

四、中间代码生成

(12. 控制流语句翻译中的地址回填技术)

魏恒峰

hfwei@nju.edu.cn

2024 年 05 月 17 日



使用标签标记跳转目标

```
1   goto L2
2   L3:
3   goto L1
4   L2:
5   ASSIGN
6   L1:
```

```
1 if (true || false) {
2     a = b;
3 }
```

Java Bytecode: 使用地址值作为跳转目标

```
outer:  
for (int i = 2; i < 1000; i++) {  
    for (int j = 2; j < i; j++) {  
        if (i % j == 0)  
            continue outer;  
    }  
    System.out.println (i);  
}
```

```
0:  iconst_2  
1:  istore_1  
2:  iload_1  
3:  sipush 1000  
6:  if_icmpge 44  
9:  iconst_2  
10: istore_2  
11: iload_2  
12: iload_1  
13: if_icmpge 31  
16: iload_1  
17: iload_2  
18: irem  
19: ifne 25  
22: goto 38  
25: iinc 2, 1  
28: goto 11  
31: getstatic #84;  
34: iload_1  
35: invokevirtual #85;  
38: iinc 1, 1  
41: goto 2  
44: return
```

31

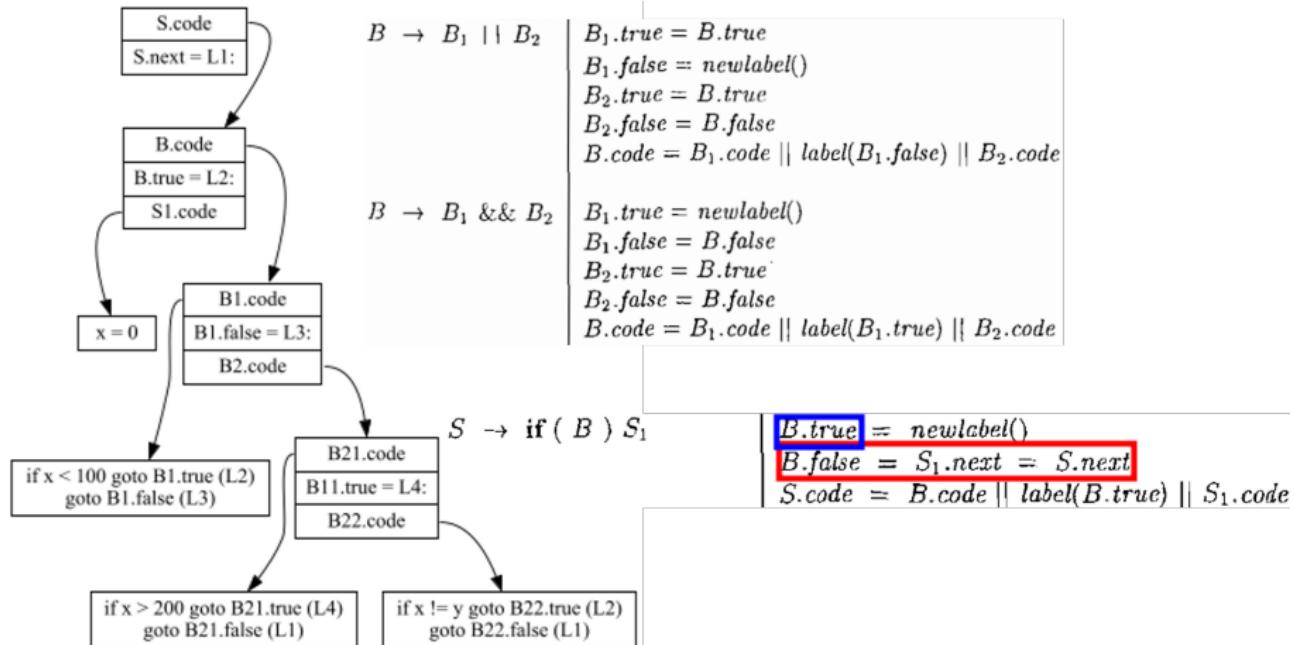
Q : 如何在一趟扫描中生成跳转目标的地址?

```
if (x < 100 || x > 200 && x != y) x = 0;
```

```
100: if x < 100 goto L2
101: goto L3
L3 102: if x > 200 goto L4
        103: goto L1
L4 104: if x != y goto L2
        105: goto L1
L2 106: x = 0
L1 : 107:
```

L_1 标签的位置是由 $P \rightarrow S (\rightarrow \text{if } B S_1)$ 确定的,
但是, 生成 B 所对应的中间代码 `goto L1` 时, 尚不知道 L_1 的地址

```
if (x < 100 || x > 200 && x != y) x = 0;
```



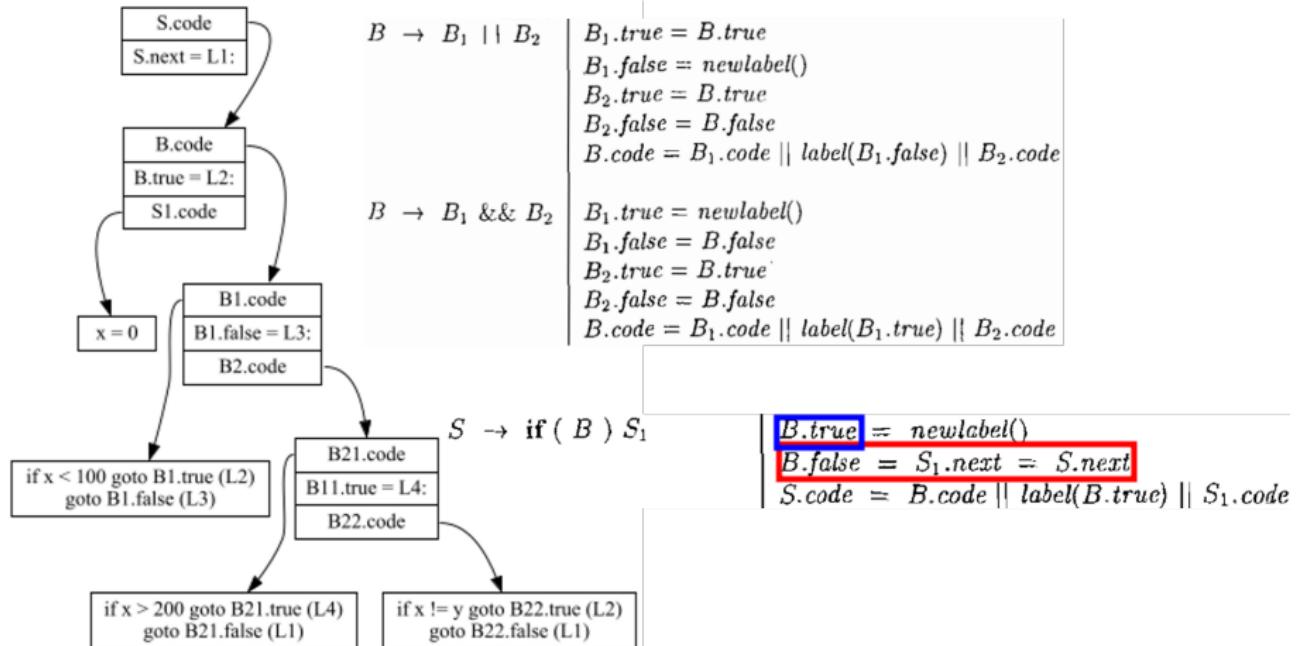
$$B_{21}.\text{false} \leftarrow B_2.\text{false} \leftarrow B.\text{false} \leftarrow S.\text{next} = L_1$$

$$B_{22}.\text{false} \leftarrow B_2.\text{false} \leftarrow B.\text{false} \leftarrow S.\text{next} = L_1$$



諸君...我有個大膽的想法!!

```
if (x < 100 || x > 200 && x != y) x = 0;
```



$$\{103\} = B_{21}.false \rightarrow B_2.false \rightarrow B.false \rightarrow S.next = 107$$

$$\{105\} = B_{22}.false \rightarrow B_2.false \rightarrow B.false \rightarrow S.next = 107$$

子节点暂时不指定跳转指令的目标地址

```
100:if x < 100 goto L2
101:goto L3
L3:102:if x > 200 goto L4
103:goto 
L4:104:if x != y goto L2
105:goto 
L2:106:x = 0
L1:107:
```



待祖先节点能够确定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



祖先节点通过**综合属性**收集子节点中具有相同目标的跳转指令

南北爱情故事







B.truelist 保存需要跳转到 *B.true* 标签的指令

B.falselist 保存需要跳转到 *B.false* 标签的指令

- 1) $S \rightarrow \text{if}(B) M S_1 \{ \text{backpatch}(B.\text{truelist}, M.\text{instr}); S.\text{nextlist} = \text{merge}(B.\text{falselist}, S_1.\text{nextlist}); \}$
- 2) $S \rightarrow \text{if}(B) M_1 S_1 \text{else } M_2 S_2 \{ \text{backpatch}(B.\text{truelist}, M_1.\text{instr}); \text{backpatch}(B.\text{falselist}, M_2.\text{instr}); \text{temp} = \text{merge}(S_1.\text{nextlist}, N.\text{nextlist}); S.\text{nextlist} = \text{merge}(\text{temp}, S_2.\text{nextlist}); \}$
- 3) $S \rightarrow \text{while } M_1 (B) M_2 S_1 \{ \text{backpatch}(S_1.\text{nextlist}, M_1.\text{instr}); \text{backpatch}(B.\text{truelist}, M_2.\text{instr}); S.\text{nextlist} = B.\text{falselist}; \text{gen('goto' } M_1.\text{instr}); \}$
- 4) $S \rightarrow \{ L \} \quad \{ S.\text{nextlist} = L.\text{nextlist}; \}$
- 5) $S \rightarrow A ; \quad \{ S.\text{nextlist} = \text{null}; \}$
- 6) $M \rightarrow \epsilon \quad \{ M.\text{instr} = \text{nextinstr}; \}$
- 7) $N \rightarrow \epsilon \quad \{ N.\text{nextlist} = \text{makelist(nextinstr)}; \text{gen('goto' '}); \}$
- 8) $L \rightarrow L_1 M S \quad \{ \text{backpatch}(L_1.\text{nextlist}, M.\text{instr}); L.\text{nextlist} = S.\text{nextlist}; \}$
- 9) $L \rightarrow S \quad \{ L.\text{nextlist} = S.\text{nextlist}; \}$

S/L.nextlist 保存需要跳转到 *S/L.next* 标签的指令

为左部非终结符 B 计算综合属性 $B.trueList$ 与 $B.falseList$

为左部非终结符 S/L 计算综合属性 $S/L.nextList$

- 1) $S \rightarrow \text{if}(B) M S_1 \{ \text{backpatch}(B.trueList, M.instr); S.nextList = \text{merge}(B.falseList, S_1.nextList); \}$
- 2) $S \rightarrow \text{if}(B) M_1 S_1 N \text{ else } M_2 S_2 \{ \text{backpatch}(B.trueList, M_1.instr); \text{backpatch}(B.falseList, M_2.instr); \text{temp} = \text{merge}(S_1.nextList, N.nextList); S.nextList = \text{merge}(\text{temp}, S_2.nextList); \}$
- 3) $S \rightarrow \text{while } M_1 (B) M_2 S_1 \{ \text{backpatch}(S_1.nextList, M_1.instr); \text{backpatch}(B.trueList, M_2.instr); S.nextList = B.falseList; \text{gen('goto' M_1.instr);} \}$
- 4) $S \rightarrow \{ L \} \quad \{ S.nextList = L.nextList; \}$
- 5) $S \rightarrow A ; \quad \{ S.nextList = \text{null}; \}$
- 6) $M \rightarrow \epsilon \quad \{ M.instr = nextinstr; \}$
- 7) $N \rightarrow \epsilon \quad \{ N.nextList = makelist(nextinstr); \text{gen('goto' _);} \}$
- 8) $L \rightarrow L_1 M S \quad \{ \text{backpatch}(L_1.nextList, M.instr); L.nextList = S.nextList; \}$
- 9) $L \rightarrow S \quad \{ L.nextList = S.nextList; \}$

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)

1) $B \rightarrow B_1 \sqcup \boxed{M} B_2$

2) $B \rightarrow B_1 \&& \boxed{M} B_2$

3) $B \rightarrow ! B_1$

4) $B \rightarrow (B_1)$

5) $B \rightarrow E_1 \text{ rel } E_2$

6) $B \rightarrow \mathbf{true}$

7) $B \rightarrow \mathbf{false}$

8) $\boxed{M \rightarrow \epsilon}$

$B.\text{truelist}$ 保存需要跳转到 $B.\text{true}$ 标签的指令

- 6) $B \rightarrow \text{true}$ { $B.\text{truelist} = \text{makelist}(\text{nextinstr});$
 $\quad \text{gen}('goto' __); \}$
- 7) $B \rightarrow \text{false}$ { $B.\text{falselist} = \text{makelist}(\text{nextinstr});$
 $\quad \text{gen}('goto' __); \}$

$B.\text{falselist}$ 保存需要跳转到 $B.\text{false}$ 标签的指令

- $B \rightarrow \text{true}$ | $B.\text{code} = \text{gen}('goto' B.\text{true})$
- $B \rightarrow \text{false}$ | $B.\text{code} = \text{gen}('goto' B.\text{false})$

5) $B \rightarrow E_1 \text{ rel } E_2$ $\{ \begin{array}{l} B.\text{truelist} = \text{makelist}(nextinstr); \\ B.\text{falselist} = \text{makelist}(nextinstr + 1); \\ \text{gen('if' } E_1.\text{addr rel.op } E_2.\text{addr 'goto' } _)'; \\ \text{gen('goto' } _'): \end{array} \}$

$B \rightarrow E_1 \text{ rel } E_2 \quad \left| \begin{array}{l} B.\text{code} = E_1.\text{code} \parallel E_2.\text{code} \\ \parallel \text{gen('if' } E_1.\text{addr rel.op } E_2.\text{addr 'goto' } B.\text{true}) \\ \parallel \text{gen('goto' } B.\text{false}) \end{array} \right.$

- 3) $B \rightarrow !B_1$ { $B.trueList = B_1.falseList;$
 $B.falseList = B_1.trueList;$ }
- 4) $B \rightarrow (B_1)$ { $B.trueList = B_1.trueList;$
 $B.falseList = B_1.falseList;$ }

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

2) $B \rightarrow B_1 \ \&\& \ M \ B_2 \quad \{ \begin{array}{l} backpatch(B_1.trueList, M.instr); \\ B.trueList = B_2.trueList; \\ B.falseList = merge(B_1.falseList, B_2.falseList); \end{array} \}$

8) $M \rightarrow \epsilon \quad \{ \begin{array}{l} M.instr = nextinstr; \end{array} \}$

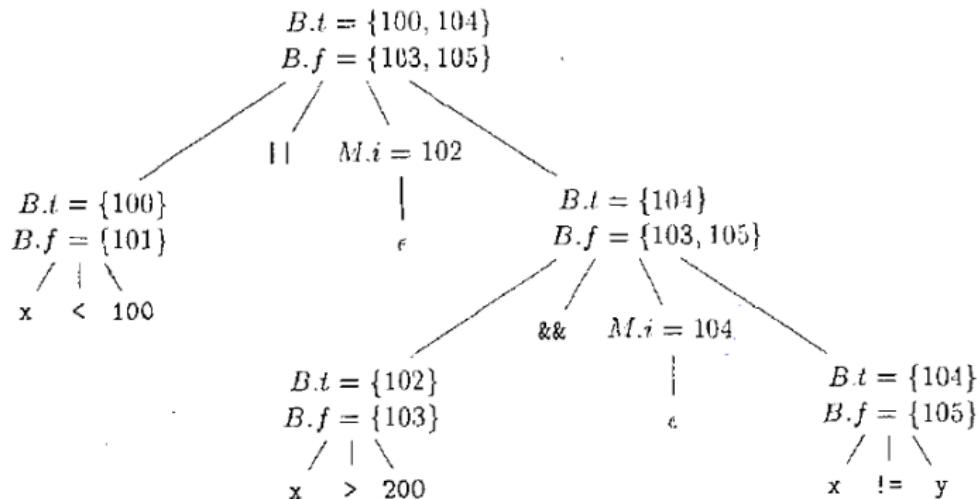
$B \rightarrow B_1 \ \&\& \ B_2 \quad \left| \begin{array}{l} B_1.true = newlabel() \\ B_1.false = B.false \\ B_2.true = B.true \\ B_2.false = B.false \\ B.code = B_1.code || label(B_1.true) || B_2.code \end{array} \right.$

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

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

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

- 1) $B \rightarrow B_1 \parallel M B_2$ { backpatch($B_1.falselist, M.instr$);
 $B.truelist = merge(B_1.truelist, B_2.truelist)$;
 $B.falselist = B_2.falselist$; }
- 2) $B \rightarrow B_1 \&& M B_2$ { backpatch($B_1.truelist, M.instr$);
 $B.truelist = B_2.truelist$;
 $B.falselist = merge(B_1.falselist, B_2.falselist)$; }
- 5) $B \rightarrow E_1 \text{ rel } E_2$ { $B.truelist = makelist(nextinstr)$;
 $B.falselist = makelist(nextinstr + 1)$;
 $gen('if' E_1.addr \text{ rel.op } E_2.addr 'goto _')$;
 $gen('goto _')$ }



```
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

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

2) $S \rightarrow \text{if}(B) M_1 S_1 [N] \text{else } M_2 S_2$

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

4) $S \rightarrow \{ L \}$

5) $S \rightarrow A ;$

6) $M \rightarrow \epsilon$

7) $[N \rightarrow \epsilon]$

8) $L \rightarrow L_1 M S$

9) $L \rightarrow S$

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

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

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

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

```
{ backpatch(B.truelist, M1.instr);
  backpatch(B.falselist, M2.instr);
  temp = merge(S1.nextlist, N.nextlist);
  S.nextlist = merge(temp, S2.nextlist); }
```

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

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

$S \rightarrow \text{if}(B) S_1 \text{ else } S_2$

```
B.true = newlabel()
B.false = newlabel()
S1.next = S2.next = S.next
S.code = B.code
  || label(B.true) || S1.code
  || gen('goto' S.next)
  || label(B.false) || S2.code
```

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

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

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

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

$$\left\{ \begin{array}{l} \text{begin} = \text{newlabel}() \\ B.\text{true} = \text{newlabel}() \\ B.\text{false} = S.\text{next} \\ S_1.\text{next} = \text{begin} \\ S.\text{code} = \text{label(begin)} \parallel B.\text{code} \\ \quad \parallel \text{label}(B.\text{true}) \parallel S_1.\text{code} \\ \quad \parallel \boxed{\text{gen('goto' begin)}} \end{array} \right.$$

- 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) $S \rightarrow \{ L \}$ $\{ S.nextlist = L.nextlist; \}$
- 5) $S \rightarrow A ;$ $\{ S.nextlist = \text{null}; \}$

- 1) $S \rightarrow \text{if}(B) M S_1 \{ \text{backpatch}(B.\text{truelist}, M.\text{instr}); S.\text{nextlist} = \text{merge}(B.\text{falselist}, S_1.\text{nextlist}); \}$
- 2) $S \rightarrow \text{if}(B) M_1 S_1 N \text{ else } M_2 S_2$
 $\quad \{ \text{backpatch}(B.\text{truelist}, M_1.\text{instr});$
 $\quad \text{backpatch}(B.\text{falselist}, M_2.\text{instr});$
 $\quad \text{temp} = \text{merge}(S_1.\text{nextlist}, N.\text{nextlist});$
 $\quad S.\text{nextlist} = \text{merge}(\text{temp}, S_2.\text{nextlist}); \}$
- 3) $S \rightarrow \text{while } M_1 (B) M_2 S_1$
 $\quad \{ \text{backpatch}(S_1.\text{nextlist}, M_1.\text{instr});$
 $\quad \text{backpatch}(B.\text{truelist}, M_2.\text{instr});$
 $\quad S.\text{nextlist} = B.\text{falselist};$
 $\quad \boxed{\text{gen('goto' } M_1.\text{instr}); \}}$
- 4) $S \rightarrow \{ L \} \quad \{ S.\text{nextlist} = L.\text{nextlist}; \}$
- 5) $S \rightarrow A ; \quad \{ S.\text{nextlist} = \text{null}; \}$
- 6) $M \rightarrow \epsilon \quad \{ M.\text{instr} = \text{nextinstr}; \}$
- 7) $N \rightarrow \epsilon \quad \{ N.\text{nextlist} = \text{makelist(nextinstr)};$
 $\quad \boxed{\text{gen('goto' ?); \}}$
- 8) $L \rightarrow L_1 M S \quad \{ \text{backpatch}(L_1.\text{nextlist}, M.\text{instr});$
 $\quad L.\text{nextlist} = S.\text{nextlist}; \}$
- 9) $L \rightarrow S \quad \{ L.\text{nextlist} = S.\text{nextlist}; \}$

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  $\geq$  60 then
3:     while true do
4:       print “WanSui”
5:     else
6:       print “Sad”
```

- 2) $S \rightarrow \text{if } (B) M_1 S_1 N \text{ else } M_2 S_2$
- $$\{ \text{backpatch}(B.\text{truelist}, M_1.\text{instr});$$
- $$\text{backpatch}(B.\text{falselist}, M_2.\text{instr});$$
- $$temp = \text{merge}(S_1.\text{nextlist}, N.\text{nextlist});$$
- $$S.\text{nextlist} = \text{merge}(temp, S_2.\text{nextlist}); \}$$
- 3) $S \rightarrow \text{while } M_1 (B) M_2 S_1$
- $$\{ \text{backpatch}(S_1.\text{nextlist}, M_1.\text{instr});$$
- $$\text{backpatch}(B.\text{truelist}, M_2.\text{instr});$$
- $$S.\text{nextlist} = B.\text{falselist};$$
- $$\text{gen('goto' } M_1.\text{instr}); \}$$
- 6) $M \rightarrow \epsilon \quad \{ M.\text{instr} = \text{nextinstr}; \}$
- 7) $N \rightarrow \epsilon \quad \{ N.\text{nextlist} = \text{makelist(nextinstr)};$

$$\text{gen('goto' } \cdot \text{'); } \}$$
- 5) $B \rightarrow E_1 \text{ rel } E_2 \quad \{ B.\text{truelist} = \text{makelist(nextinstr)};$

$$B.\text{falselist} = \text{makelist(nextinstr} + 1\text{)};$$

$$\text{gen('if' } E_1.\text{addr rel.op } E_2.\text{addr 'goto' } \cdot \text{'); }$$

$$\text{gen('goto' } \cdot \text{'); } \}$$
- 6) $B \rightarrow \text{true} \quad \{ B.\text{truelist} = \text{makelist(nextinstr)};$

$$\text{gen('goto' } \cdot \text{'); } \}$$

B.truelist 保存需要跳转到 *B.true* 标签的指令

B.falselist 保存需要跳转到 *B.false* 标签的指令

- 1) $S \rightarrow \text{if}(B) M S_1 \{ \text{backpatch}(B.\text{truelist}, M.\text{instr}); S.\text{nextlist} = \text{merge}(B.\text{falselist}, S_1.\text{nextlist}); \}$
- 2) $S \rightarrow \text{if}(B) M_1 S_1 \text{ else } M_2 S_2 \{ \text{backpatch}(B.\text{truelist}, M_1.\text{instr}); \text{backpatch}(B.\text{falselist}, M_2.\text{instr}); \text{temp} = \text{merge}(S_1.\text{nextlist}, N.\text{nextlist}); S.\text{nextlist} = \text{merge}(\text{temp}, S_2.\text{nextlist}); \}$
- 3) $S \rightarrow \text{while } M_1 (B) M_2 S_1 \{ \text{backpatch}(S_1.\text{nextlist}, M_1.\text{instr}); \text{backpatch}(B.\text{truelist}, M_2.\text{instr}); S.\text{nextlist} = B.\text{falselist}; \text{gen('goto' } M_1.\text{instr}); \}$
- 4) $S \rightarrow \{ L \} \quad \{ S.\text{nextlist} = L.\text{nextlist}; \}$
- 5) $S \rightarrow A ; \quad \{ S.\text{nextlist} = \text{null}; \}$
- 6) $M \rightarrow \epsilon \quad \{ M.\text{instr} = \text{nextinstr}; \}$
- 7) $N \rightarrow \epsilon \quad \{ N.\text{nextlist} = \text{makelist(nextinstr)}; \text{gen('goto' '}); \}$
- 8) $L \rightarrow L_1 M S \quad \{ \text{backpatch}(L_1.\text{nextlist}, M.\text{instr}); L.\text{nextlist} = S.\text{nextlist}; \}$
- 9) $L \rightarrow S \quad \{ L.\text{nextlist} = S.\text{nextlist}; \}$

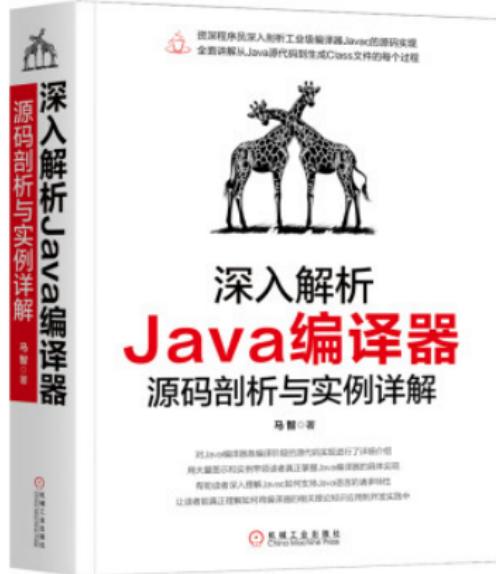
S/L.nextlist 保存需要跳转到 *S/L.next* 标签的指令

为左部非终结符 B 计算综合属性 $B.trueList$ 与 $B.falseList$

为左部非终结符 S/L 计算综合属性 $S/L.nextList$

- 1) $S \rightarrow \text{if}(B) M S_1 \{ \text{backpatch}(B.trueList, M.instr); S.nextList = \text{merge}(B.falseList, S_1.nextList); \}$
- 2) $S \rightarrow \text{if}(B) M_1 S_1 N \text{ else } M_2 S_2 \{ \text{backpatch}(B.trueList, M_1.instr); \text{backpatch}(B.falseList, M_2.instr); \text{temp} = \text{merge}(S_1.nextList, N.nextList); S.nextList = \text{merge}(\text{temp}, S_2.nextList); \}$
- 3) $S \rightarrow \text{while } M_1 (B) M_2 S_1 \{ \text{backpatch}(S_1.nextList, M_1.instr); \text{backpatch}(B.trueList, M_2.instr); S.nextList = B.falseList; \text{gen('goto' M_1.instr);} \}$
- 4) $S \rightarrow \{ L \} \quad \{ S.nextList = L.nextList; \}$
- 5) $S \rightarrow A ; \quad \{ S.nextList = \text{null}; \}$
- 6) $M \rightarrow \epsilon \quad \{ M.instr = nextinstr; \}$
- 7) $N \rightarrow \epsilon \quad \{ N.nextList = makelist(nextinstr); \text{gen('goto' _);} \}$
- 8) $L \rightarrow L_1 M S \quad \{ \text{backpatch}(L_1.nextList, M.instr); L.nextList = S.nextList; \}$
- 9) $L \rightarrow S \quad \{ L.nextList = S.nextList; \}$

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)



第17章 重要结构的字节码指令生成 527

17.1 控制转移指令与地址回填 527

17.1.1 认识控制转移指令 527

17.1.2 地址回填 529

17.2 语句的条件判断表达式 530

17.2.1 CondItem类 530

17.2.2 一元与二元条件判断表达式 533

17.2.3 三元条件判断表达式 534

17.3 if语句 536

17.4 循环语句 537

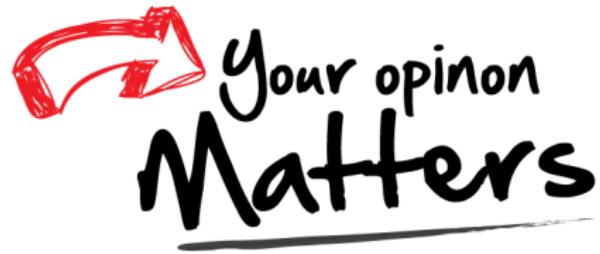
17.5 switch语句 539

17.6 异常与finally语句 545

17.6.1 异常的抛出 545

17.6.2 异常的捕获与finally语句 545

Thank You!



Office 926

hfwei@nju.edu.cn