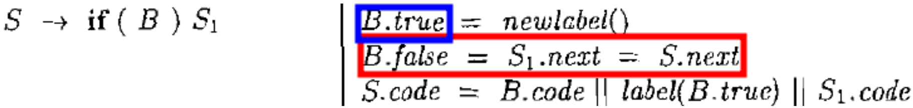
中间代码生成
(2. 回填技术)

魏恒峰

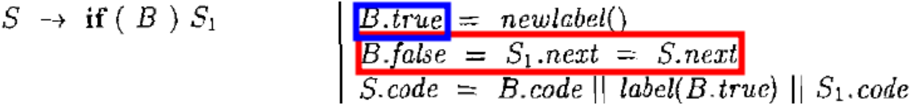
hfwei@nju.edu.cn

2021 年 12 月 21 日



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

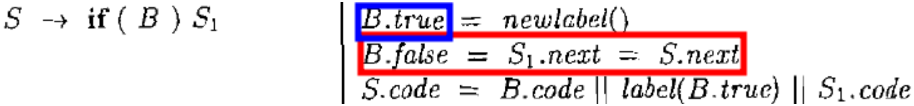
B 还不知道 $S.next$ 的指令地址，如何跳转？



$$S \to \textbf{if} \; (\; B \;) \; S_1 \qquad \begin{array}{|l} \boxed{B.true} = newlabel() \\ \boxed{B.false = S_1.next = S.next} \\ S.code = B.code \;||\; label(B.true) \;||\; S_1.code \end{array}$$

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



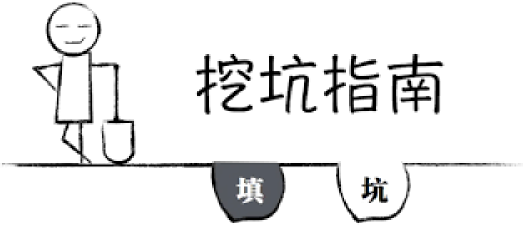
$$S \rightarrow \textbf{if} \; (\; B \;) \; S_1 \qquad \begin{array}{|l} \boxed{B.true} = newlabel() \\ \boxed{B.false = S_1.next = S.next} \\ S.code = B.code \;||\; label(B.true) \;||\; S_1.code \end{array}$$

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

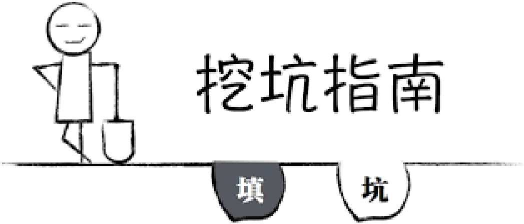
可否在生成中间代码的时候就填入指令地址？

回填 (Backpatching) 技术



子节点挖坑、祖先节点填坑

回填（Backpatching）技术

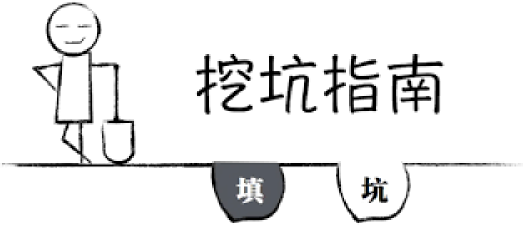


子节点挖坑、祖先节点填坑

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

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

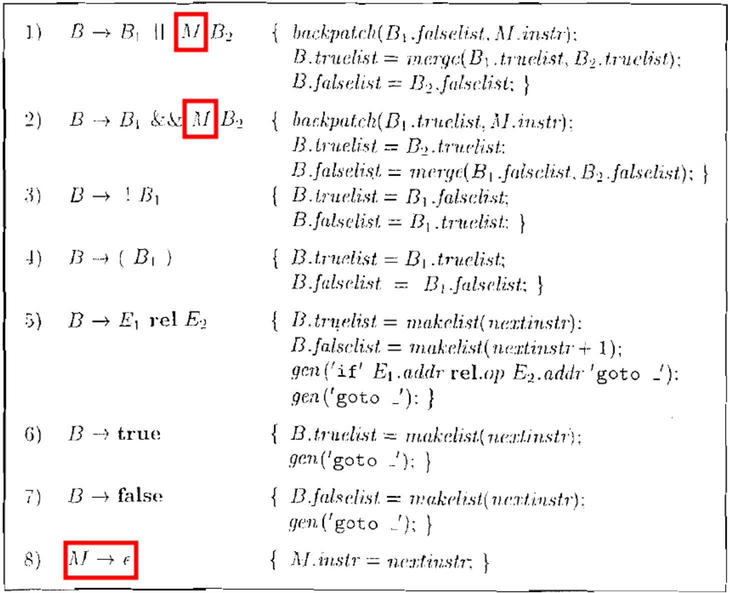
在自底向上的分析过程中

为左部非终结符 B 计算 B.truelist 与 B.falselist

为左部非终结符 S 计算 S.nextlist

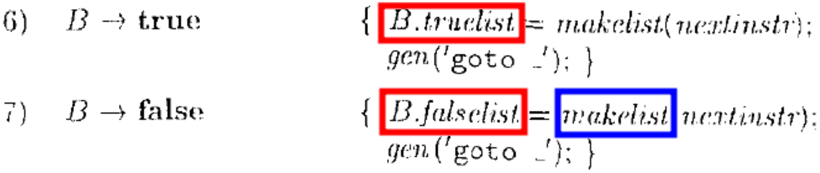
并为已能确定目标地址的跳转指令进行回填

针对布尔表达式的回填技术



1)	$B \to B_1 \ \mid\mid\ \boxed{M}\ B_2$	$\{\ backpatch(B_1.falselist, M.instr);$ $B.truelist = merge(B_1.truelist, B_2.truelist);$ $B.falselist = B_2.falselist;\ \}$
2)	$B \to B_1\ \&\&\ \boxed{M}\ B_2$	$\{\ backpatch(B_1.truelist, M.instr);$ $B.truelist = B_2.truelist;$ $B.falselist = merge(B_1.falselist, B_2.falselist);\ \}$
3)	$B \to\ !\ B_1$	$\{\ B.truelist = B_1.falselist;$ $B.falselist = B_1.truelist;\ \}$
4)	$B \to (\ B_1\ )$	$\{\ B.truelist = B_1.truelist;$ $B.falselist = B_1.falselist;\ \}$
5)	$B \to E_1\ \mathbf{rel}\ E_2$	$\{\ B.truelist = makelist(nextinstr);$ $B.falselist = makelist(nextinstr + 1);$ $gen($`'if'` $E_1.addr\ \mathbf{rel}.op\ E_2.addr$ `'goto _'`$);$ $gen($`'goto _'`$);\ \}$
6)	$B \to \mathbf{true}$	$\{\ B.truelist = makelist(nextinstr);$ $gen($`'goto _'`$);\ \}$
7)	$B \to \mathbf{false}$	$\{\ B.falselist = makelist(nextinstr);$ $gen($`'goto _'`$);\ \}$
8)	$\boxed{M \to \epsilon}$	$\{\ M.instr = nextinstr;\ \}$

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

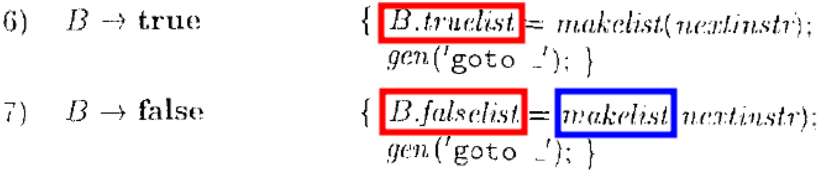


6) $B \to$ **true** $\{ \boxed{B.truelist} = makelist(nextinstr);$
$gen($'goto _'$); \}$

7) $B \to$ **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

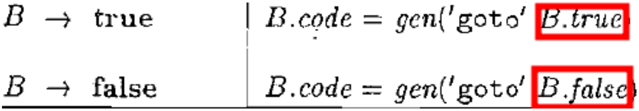
综合属性 $B.truelist$ 保存 需要跳转到 $B.true$ 的指令地址



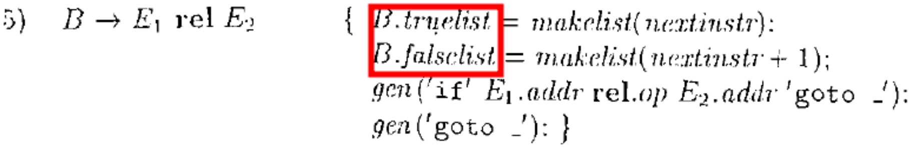
6) $B \rightarrow$ **true** { $B.truelist = makelist(nextinstr)$;
gen('goto _'); }

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

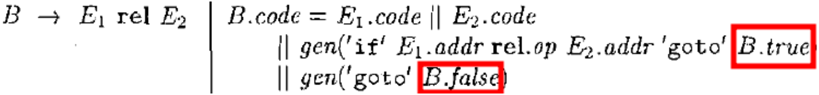
综合属性 $B.falselist$ 保存 需要跳转到 $B.false$ 的指令地址



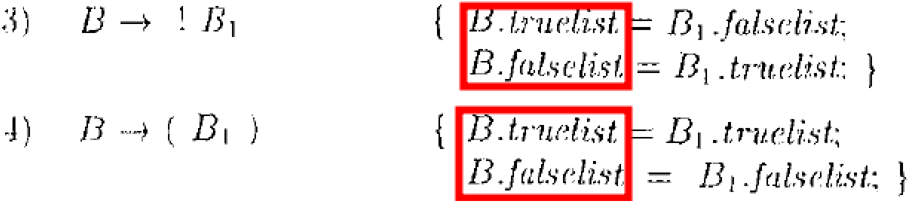
$B \rightarrow$ **true**	$B.code = gen($'goto' $B.true)$
$B \rightarrow$ **false**	$B.code = gen($'goto' $B.false)$



5) $B \to E_1$ **rel** E_2 $\{$ $\boxed{B.truelist}$ $= makelist(nextinstr);$
$\boxed{B.falselist}$ $= makelist(nextinstr + 1);$
$gen($'`if`' $E_1.addr$ **rel**$.op$ $E_2.addr$ '`goto _`'$);$
$gen($'`goto _`'$);$ $\}$

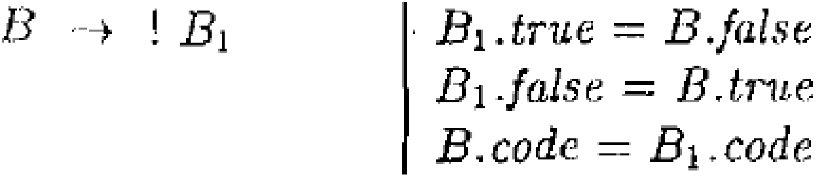


$B \to E_1$ **rel** E_2 | $B.code = E_1.code \;||\; E_2.code$
$||\; gen($'`if`' $E_1.addr$ **rel**$.op$ $E_2.addr$ '`goto`' $\boxed{B.true}$$)$
$||\; gen($'`goto`' $\boxed{B.false}$$)$

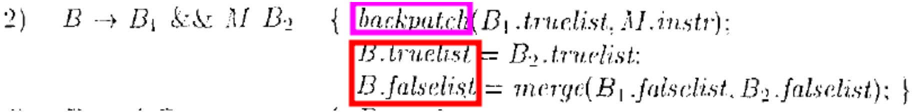


3) $B \rightarrow \; ! \; B_1$ $\qquad \{ \; B.truelist = B_1.falselist; \quad B.falselist = B_1.truelist; \; \}$

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



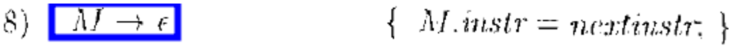
$$B \rightarrow \; ! \; B_1 \qquad \left| \begin{array}{l} B_1.true = B.false \\ B_1.false = B.true \\ B.code = B_1.code \end{array} \right.$$



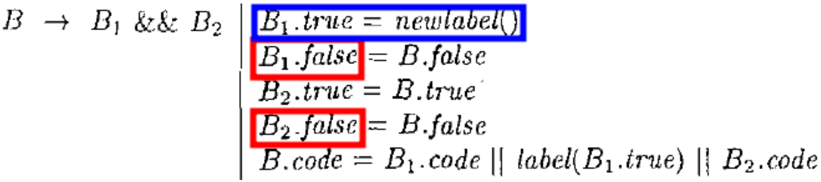
2) $B \to B_1 \;\&\&\; M \; B_2$ $\{\ \boxed{backpatch}(B_1.truelist, M.instr);$

$\boxed{B.truelist} = B_2.truelist;$

$\boxed{B.falselist} = merge(B_1.falselist, B_2.falselist);\ \}$



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



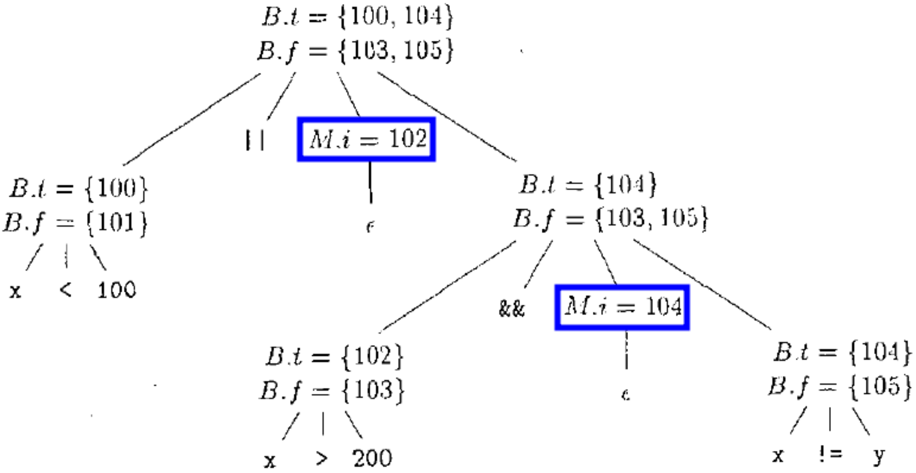
$$B \to B_1 \;\&\&\; B_2 \quad \begin{array}{|l} \boxed{B_1.true = newlabel()} \\ \boxed{B_1.false} = B.false \\ B_2.true = B.true \\ \boxed{B_2.false} = B.false \\ B.code = B_1.code \;||\; label(B_1.true) \;||\; B_2.code \end{array}$$

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

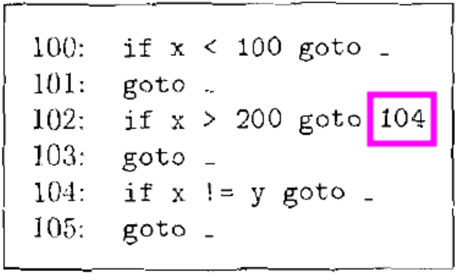
$$8)\quad \boxed{M \rightarrow \epsilon} \qquad \{\ M.instr = nextinstr;\ \}$$

$$B \rightarrow B_1 \;||\; B_2 \quad \left|\ \begin{array}{l} \boxed{B_1.true} = B.true \\ \boxed{B_1.false = newlabel()} \\ \boxed{B_2.true} = B.true \\ B_2.false = B.false \\ B.code = B_1.code \;||\; label(B_1.false) \;||\; B_2.code \end{array}\right.$$

x < 100 || x > 200 && x != y

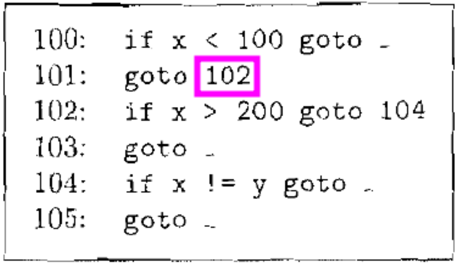


$B.t = \{100, 104\}$
$B.f = \{103, 105\}$
||
$M.i = 102$
ϵ
$B.t = \{100\}$
$B.f = \{101\}$
x
<
100
$B.t = \{104\}$
$B.f = \{103, 105\}$
&&
$M.i = 104$
ϵ
$B.t = \{102\}$
$B.f = \{103\}$
x
>
200
$B.t = \{104\}$
$B.f = \{105\}$
x
!=
y



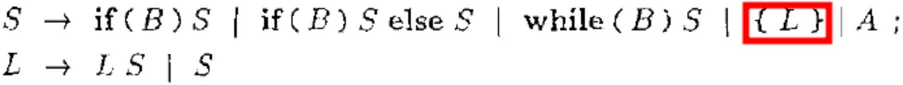
```
100:   if x < 100 goto _
101:   goto _
102:   if x > 200 goto 104
103:   goto _
104:   if x != y goto _
105:   goto _
```

a) 将 104 回填到指令 102 中之后



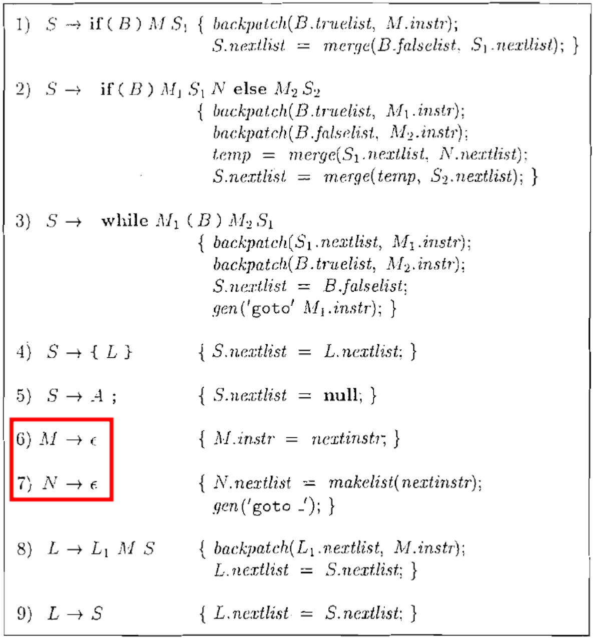
```
100:   if x < 100 goto _
101:   goto 102
102:   if x > 200 goto 104
103:   goto _
104:   if x != y goto _
105:   goto _
```

b) 将 102 回填到指令 101 中之后



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

$$L \to L\,S \mid S$$



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

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

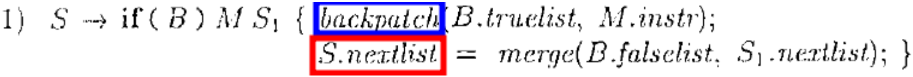
5) $S \rightarrow A\ ;$ $\{$ $S.nextlist\ =\ \textbf{null};$ $\}$

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

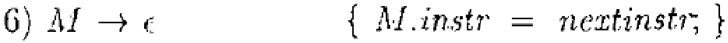
7) $N \rightarrow \epsilon$ $\{$ $N.nextlist\ =\ makelist(nextinstr);$
$gen(\text{'goto \_'});$ $\}$

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

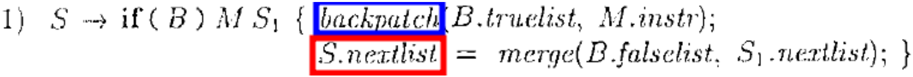
9) $L \rightarrow S$ $\{$ $L.nextlist\ =\ S.nextlist;$ $\}$



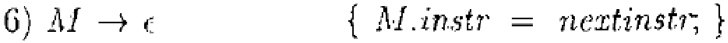
$$1)\quad S \rightarrow \textbf{if}\,(\,B\,)\; M\; S_1 \quad \{\; \boxed{backpatch}(B.truelist,\;\; M.instr);$$
$$\boxed{S.nextlist} \;=\; merge(B.falselist,\;\; S_1.nextlist);\;\}$$



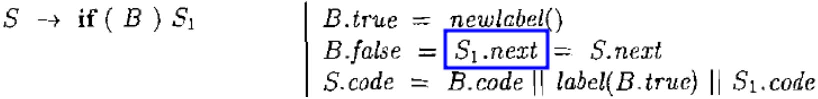
$$6)\; M \;\rightarrow\; \epsilon \qquad\qquad\qquad \{\; M.instr \;=\; nextinstr;\;\}$$



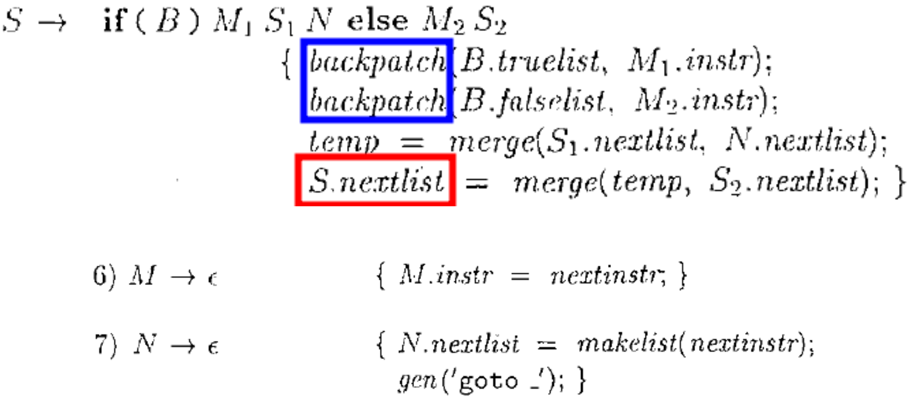
$$1)\quad S \rightarrow \textbf{if}\ (\ B\ )\ M\ S_1 \quad \{\ \boxed{backpatch}(B.truelist,\ M.instr);$$
$$\boxed{S.nextlist} = merge(B.falselist,\ S_1.nextlist);\ \}$$



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



$$S \rightarrow \textbf{if}\ (\ B\ )\ S_1 \quad \left|\ \begin{array}{l} B.true = newlabel() \\ B.false = \boxed{S_1.next} = S.next \\ S.code = B.code \mathrel{||} label(B.true) \mathrel{||} S_1.code \end{array}\right.$$

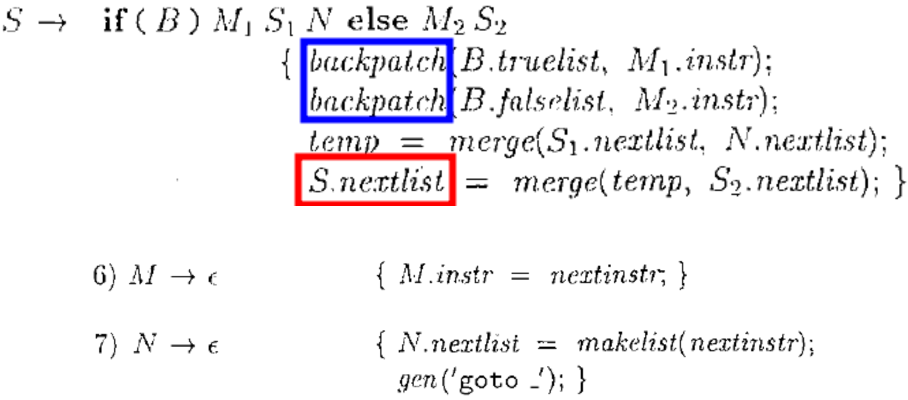


$S \rightarrow$ **if** (B) M_1 S_1 N **else** M_2 S_2

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

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

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



$$S \rightarrow \textbf{if} \; (\; B \;) \; M_1 \; S_1 \; N \; \textbf{else} \; M_2 \; S_2$$

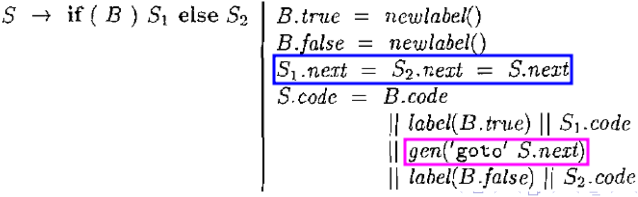
$$\{ \; \boxed{\begin{array}{l} backpatch \\ backpatch \end{array}} \begin{array}{l} (B.truelist, \;\; M_1.instr); \\ (B.falselist, \;\; M_2.instr); \end{array}$$

$$temp \; = \; merge(S_1.nextlist, \;\; N.nextlist);$$

$$\boxed{S.nextlist} \; = \; merge(temp, \;\; S_2.nextlist); \; \}$$

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

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

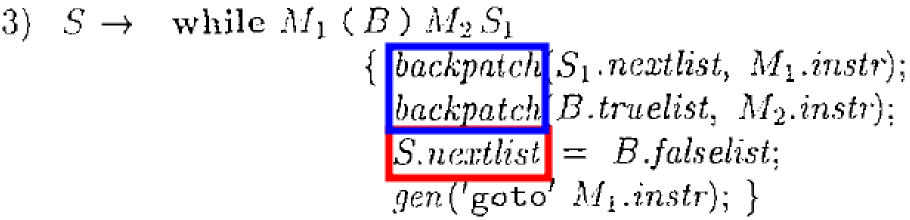


$$S \; \rightarrow \; \textbf{if} \; (\; B \;) \; S_1 \; \textbf{else} \; S_2 \quad \left| \; \begin{array}{l} B.true \; = \; newlabel() \\ B.false \; = \; newlabel() \\ \boxed{S_1.next \; = \; S_2.next \; = \; S.next} \\ S.code \; = \; B.code \\ \qquad\qquad || \; label(B.true) \; || \; S_1.code \\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

$$3)\quad S \rightarrow \quad \textbf{while}\ M_1\ (\ B\ )\ M_2\ S_1$$

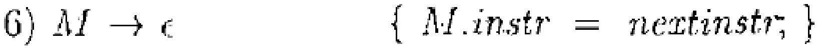
$$\begin{array}{l} \{\ \boxed{backpatch}(S_1.nextlist,\ M_1.instr); \\ \quad \boxed{backpatch}(B.truelist,\ M_2.instr); \\ \quad \boxed{S.nextlist} = B.falselist; \\ \quad gen(\text{'goto'}\ M_1.instr);\ \} \end{array}$$

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

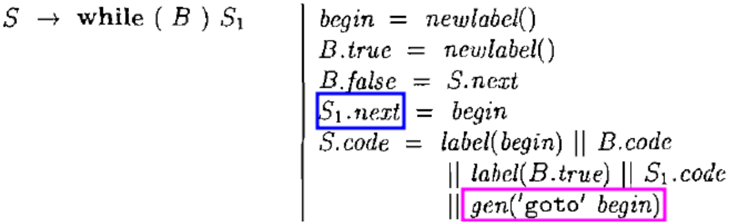


3) $S \rightarrow$ **while** M_1 (B) $M_2\, S_1$

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

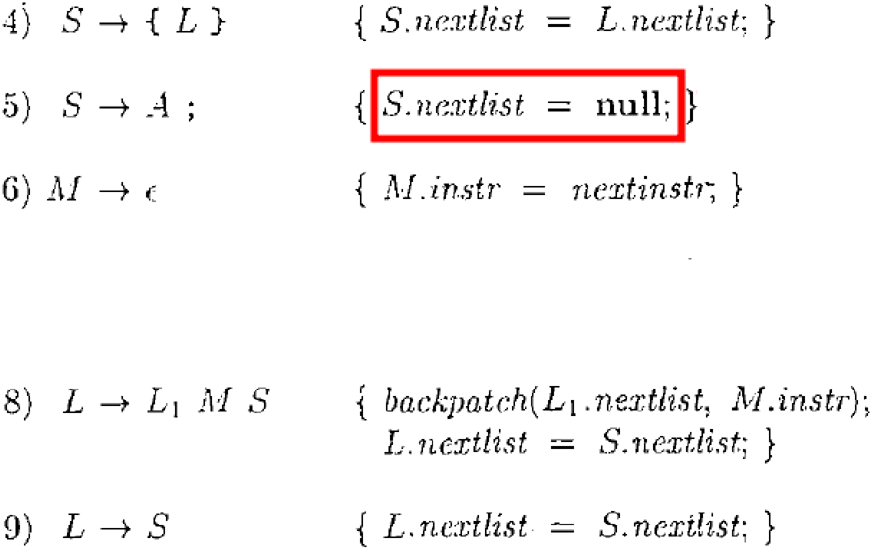


6) $M \rightarrow \epsilon$ $\qquad \{\ M.instr = nextinstr;\ \}$



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

$$\begin{array}{l} begin = newlabel() \\ B.true = newlabel() \\ B.false = S.next \\ \boxed{S_1.next} = begin \\ S.code = label(begin) \,||\, B.code \\ \qquad\qquad ||\, label(B.true) \,||\, S_1.code \\ \qquad\qquad ||\, \boxed{gen('\mathrm{goto}'\ begin)} \end{array}$$



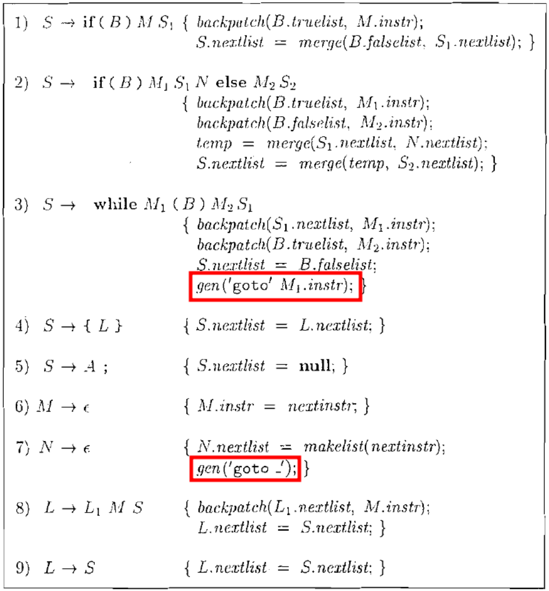
4) $S \rightarrow \{ \ L \ \}$ $\quad \{ \ S.nextlist = L.nextlist; \ \}$

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

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

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

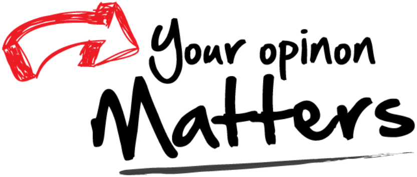
9) $L \rightarrow S$ $\quad \{ \ L.nextlist = S.nextlist; \ \}$



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) $S \rightarrow \textbf{while}\ M_1\ (\ B\ )\ M_2\ S_1$	$\{\ backpatch(S_1.nextlist,\ M_1.instr);$ $backpatch(B.truelist,\ M_2.instr);$ $S.nextlist\ =\ B.falselist;$ $\boxed{gen('\texttt{goto}'\ M_1.instr);}\ \}$
4) $S \rightarrow \{\ L\ \}$	$\{\ S.nextlist\ =\ L.nextlist;\ \}$
5) $S \rightarrow A\ ;$	$\{\ S.nextlist\ =\ \textbf{null};\ \}$
6) $M \rightarrow \epsilon$	$\{\ M.instr\ =\ nextinstr;\ \}$
7) $N \rightarrow \epsilon$	$\{\ N.nextlist\ =\ makelist(nextinstr);$ $\boxed{gen('\texttt{goto}\ \_');}\ \}$
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。

Thank You!



Office 926

hfwei@nju.edu.cn