四、中间代码生成
(12. 控制流语句的翻译 (Hard 模式))

魏恒峰

hfwei@nju.edu.cn

2024 年 05 月 15 日

各个非终结符的**分工**、**合作**明确易懂

只需要使用**综合属性**

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

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

如何生成更短、更高效的代码?

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

```
goto L2
L3:
goto L1
L2:
ASSIGN
L1:
```

```
br true or.true1 or.false2
or.false2:
t1 = OR true false
br or.end3:
or.true1:
t1 = true
or.end3:
br t1 b.true4 b.false5
b.true4:
a = b
b.false5:
```

如何生成更短、更高效的代码?

```
while (true) {
  if (false) {
    a = b;
  }
}
```

```
L2begin:
goto L3
L3:
goto L2begin
L4:
ASSIGN
goto L2begin
L1:
```

```
begin1:
br true b.true2 b.false3
b.true2:
br false b.true4 b.false5
b.true4:
a = b
b.false5:
br begin1
b.false3:
```

直接**用布尔表达式改变控制流**，无需计算最终逻辑值

从父节点获取**更具体的跳转目标**，缩短跳转路径

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

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

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**

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

继承属性 $S.next$

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

|| 要理解成append的意思

$S.next$ 为语句 S 指明了“跳出” S 的目标

赋值语句

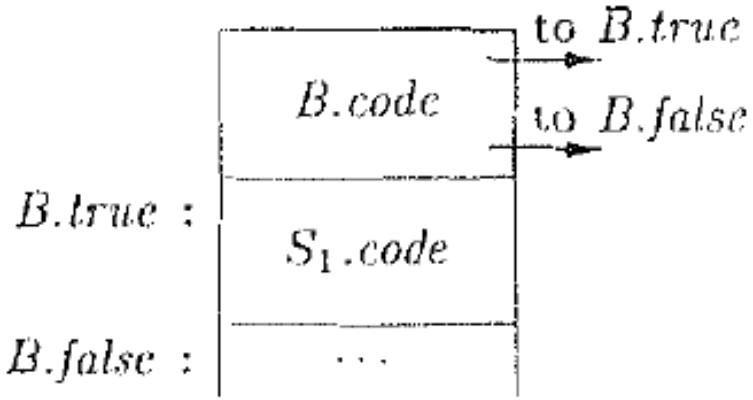
$$S \rightarrow \textbf{assign} \quad | \quad S.code = \textbf{assign}.code$$

代表了表达式的翻译, 包括数组引用

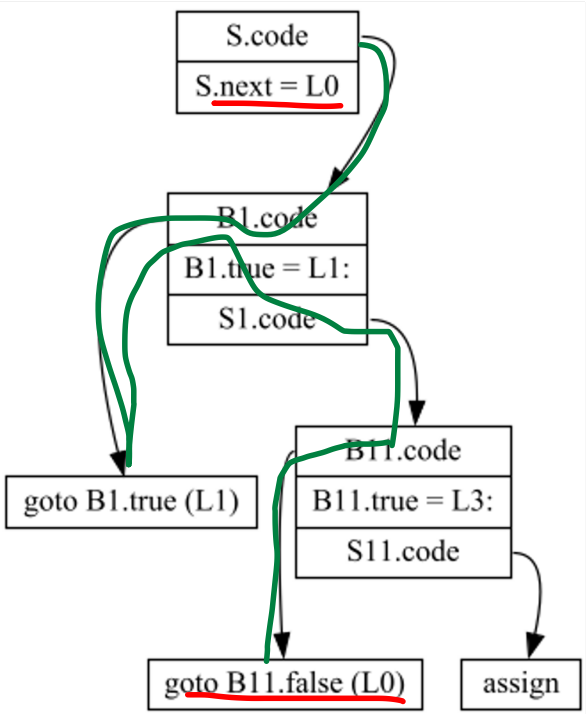
Btrue,进入括号内，后续还有内容，因此需要在括号后/内容前打上label

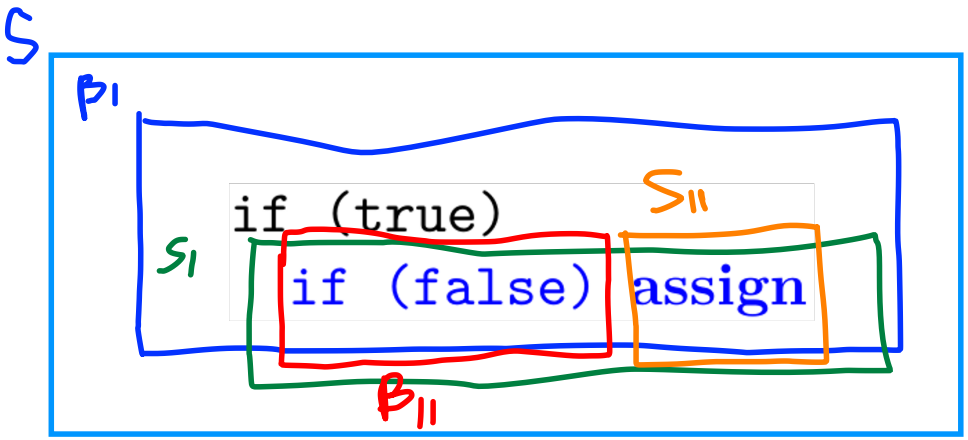
$$S \rightarrow \textbf{if} \; (\; B \;) \; S_1$$

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



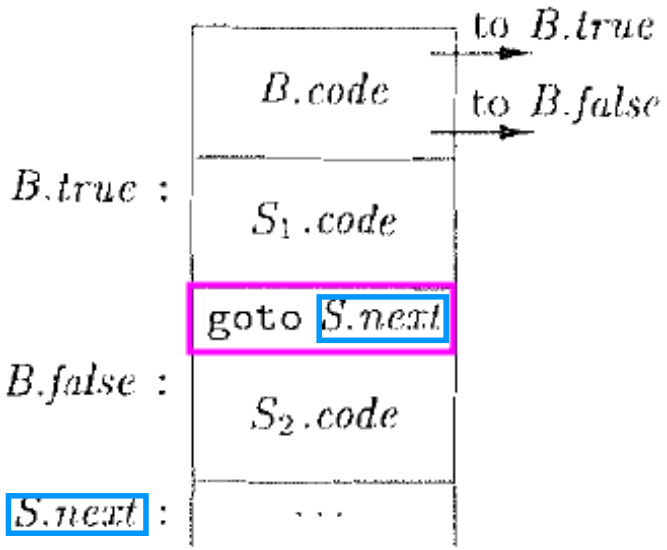
$S \rightarrow \textbf{if} \; (\; B \;) \; S_1$	$B.true = newlabel()$ $B.false = S_1.next = S.next$ $S.code = B.code \; \| \; label(B.true) \; \| \; S_1.code$





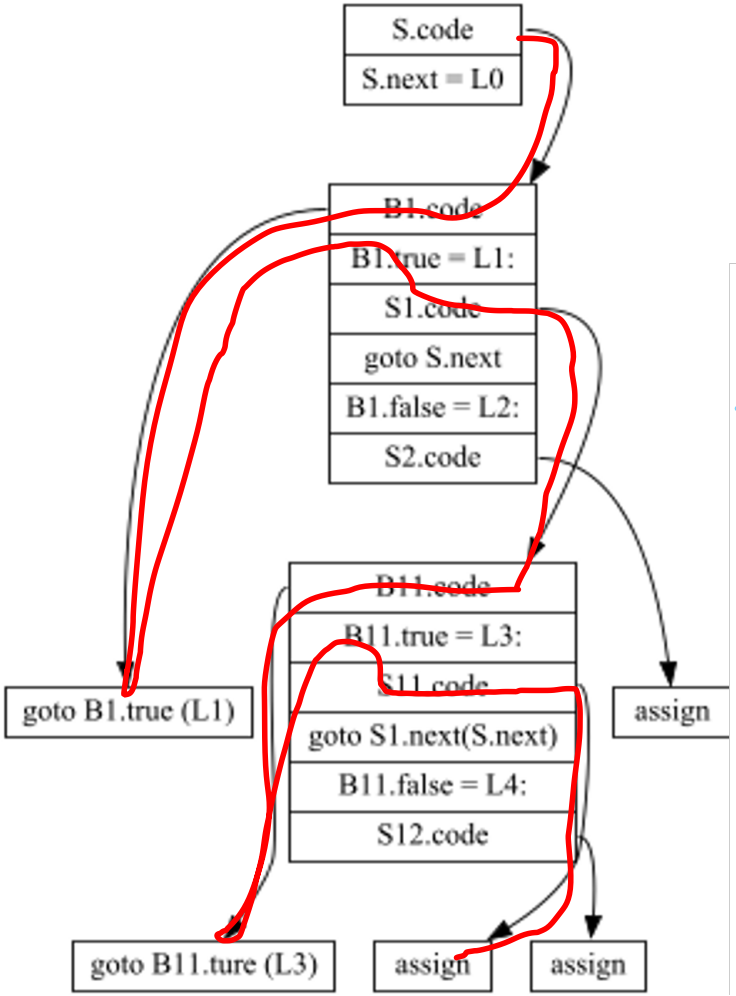
$B \rightarrow \textbf{true}$	$B.code = gen('\texttt{goto}' \; B.true)$
$B \rightarrow \textbf{false}$	$B.code = gen('\texttt{goto}' \; B.false)$

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



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

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



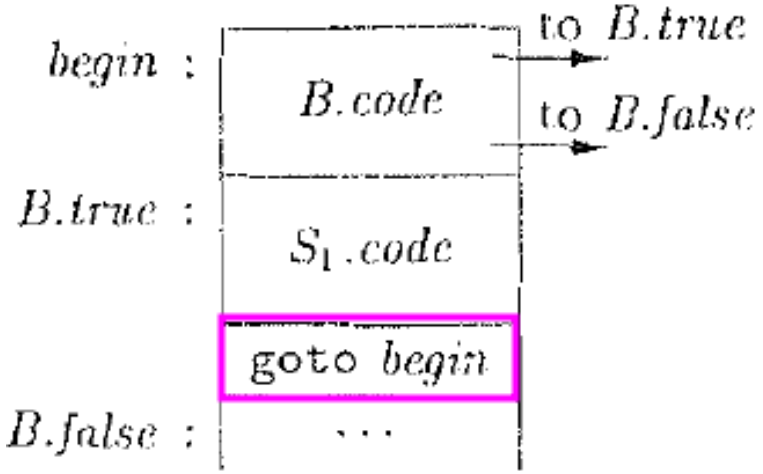
S

B_1

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

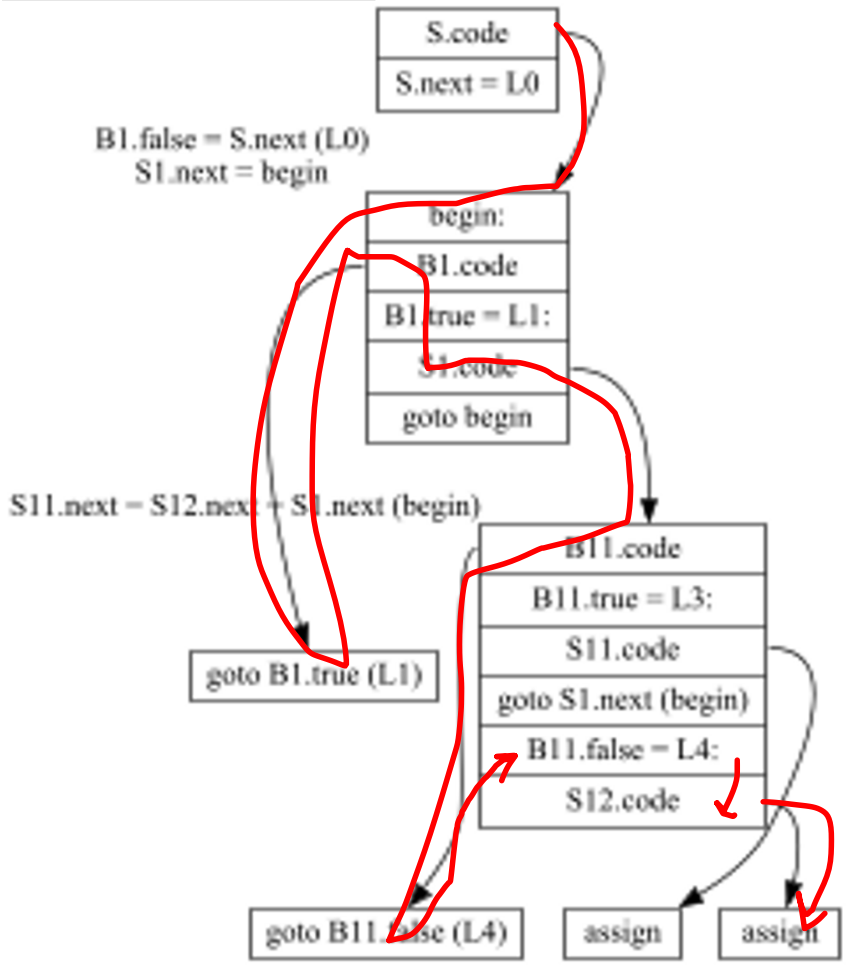
S_1 S_{11} S_{12} B_{11}

$$
\begin{array}{lll}
S \rightarrow \textbf{while}\ (B)\ S_1 & \left|\begin{array}{l}
begin = newlabel() \\
B.true = newlabel() \\
B.false = S.next \\
\boxed{S_1.next} = begin \\
S.code = label(begin) \mid\mid B.code \\
\qquad \mid\mid label(B.true) \mid\mid S_1.code \\
\qquad \mid\mid \boxed{gen(\text{'goto'}\ begin)}
\end{array}\right.
\end{array}
$$



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

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




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

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

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

bool代数类

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

$$B \rightarrow \text{true} \quad \Big| \quad B.code = gen(\text{'goto'}\ \boxed{B.true})$$

$$B \rightarrow \text{false} \quad \Big| \quad B.code = gen(\text{'goto'}\ \boxed{B.false})$$

`if (true)` **assign**

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

`if (false)` **assign**

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

if (!true) assign

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

if (!false) assign

短路求值

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

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

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

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

短路求值

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

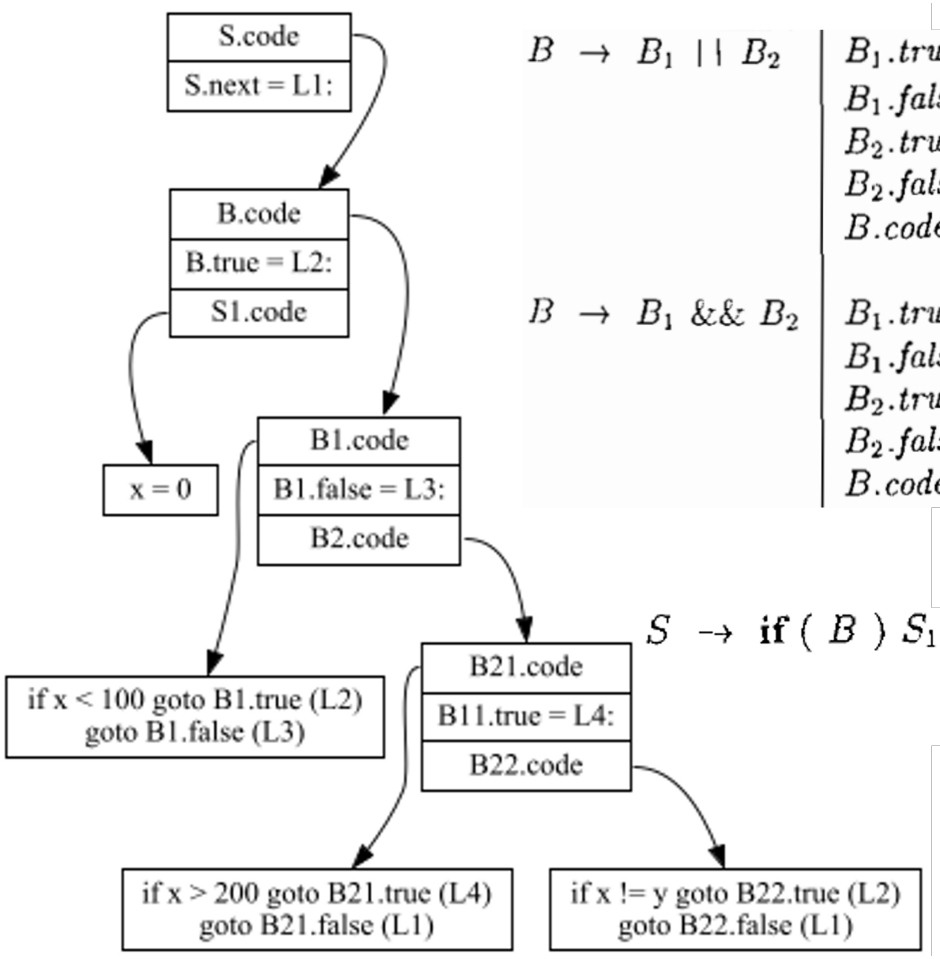
```
if (true && false) assign
```

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

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

$$
\begin{array}{lll}
B \rightarrow E_1 \ \mathbf{rel} \ E_2 & \Big| & B.code = E_1.code \ || \ E_2.code \\
 & & \quad || \ \boxed{gen}(\texttt{'if'} \ E_1.addr \ \mathbf{rel}.op \ E_2.addr \ \texttt{'goto'} \ B.true) \\
 & & \quad || \ \boxed{gen}(\texttt{'goto'} \ B.false)
\end{array}
$$

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



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

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

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

```
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:
```

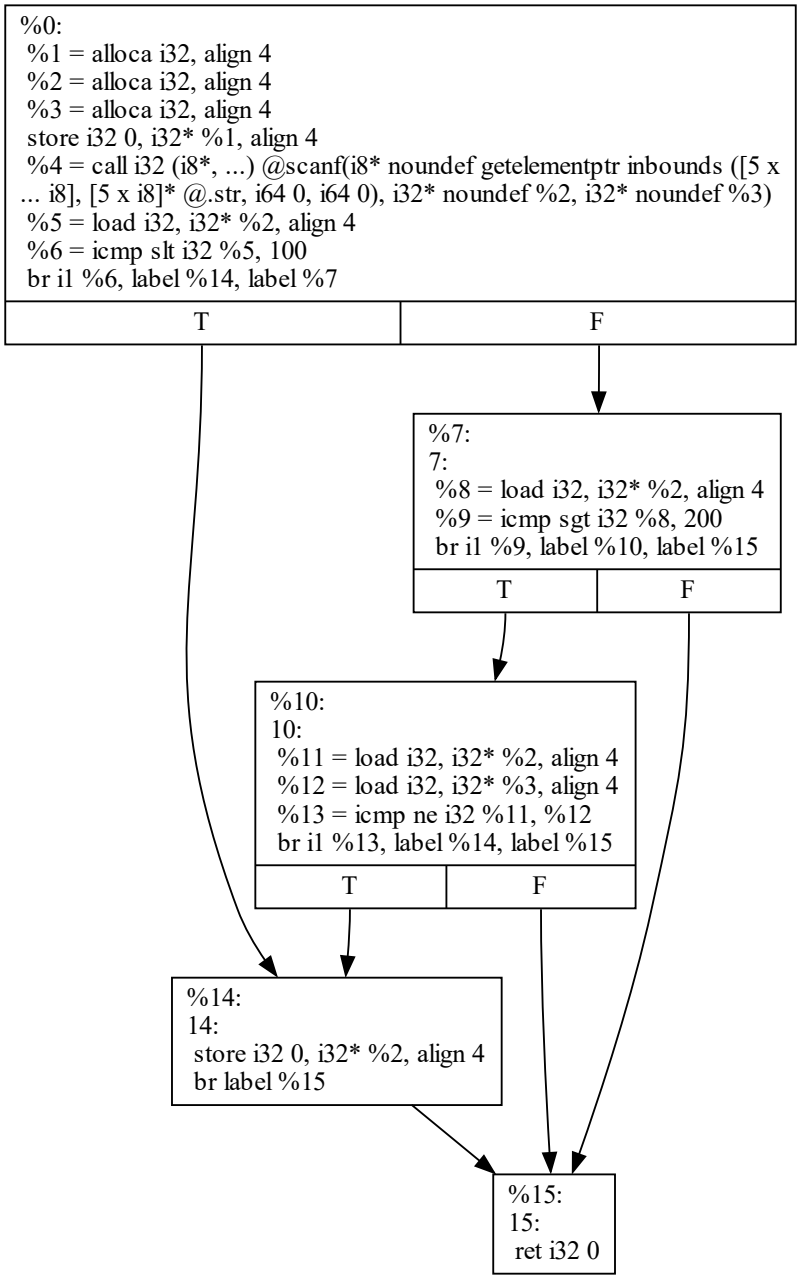
```
clang -S -emit-llvm if-boolexpr.c -o if-boolexpr-opt0.ll
```

```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

CodeGenListener.java

思考：如何实现增量式翻译？

```
while (true) {
  if (false) {
    a = b;
  }
}
```

```
L2begin:
goto L3
L3:
goto L2begin
L4:
ASSIGN
goto L2begin
L1:
```

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

```
goto L2
L3:
goto L1
L2:
ASSIGN
L1:
```

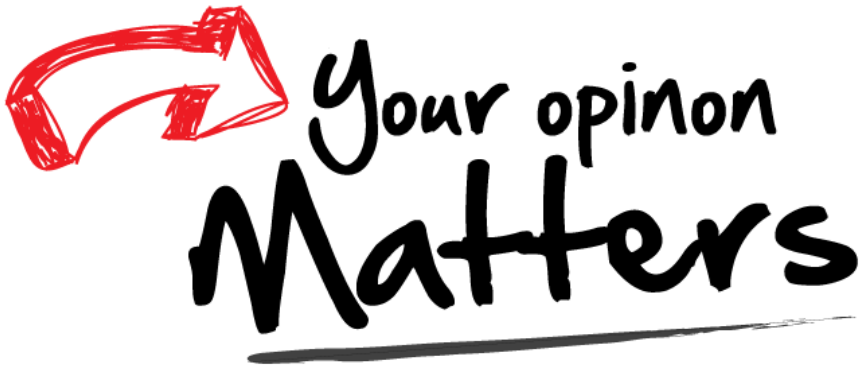
产生式	语义规则
$S \rightarrow \textbf{id} = E\ ;$	$S.code = E.code \mid\mid gen(top.get(\textbf{id}.lexeme)\ '='\ E.addr)$
$E \rightarrow E_1 + E_2$	$E.addr = \textbf{new}\ Temp()$ $E.code = E_1.code \mid\mid E_2.code \mid\mid gen(E.addr\ '='\ E_1.addr\ '+'\ E_2.addr)$
$\mid\ -E_1$	$E.addr = \textbf{new}\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

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

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

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

Thank
You!



Office 926

hfwei@nju.edu.cn