

7) $N \rightarrow \epsilon$ $\{ N.nextlist = makelist(nextinstr);$
$gen('goto' \_'); \}$

6) $B \rightarrow$ **true** $\{ B.truelist = makelist(nextinstr);$
$gen('goto' \_'); \}$

Office 926

hfwei@nju.edu.cn