四、中间代码生成
(3. 控制流语句的翻译)

魏恒峰

hfwei@nju.edu.cn

2023 年 05 月 10 日

Control.g4

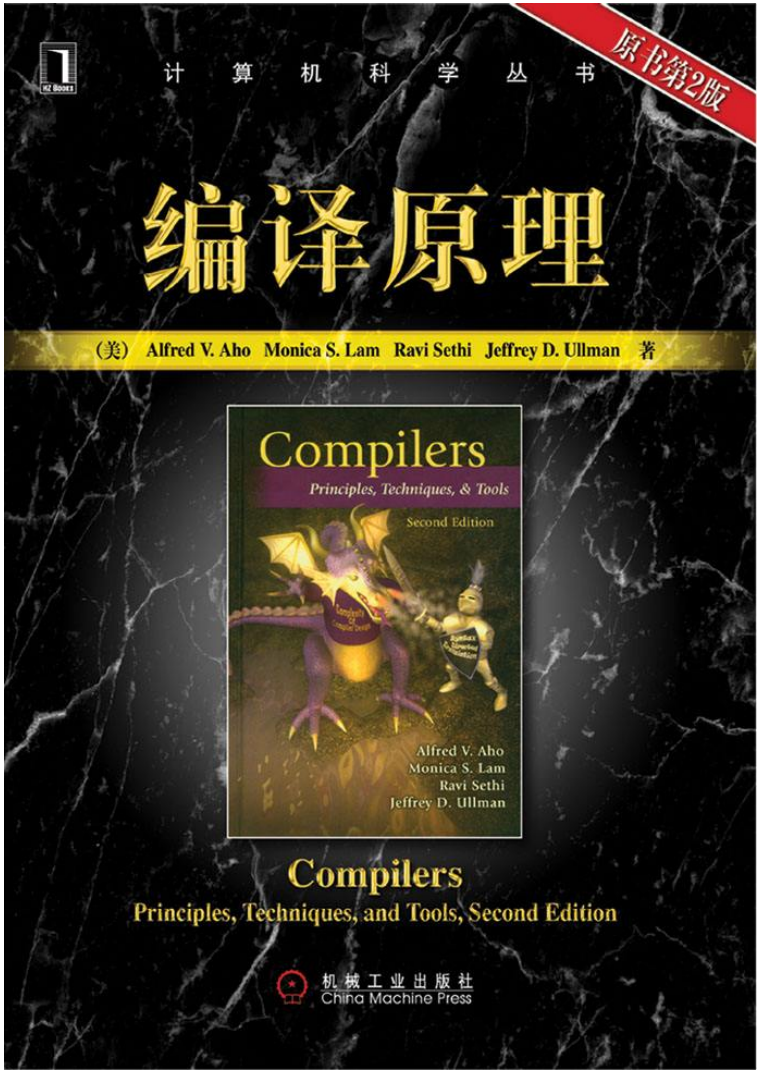
产生式
$P \rightarrow S$
$S \rightarrow$ **assign**
$S \rightarrow$ **if** (B) S_1
$S \rightarrow$ **if** (B) S_1 **else** S_2
$S \rightarrow$ **while** (B) S_1
$S \rightarrow S_1\ S_2$

Control.g4

产生式
$P \rightarrow S$
$S \rightarrow$ **assign**
$S \rightarrow$ **if** (B) S_1
$S \rightarrow$ **if** (B) S_1 **else** S_2
$S \rightarrow$ **while** (B) S_1
$S \rightarrow S_1\ S_2$

产生式
$B \rightarrow B_1$ || B_2
$B \rightarrow B_1$ && B_2
$B \rightarrow$! B_1
$B \rightarrow E_1$ **rel** E_2
$B \rightarrow$ **true**
$B \rightarrow$ **false**

本讲内容**非常重要**, 但**颇有难度**, 需要多多思考



虽然有**多种**讲法，教材偏偏采用了让初学者**望而生畏**的那一种。

“We will overcome all difficulties.” (我们有困难要上)

“If we don’t have enough difficulties, we’ll create them!”
(没困难, 我们创造困难还要上)

关键问题：为什么要“创造困难”？

关键问题：为什么要“创造困难”？

生成更短、更高效的代码


Easy Mode
Hard Mode

心中有“树”(语法分析树)

分工　　合作

为布尔表达式 B 计算逻辑值 (假设保存在临时变量 `t1` 中)

if、**while** 等语句根据 B 的结果改变控制流

$$\textbf{if}\,(B)\,S_1$$

CodeGenVisitor.java

选择实现方式: Listeners, Visitors, Attributed Grammar

产生式
$P \rightarrow S$
$S \rightarrow$ **assign**
$S \rightarrow$ **if** (B) S_1
$S \rightarrow$ **if** (B) S_1 **else** S_2
$S \rightarrow$ **while** (B) S_1
$S \rightarrow S_1\ S_2$

产生式
$B \rightarrow B_1$ **||** B_2
$B \rightarrow B_1$ **&&** B_2
$B \rightarrow$ **!** B_1
$B \rightarrow E_1$ **rel** E_2
$B \rightarrow$ **true**
$B \rightarrow$ **false**

及时输出生成的中间代码, 避免频繁的字符串拼接操作

How to implement "`break`" statements?

How to implement “Short-Circuit” evaluation?

直接**用布尔表达式改变控制流**

(使用控制流跳转间接表明布尔表达式的值)

分工　　合作

父节点为子节点准备跳转指令的目标标签

子节点通过**继承属性**确定跳转目标

$$P \to S \qquad S \to \textbf{if}\,(B)\,S_1$$

继承属性 $B.true$, $B.false$, $S.next$ 指明了控制流跳转目标

产生式	语义规则
$P \rightarrow S$	$S.next = newlabel()$ $P.code = S.code \mid\mid label(S.next)$
$S \rightarrow$ **assign**	$S.code =$ **assign**$.code$
$S \rightarrow$ **if** (B) S_1	$B.true = newlabel()$ $B.false = S_1.next = S.next$ $S.code = B.code \mid\mid label(B.true) \mid\mid S_1.code$
$S \rightarrow$ **if** (B) S_1 **else** S_2	$B.true = newlabel()$ $B.false = newlabel()$ $S_1.next = S_2.next = S.next$ $S.code = B.code$ $\mid\mid label(B.true) \mid\mid S_1.code$ $\mid\mid gen('\texttt{goto}'\ S.next)$ $\mid\mid label(B.false) \mid\mid S_2.code$
$S \rightarrow$ **while** (B) S_1	$begin = newlabel()$ $B.true = newlabel()$ $B.false = S.next$ $S_1.next = begin$ $S.code = label(begin) \mid\mid B.code$ $\mid\mid label(B.true) \mid\mid S_1.code$ $\mid\mid gen('\texttt{goto}'\ begin)$
$S \rightarrow S_1\ S_2$	$S_1.next = newlabel()$ $S_2.next = S.next$ $S.code = S_1.code \mid\mid label(S_1.next) \mid\mid S_2.code$

继承属性 $S.next$

$$P \rightarrow S \quad \left| \begin{array}{rcl} S.next & = & newlabel() \\ P.code & = & S.code \,||\, label(S.next) \end{array} \right.$$

$S.next$ 为语句 S 指明了“跳出” S 的目标

继承属性 $S.next$

$$P \rightarrow S \quad \left| \begin{array}{l} S.next = newlabel() \\ P.code = S.code \,||\, label(S.next) \end{array} \right.$$

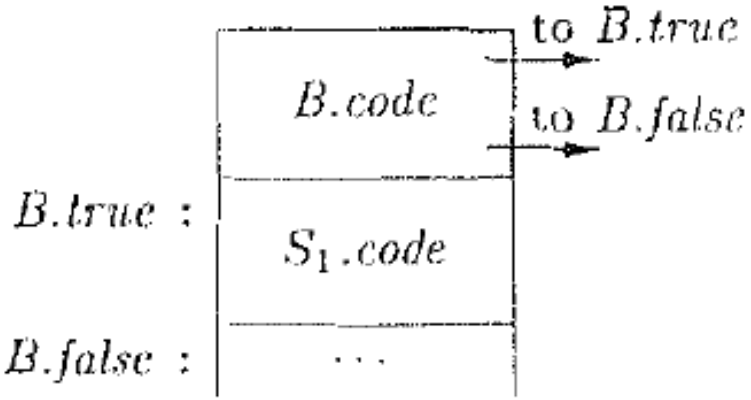
$S.next$ 为语句 S 指明了“跳出” S 的目标

$$S \rightarrow \textbf{assign} \quad \left| \; S.code = \textbf{assign}.code \right.$$

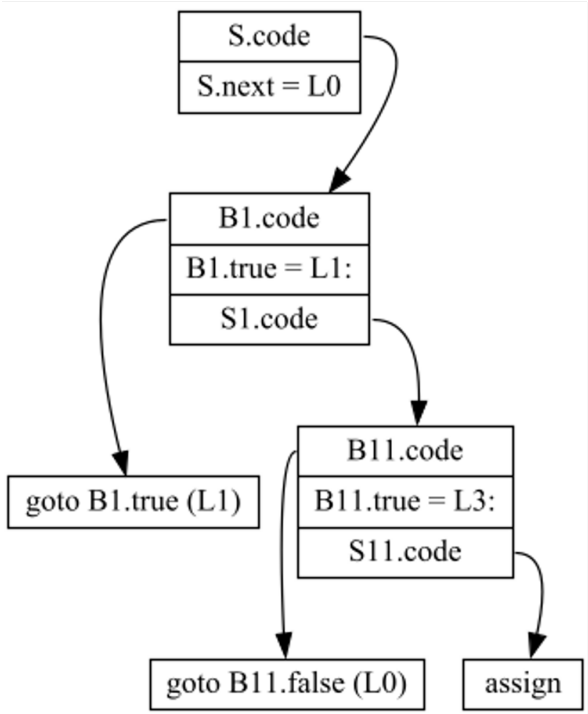
代表了表达式的翻译, 包括数组引用

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

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



$S \rightarrow \textbf{if}\ (\ B\ )\ S_1$	$B.true = newlabel()$ $B.false = S_1.next = S.next$ $S.code = B.code \mid\mid label(B.true) \mid\mid S_1.code$

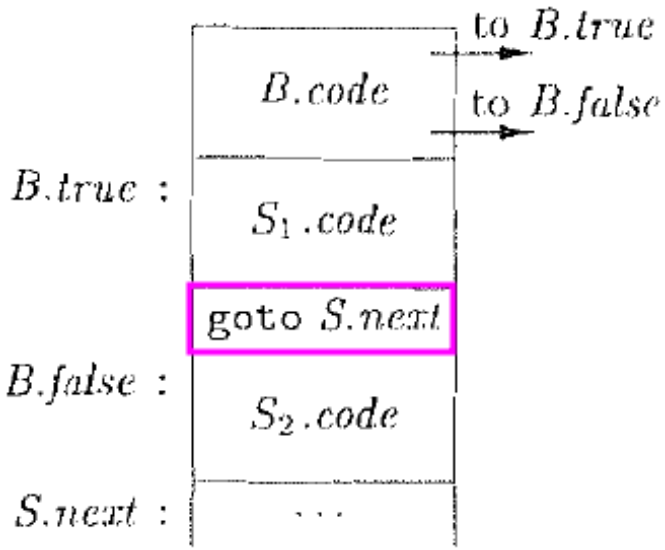



```
if (true)
  if (false) assign
```

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

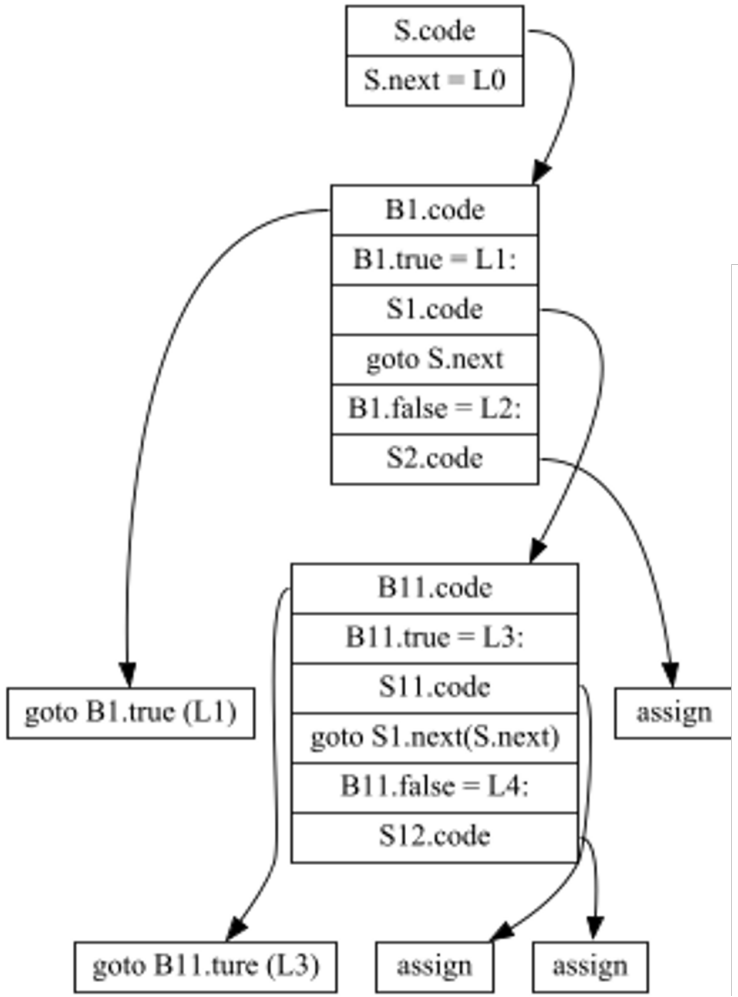
$$S \rightarrow \textbf{if}\ (\ B\ )\ S_1\ \textbf{else}\ S_2$$

$$
\begin{aligned}
&B.true = newlabel() \\
&B.false = newlabel() \\
&S_1.next = S_2.next = S.next \\
&S.code = B.code \\
&\quad || \ label(B.true) \ || \ S_1.code \\
&\quad || \ gen(\texttt{'goto'}\ S.next) \\
&\quad || \ label(B.false) \ || \ S_2.code
\end{aligned}
$$



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

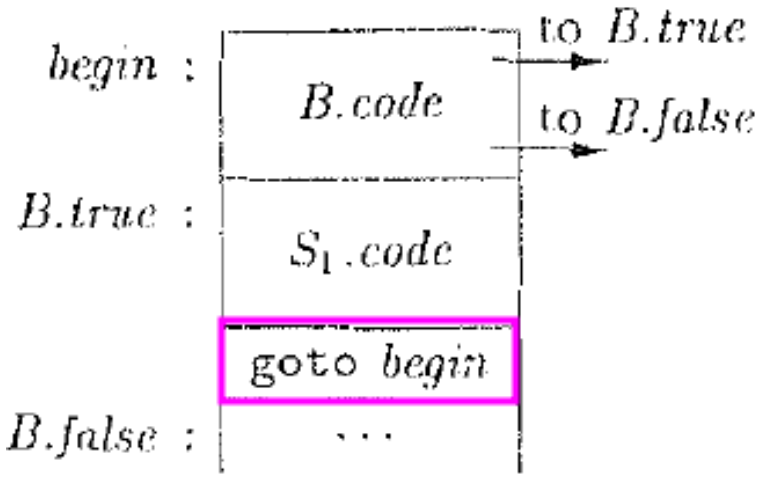
$B.true = newlabel()$
$B.false = newlabel()$
$S_1.next = S_2.next = S.next$
$S.code = B.code$
$\quad || label(B.true) || S_1.code$
$\quad || gen('\texttt{goto}'\ S.next)$
$\quad || label(B.false) || S_2.code$




```
if (true)
  if (true) assign else assign
else
  assign
```

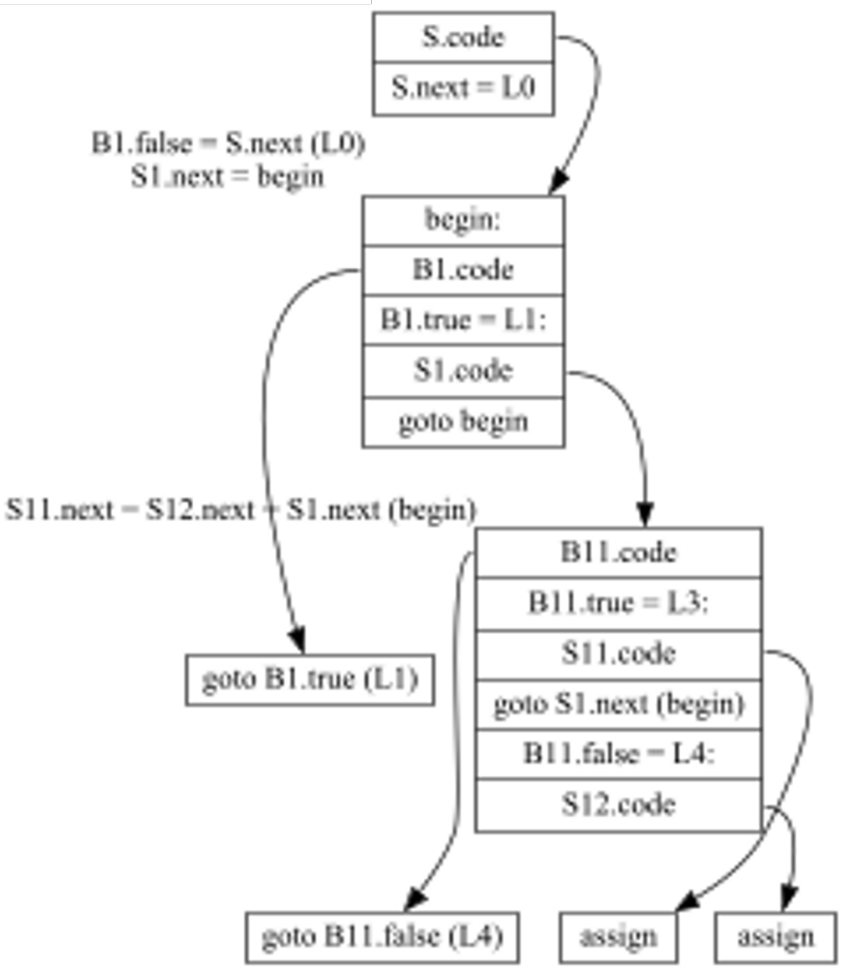
$S \rightarrow$ **while** (B) S_1

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



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

$$
\begin{aligned}
&begin = newlabel() \\
&B.true = newlabel() \\
&B.false = S.next \\
&S_1.next = begin \\
&S.code = label(begin) \mid\mid B.code \\
&\qquad \mid\mid label(B.true) \mid\mid S_1.code \\
&\qquad \mid\mid gen(\text{'goto'}\ begin)
\end{aligned}
$$




```
while (true)
  if (false) assign else assign
```

$$S \rightarrow S_1\ S_2$$

$$\begin{array}{|l} S_1.next = newlabel() \\ S_2.next = S.next \\ S.code = S_1.code \mid\mid label(S_1.next) \mid\mid S_2.code \end{array}$$

$$S \rightarrow S_1\ S_2$$

$$\begin{array}{|l} S_1.next = newlabel() \\ S_2.next = S.next \\ S.code = S_1.code \mid\mid label(S_1.next) \mid\mid S_2.code \end{array}$$

```
if (true) assign else assign assign
```

产生式	语义规则
$B \rightarrow B_1 \mid\mid B_2$	$B_1.true = B.true$ $B_1.false = newlabel()$ $B_2.true = B.true$ $B_2.false = B.false$ $B.code = B_1.code \mid\mid label(B_1.false) \mid\mid B_2.code$
$B \rightarrow B_1 \;\&\&\; B_2$	$B_1.true = newlabel()$ $B_1.false = B.false$ $B_2.true = B.true$ $B_2.false = B.false$ $B.code = B_1.code \mid\mid label(B_1.true) \mid\mid B_2.code$
$B \rightarrow ! B_1$	$B_1.true = B.false$ $B_1.false = B.true$ $B.code = B_1.code$
$B \rightarrow E_1$ **rel** E_2	$B.code = E_1.code \mid\mid E_2.code$ $\mid\mid gen('\texttt{if}'\ E_1.addr\ \textbf{rel}.op\ E_2.addr\ '\texttt{goto}'\ B.true)$ $\mid\mid gen('\texttt{goto}'\ B.false)$
$B \rightarrow$ **true**	$B.code = gen('\texttt{goto}'\ B.true)$
$B \rightarrow$ **false**	$B.code = gen('\texttt{goto}'\ B.false)$

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

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

if (true) assign

$S \rightarrow$ **if** (B) S_1	$B.true$	$= newlabel()$
	$B.false$	$= S_1.next = S.next$
	$S.code$	$= B.code \mid\mid label(B.true) \mid\mid S_1.code$

if (false) assign

$$B \rightarrow\ !\ B_1$$

$$\begin{cases} B_1.true = B.false \\ B_1.false = B.true \\ B.code = B_1.code \end{cases}$$

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

if (!true) assign

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

if (!false) assign

短路求值

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

短路求值

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

```
if (true || false) assign
```

$S \rightarrow \textbf{if}\ (\ B\ )\ S_1$	$B.true = newlabel()$ $B.false = S_1.next = S.next$ $S.code = B.code \mid\mid label(B.true) \mid\mid S_1.code$

```
if (false || true) assign
```

短路求值

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

短路求值

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

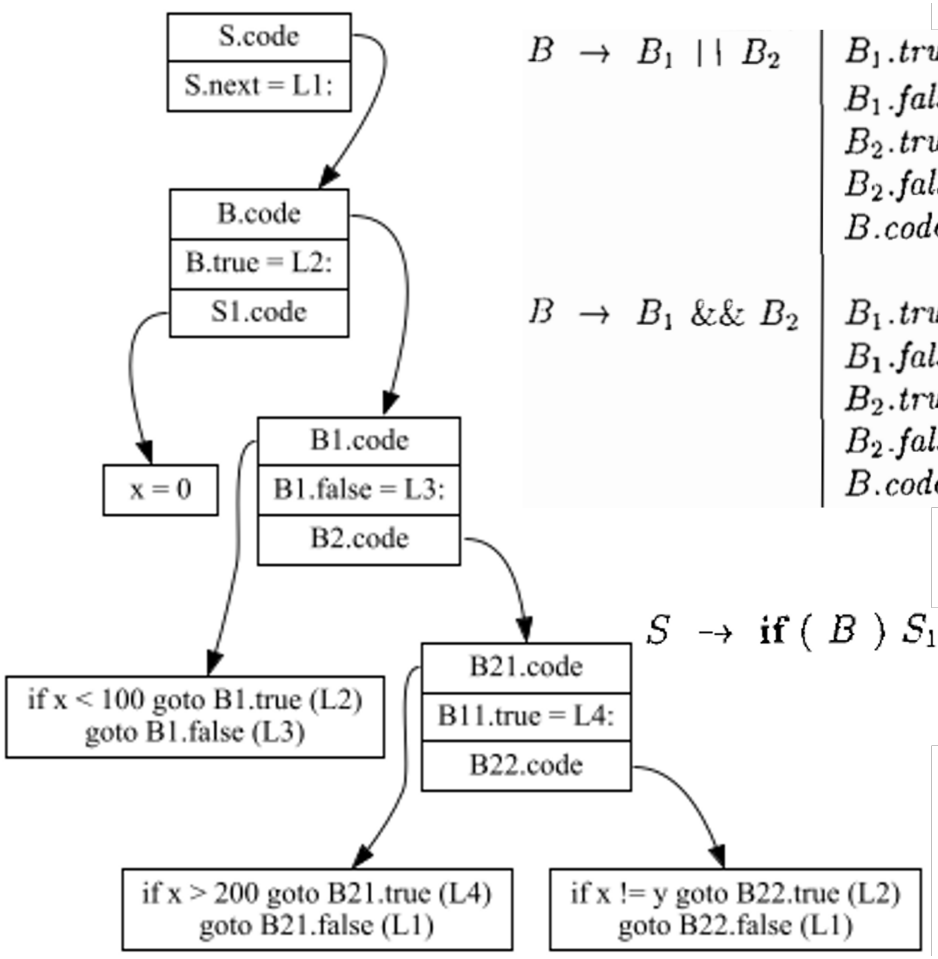
```
if (true && false) assign
```

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

```
if (false && true) assign
```

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

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



$B \rightarrow B_1 \;||\; B_2$

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

$B \rightarrow B_1 \;\&\&\; B_2$

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

$S \rightarrow \textbf{if} \; (\; B \;) \; S_1$

$B.true = newlabel()$
$B.false = S_1.next = S.next$
$S.code = B.code \;||\; label(B.true) \;||\; S_1.code$

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

```
        if x < 100 goto L2
        goto L3
L3:     if x > 200 goto L4
        goto L1
L4:     if x != y goto L2
        goto L1
L2:     x = 0
L1:
```

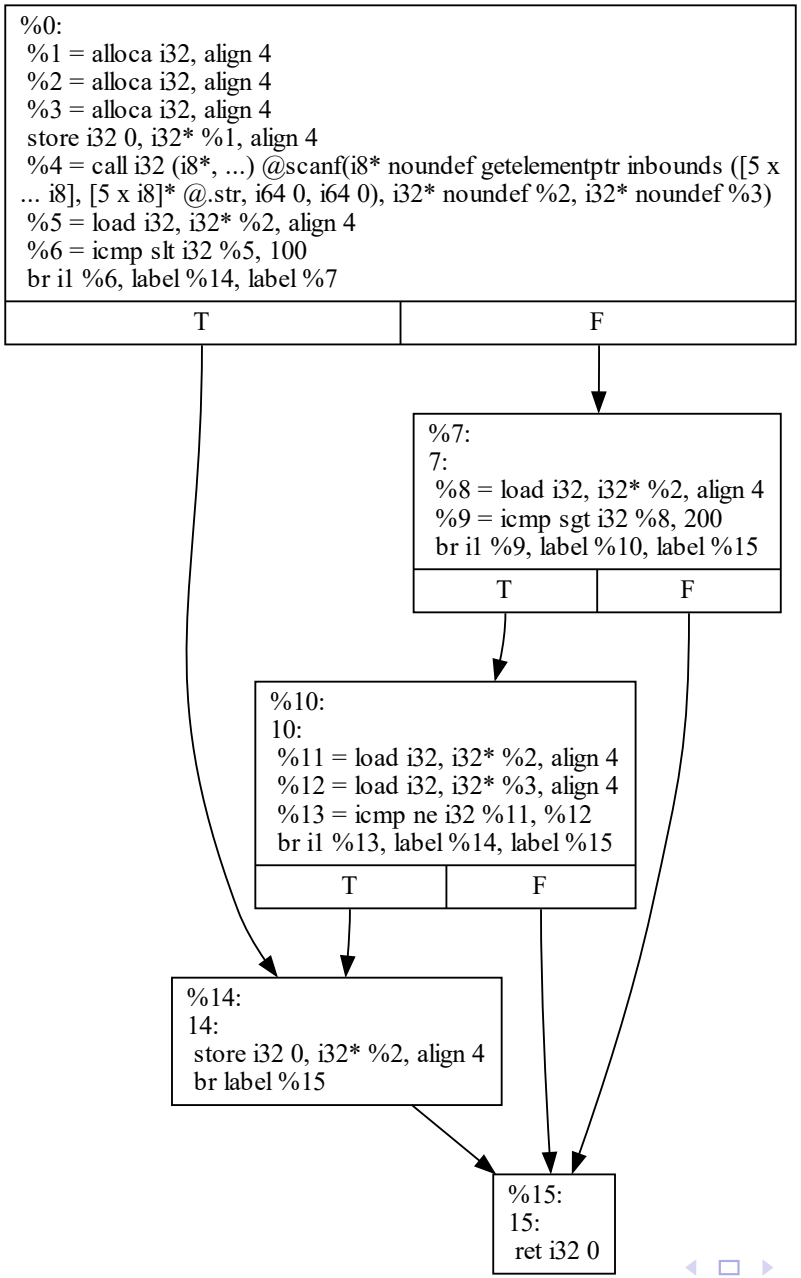
```
clang -S -emit-llvm if-boolexpr.c -o if-boolexpr-opt0.ll
```

```c
int main() {
  int x, y;
  scanf( Format: "%d%d", &x, &y);

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

  return 0;
}
```

```
opt -dot-cfg if-boolexpr-opt0.ll
dot -Tpdf .main.dot -o if-boolexpr-opt0-cfg.pdf
```



CFG for 'main' function

产生式	语义规则
$S \rightarrow \textbf{id} = E\ ;$	$S.code = E.code \mid\mid gen(top.get(\textbf{id}.lexeme)\ '='\ E.addr)$
$E \rightarrow E_1 + E_2$	$E.addr = \textbf{new}\ Temp()$ $E.code = E_1.code \mid\mid E_2.code \mid\mid gen(E.addr\ '='\ E_1.addr\ '+'\ E_2.addr)$
$\mid\ - E_1$	$E.addr = \textbf{new}\ Temp()$ $E.code = E_1.code \mid\mid gen(E.addr\ '='\ \textbf{'minus'}\ E_1.addr)$
$\mid\ (\ E_1\ )$	$E.addr = E_1.addr$ $E.code = E_1.code$
$\mid\ \textbf{id}$	$E.addr = top.get(\textbf{id}.lexeme)$ $E.code = ''$

产生式	语义规则
$P \rightarrow S$	$S.next = newlabel()$ $P.code = S.code \mid\mid label(S.next)$
$S \rightarrow \textbf{assign}$	$S.code = \textbf{assign}.code$
$S \rightarrow \textbf{if}\ (\ B\ )\ S_1$	$B.true = newlabel()$ $B.false = S_1.next = S.next$ $S.code = B.code \mid\mid label(B.true) \mid\mid S_1.code$
$S \rightarrow \textbf{if}\ (\ B\ )\ S_1\ \textbf{else}\ S_2$	$B.true = newlabel()$ $B.false = newlabel()$ $S_1.next = S_2.next = S.next$ $S.code = B.code$ $\mid\mid label(B.true) \mid\mid S_1.code$ $\mid\mid gen('\textbf{goto}'\ S.next)$ $\mid\mid label(B.false) \mid\mid S_2.code$
$S \rightarrow \textbf{while}\ (\ B\ )\ S_1$	$begin = newlabel()$ $B.true = newlabel()$ $B.false = S.next$ $S_1.next = begin$ $S.code = label(begin) \mid\mid B.code$ $\mid\mid label(B.true) \mid\mid S_1.code$ $\mid\mid gen('\textbf{goto}'\ begin)$
$S \rightarrow S_1\ S_2$	$S_1.next = newlabel()$ $S_2.next = S.next$ $S.code = S_1.code \mid\mid label(S_1.next) \mid\mid S_2.code$

$S \rightarrow \textbf{id} = E\ ;$	$\{\ gen(top.get(\textbf{id}.lexeme)\ '='\ E.addr);\ \}$
$\mid\ L = E\ ;$	$\{\ gen(L.array.base\ '['\ L.addr\ ']'\ '='\ E.addr);\ \}$
$E \rightarrow E_1 + E_2$	$\{\ E.addr = \textbf{new}\ Temp();$ $gen(E.addr\ '='\ E_1.addr\ '+'\ E_2.addr);\ \}$
$\mid\ \textbf{id}$	$\{\ E.addr = top.get(\textbf{id}.lexeme);\ \}$
$\mid\ L$	$\{\ E.addr = \textbf{new}\ Temp();$ $gen(E.addr\ '='\ L.array.base\ '['\ L.addr\ ']');\ \}$
$L \rightarrow \textbf{id}\ [\ E\ ]$	$\{\ L.array = top.get(\textbf{id}.lexeme);$ $L.type = L.array.type.elem;$ $L.addr = \textbf{new}\ Temp();$ $gen(L.addr\ '='\ E.addr\ '*'\ L.type.width);\ \}$
$\mid\ L_1\ [\ E\ ]$	$\{\ L.array = L_1.array;$ $L.type = L_1.type.elem;$ $t = \textbf{new}\ Temp();$ $L.addr = \textbf{new}\ Temp();$ $gen(t\ '='\ E.addr\ '*'\ L.type.width);$ $gen(L.addr\ '='\ L_1.addr\ '+'\ t);\ \}$

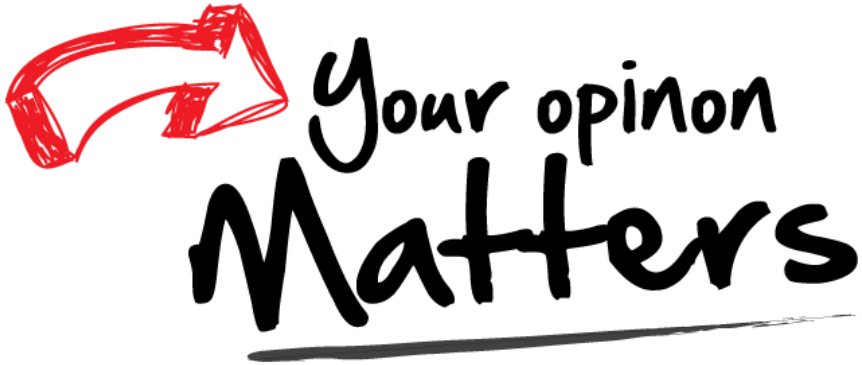
产生式	语义规则
$B \rightarrow B_1\ \mid\mid\ B_2$	$B_1.true = B.true$ $B_1.false = newlabel()$ $B_2.true = B.true$ $B_2.false = B.false$ $B.code = B_1.code \mid\mid label(B_1.false) \mid\mid B_2.code$
$B \rightarrow B_1\ \&\&\ B_2$	$B_1.true = newlabel()$ $B_1.false = B.false$ $B_2.true = B.true$ $B_2.false = B.false$ $B.code = B_1.code \mid\mid label(B_1.true) \mid\mid B_2.code$
$B \rightarrow !\ B_1$	$B_1.true = B.false$ $B_1.false = B.true$ $B.code = B_1.code$
$B \rightarrow E_1\ \textbf{rel}\ E_2$	$B.code = E_1.code \mid\mid E_2.code$ $\mid\mid gen('\textbf{if}'\ E_1.addr\ \textbf{rel}.op\ E_2.addr\ '\textbf{goto}'\ B.true)$ $\mid\mid gen('\textbf{goto}'\ B.false)$
$B \rightarrow \textbf{true}$	$B.code = gen('\textbf{goto}'\ B.true)$
$B \rightarrow \textbf{false}$	$B.code = gen('\textbf{goto}'\ B.false)$

Control.g4 CodeGenListener.java

CODEGEN

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


Thank
You!



Office 926

hfwei@nju.edu.cn