编译原理

好好学习！！！天天向上！！！

任课老师：谢晓园 邮箱：xxie@whu.edu.cn 办公室：计算机学院E301

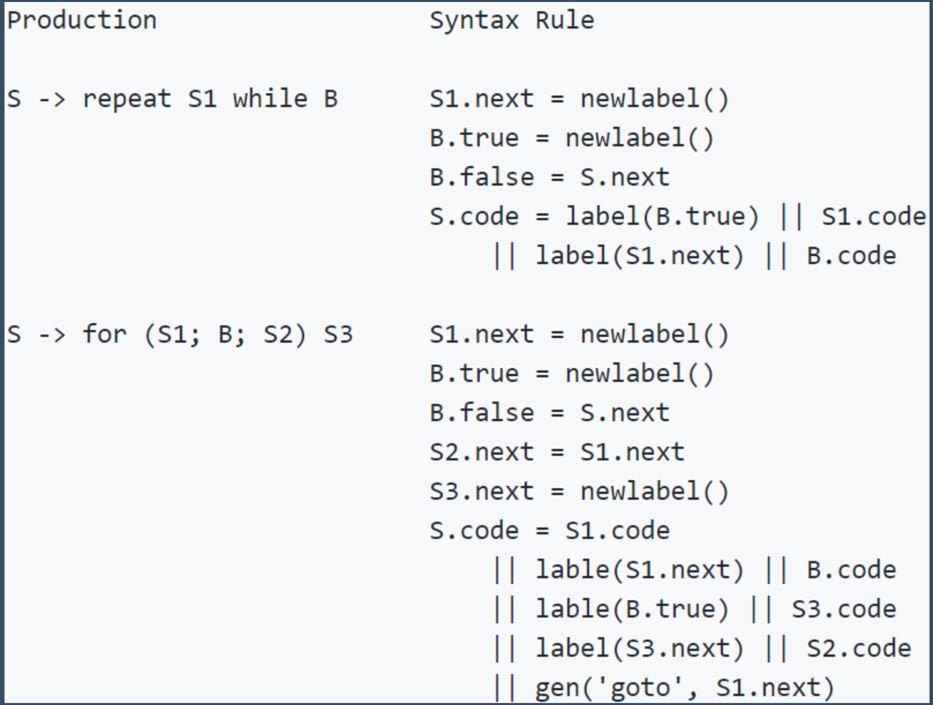
助教：黎源 邮箱：1445660426@qq.com

P263 6.6.1

在图6-36的语法制导定义中添加处理下列控制流构造的规则：

1) 一个repeat语句，**repeat** S **while** B

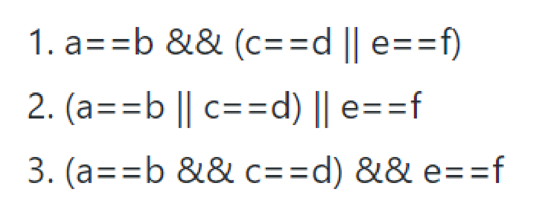
!2) 一个for循环语句，**for** (S_1 ; B ; S_2) S_3



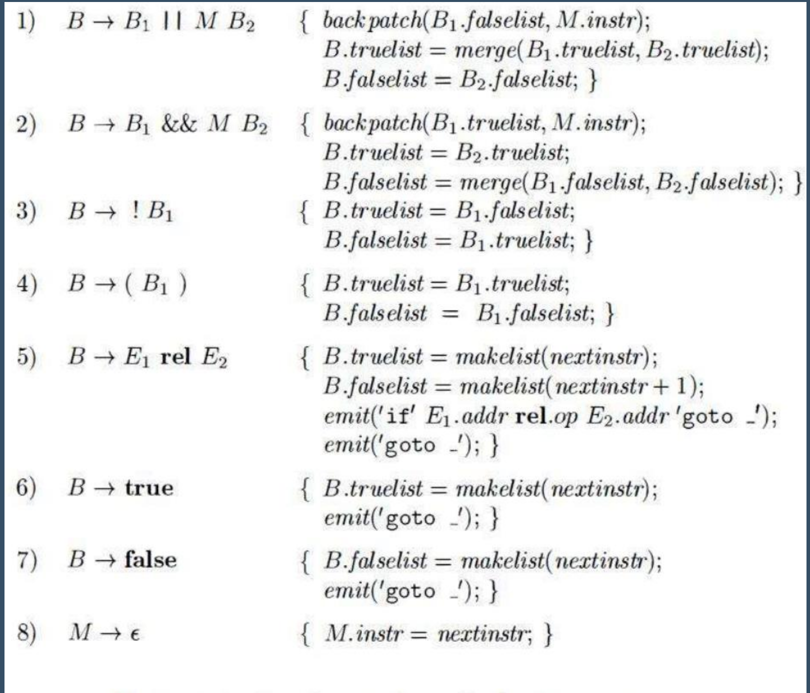
```
Production                  Syntax Rule

S -> repeat S1 while B      S1.next = newlabel()
                            B.true = newlabel()
                            B.false = S.next
                            S.code = label(B.true) || S1.code
                                || label(S1.next) || B.code

S -> for (S1; B; S2) S3     S1.next = newlabel()
                            B.true = newlabel()
                            B.false = S.next
                            S2.next = S1.next
                            S3.next = newlabel()
                            S.code = S1.code
                                || lable(S1.next) || B.code
                                || lable(B.true) || S3.code
                                || label(S3.next) || S2.code
                                || gen('goto', S1.next)
```

P268 6.7.1

使用图6-43中的翻译方案翻译下列表达式。给出每个子表达式的*truelist*和*falselist*。你可以假设第一条被生成的指令的地址是100。



1. a==b && (c==d || e==f)
2. (a==b || c==d) || e==f
3. (a==b && c==d) && e==f



1)	$B \to B_1 \mid\mid M\ B_2$	{ *backpatch*(B_1.*falselist*, M.*instr*); B.*truelist* = *merge*(B_1.*truelist*, B_2.*truelist*); B.*falselist* = B_2.*falselist*; }
2)	$B \to B_1\ \&\&\ M\ B_2$	{ *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)	$B \to E_1$ **rel** E_2	{ B.*truelist* = *makelist*(*nextinstr*);
		B.*falselist* = *makelist*(*nextinstr* + 1);
		emit('`if`' E_1.*addr* **rel**.*op* E_2.*addr* '`goto _`');
		emit('`goto _`'); }

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

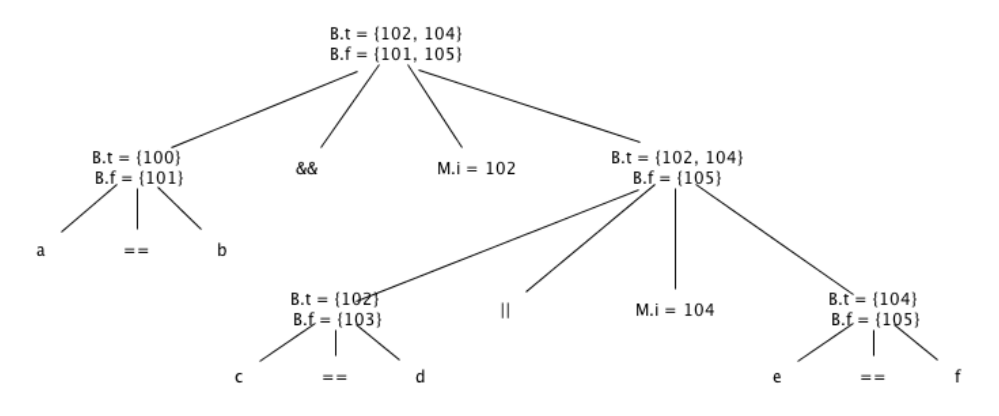
7)	$B \to$ **false**	{ B.*falselist* = *makelist*(*nextinstr*);
		emit('`goto _`'); }

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

Figure 6.43: Translation scheme for boolean expressions

P268 6.7.1

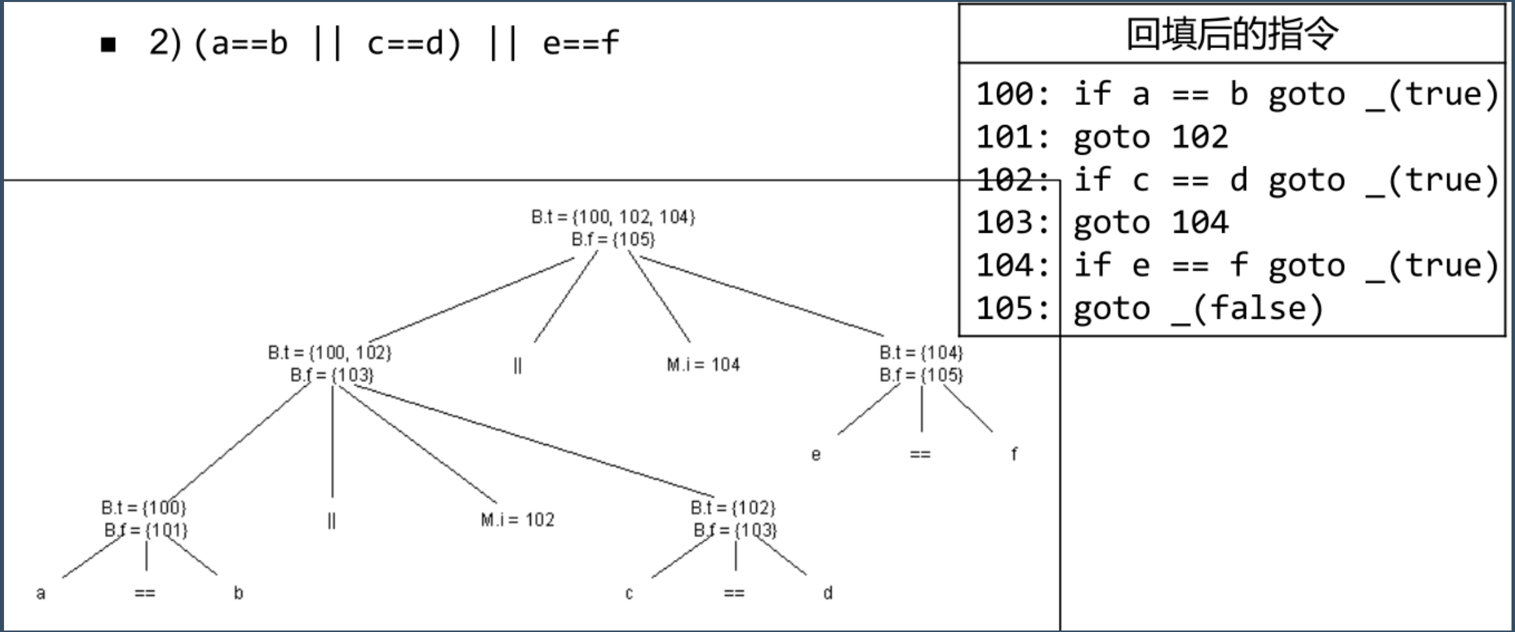
1) a==b && (c==d || e==f)



回填后的指令

```
100: if a == b goto 102
101: goto _(false)
102: if c == d goto _(true)
103: goto 104
104: if e == f goto _(true)
105: goto _(false)
```

P268 6.7.1

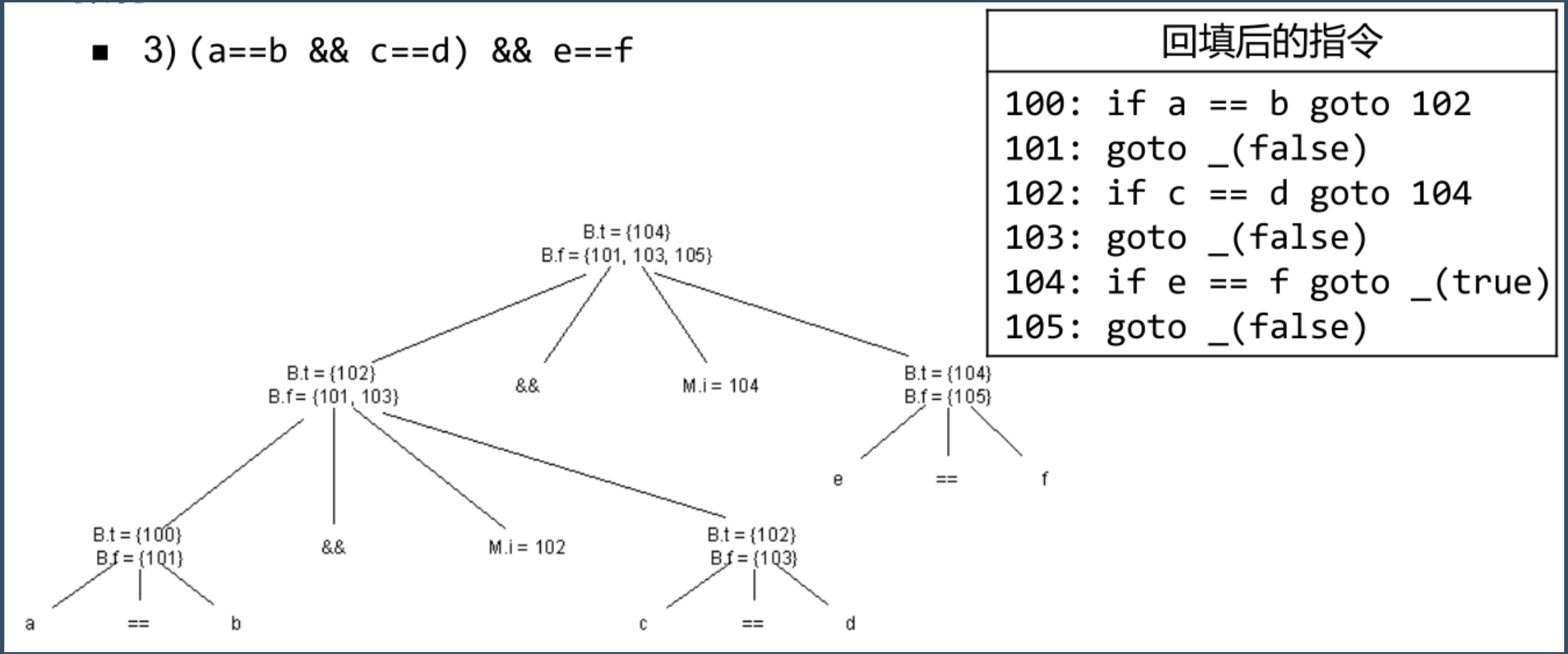


- 2) (a==b || c==d) || e==f

回填后的指令

```
100: if a == b goto _(true)
101: goto 102
102: if c == d goto _(true)
103: goto 104
104: if e == f goto _(true)
105: goto _(false)
```

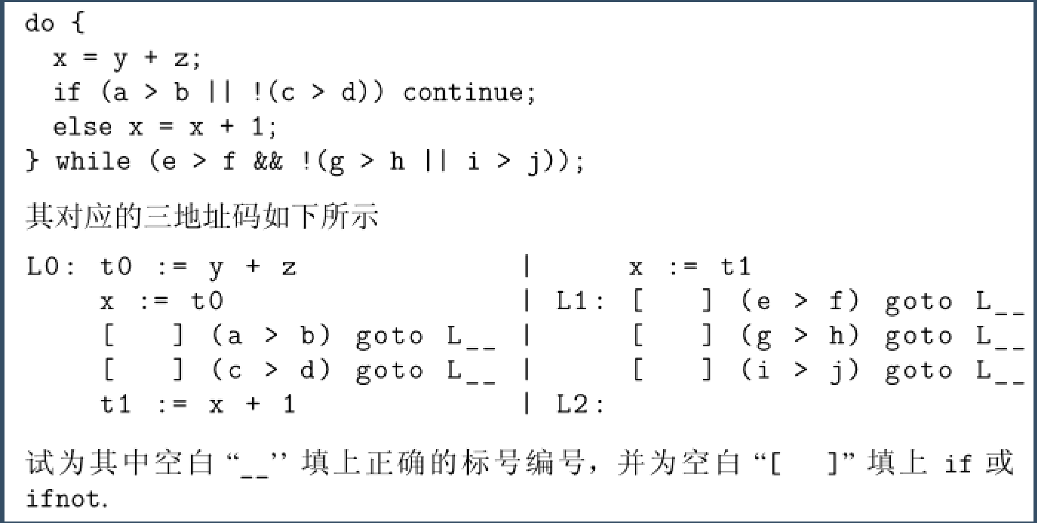
P268 6.7.1



- 3) (a==b && c==d) && e==f

回填后的指令

```
100: if a == b goto 102
101: goto _(false)
102: if c == d goto 104
103: goto _(false)
104: if e == f goto _(true)
105: goto _(false)
```

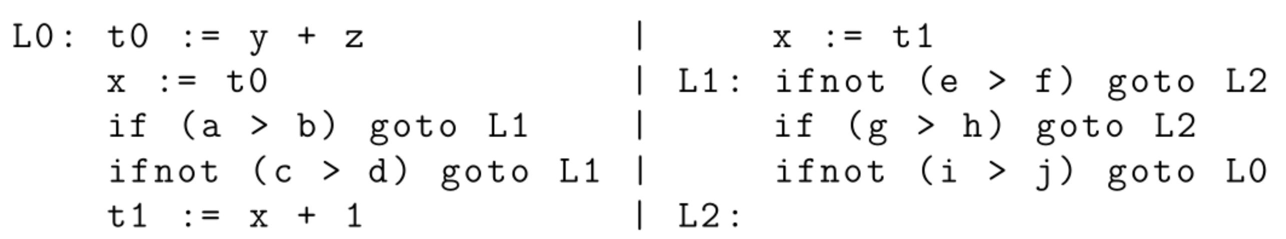


```
do {
  x = y + z;
  if (a > b || !(c > d)) continue;
  else x = x + 1;
} while (e > f && !(g > h || i > j));
```

其对应的三地址码如下所示

```
L0: t0  := y + z                | x := t1
    x  := t0                    | L1: [    ] (e > f) goto L__
    [    ] (a > b) goto L__     |     [    ] (g > h) goto L__
    [    ] (c > d) goto L__     |     [    ] (i > j) goto L__
    t1  := x + 1                | L2:
```

试为其中空白“__”填上正确的标号编号，并为空白“[   ]”填上 if 或 ifnot.



```
L0: t0  := y + z                | x := t1
    x  := t0                    | L1: ifnot (e > f) goto L2
    if (a > b) goto L1          |     if (g > h) goto L2
    ifnot (c > d) goto L1       |     ifnot (i > j) goto L0
    t1  := x + 1                | L2:
```

THANK YOU

Find that the harder I work, the more luck I seem to have.

——越努力，越幸运