– INF01147 –
Compiladores

Otimização de Código
Local e Global

Prof. Lucas M. Schnorr
– Universidade Federal do Rio Grande do Sul –

Plano da Aula de Hoje

- Exercício de revisão da aula anterior

- Otimização Local e Global

Exercício de Revisão – Calcule o grafo de blocos básicos

```
(1)   i = 1
(2)   j = 1
(3)   t1 = 10 * i
(4)   t2 = t1 + j
(5)   t3 = 8 * t2
(6)   if i < 10 goto (16)
(7)   t4 = t3 - 88
(8)   a[t4] = 0.0
(9)   j = j + 1
(10)  if j <= 10 goto (3)
(11)  i = i + 1
(12)  if i <= 10 goto (20)
(13)  i = 1
(14)  t5 = i - 1
(15)  t6 = 88 * t5
(16)  a[t6] = 1.0
(17)  i = i + 1
(18)  if i <= 10 goto (13)
(19)  goto (5)
(20)  if j < 2 goto (5)
```

Otimização de Código

Local e Global

Otimização de Código

- Considerar um grafo de blocos básicos é fundamental
- LLVM/Clang
  - http://llvm.org/docs/Passes.html

- Local – **Eliminação de sub-expressões comuns**
  - Todas aquelas vistas em Otimização de Janela
    $\rightarrow$ mas limitadas as instruções de um blocos básico por vez
- Global – **Otimização de Laços**
  - Movimentação de código
  - Variáveis de indução
  - Redução de força

Otimização de Código

Eliminação de sub-expressões comuns

Otimização Local

- Eliminação de sub-expressões comuns
- Duas operações são comuns se produzem o mesmo resultado
  - Calcular uma única vez
  - Reutilizar nas próximas vezes, referenciando o resultado

- Expressão viva
  - Se os operandos usados para calculá-la não mudaram

Otimização Local – Primeiro exemplo

- TAC original

```
1     t1 = 1 + 20
2     t2 = -x
3     x = t1 * t2
4     t3 = x * x
5     t4 = x / y
6     t5 = x / y
7     y = t3 + t4
8     t6 = x * x
9     z = t5 / t6
10    y = z
```

- TAC melhorado

```
1     t2 = -x
2     x = 21 * t2
3     t3 = x * x
4     t4 = x / y
5     y = t3 + t4
6     z = t4 / t3
7     y = z
```

Otimização Local – Segundo Exemplo

- TAC original

```
1    t6 = 4 * i
2    x = a[t6]
3    t7 = 4 * i
4    t8 = 4 * j
5    t9 = a[t8]
6    a[t7] = t9
7    t10 = 4 * j
8    a[t10] = x
9    goto (5)
```

- TAC melhorado

```
1    t6 = 4 * i
2    x = a[t6]
3    t8 = 4 * j
4    t9 = a[t8]
5    a[t6] = t9
6    a[t8] = x
7    goto (5)
```

Eliminação de sub-expressões comuns

- Como implementar?
  - Representar expressões com um DAG

- Folhas são identificadores e constantes
  - Valores inicias da computação efetuado no bloco
- Nós intermediários são operadores/lista de variáveis

Exemplo

TAC original

```
t1 = 4 * i
t2 = a[t1]
t3 = 4 * i
t4 = b[t3]
t5 = t2 * t4
t6 = prod + t5
prod = t6
t7 = i + 1
i = t7
if i <= 20 goto (3)
```

TAC melhorado

```
t1 = 4 * i
t2 = a[t1]
t4 = b[t1]
t5 = t2*t4
prod = prod + t5
i = i + 1
if i <= 20 goto (3)
```

Exercício

- Crie o DAG de expressões e otimize o código

TAC original

```
14   t6 = 4 * i
15   x = a[t6]
16   t7 = 4 * i
17   t8 = 4 * j
18   t9 = a[t8]
19   a[t7] = t9
20   t10 = 4 * j
21   a[t10] = x
22   goto (5)
```

TAC melhorado

```
14   t6 = 4 * i
15   x = a[t6]
16   t8 = 4 * j
17   t9 = a[t8]
18   a[t6] = t9
19   a[t8] = x
20   goto (5)
```

Otimização de Código

Otimização de Laços

Otimização de Laços

- Laços constituem um local importante para otimização
  - Laços internos ainda mais importantes

- Objetivo
  - Reduzir a quantidade de instruções dentro do laço
  - Reduzir o custo de instruções dentro do laço
- Estratégias
  - Movimentação de código
  - Remover variáveis de indução
  - Reduzir poder de instruções

Movimentação de Código

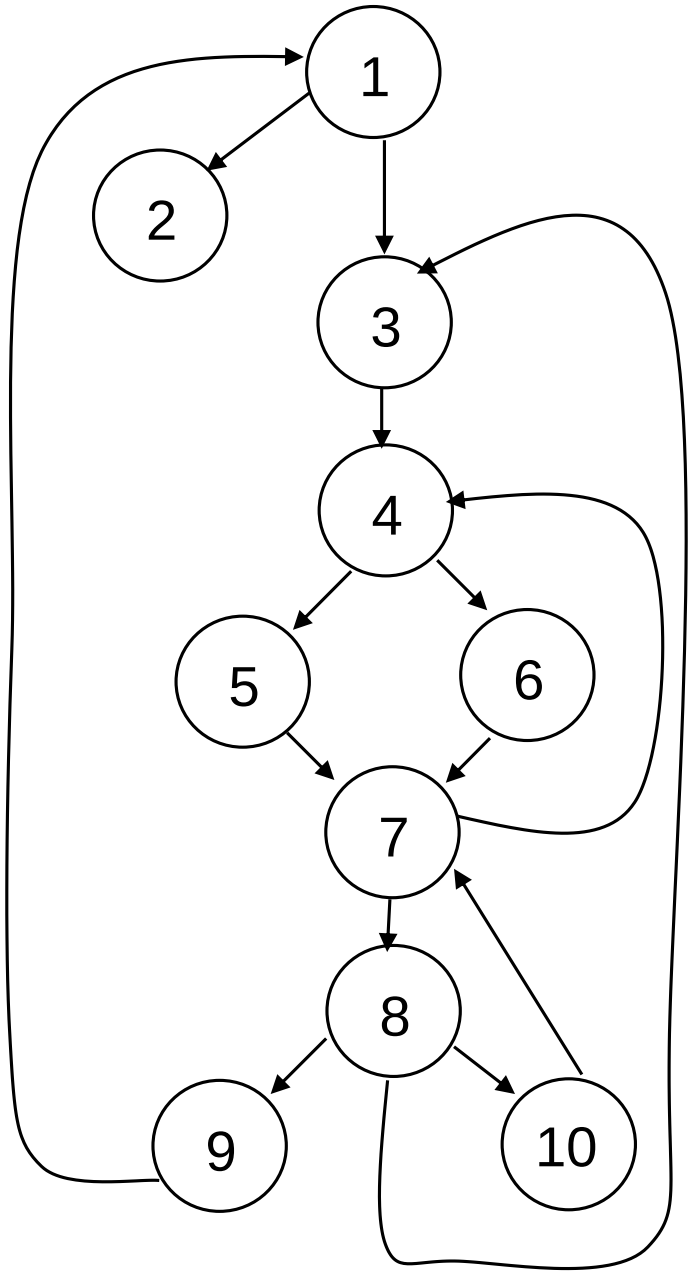
- Movimentar código para fora do laço
- Exemplo

```
while (i <= limit-2)
```

- Exemplo

```
for (i = 0; i < n; i++)
   for (j = 0; j < n; j++)
      a[n*i + j] = b[j];
```

Definição de laço?

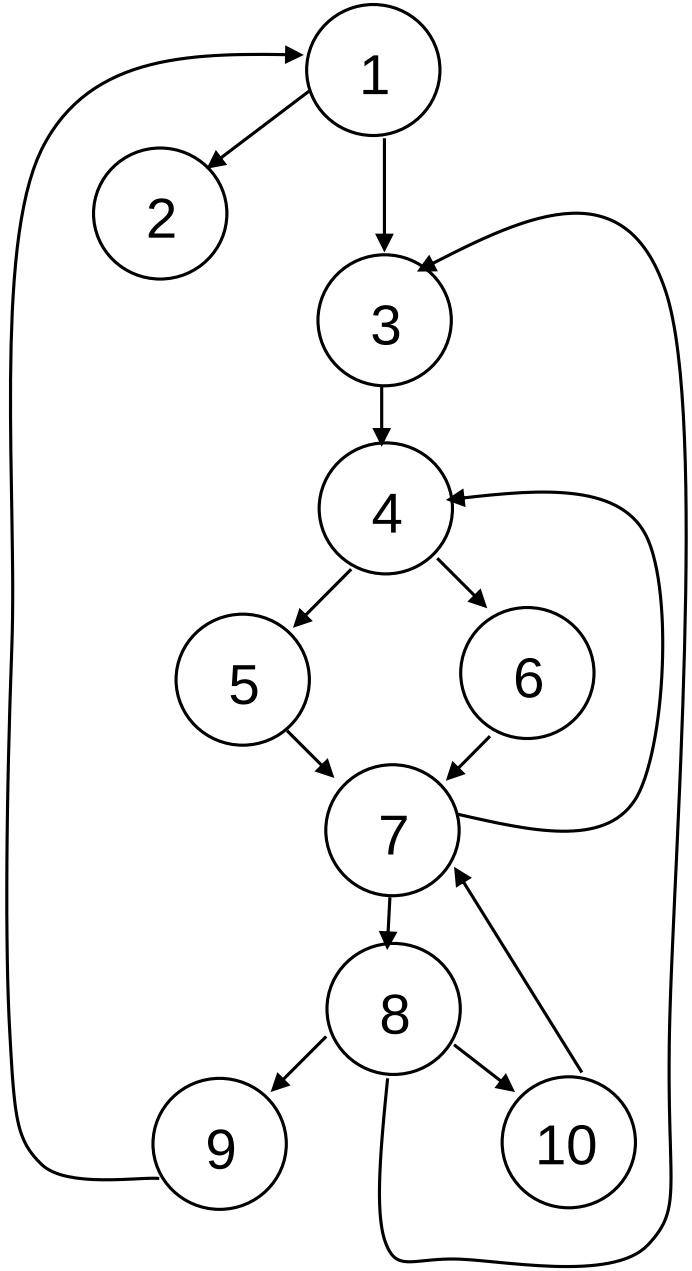


Dominadores

- Usados para encontrar laços no grafo de fluxo de controle

- Um nó n domina m se
  - Todos os caminhos na direção de m passam por n
    - A partir de um nó dito inicial do grafo
  - $m = n$ (um nó se domina)
- Árvore de dominadores

Calculando a Árvore de Dominadores

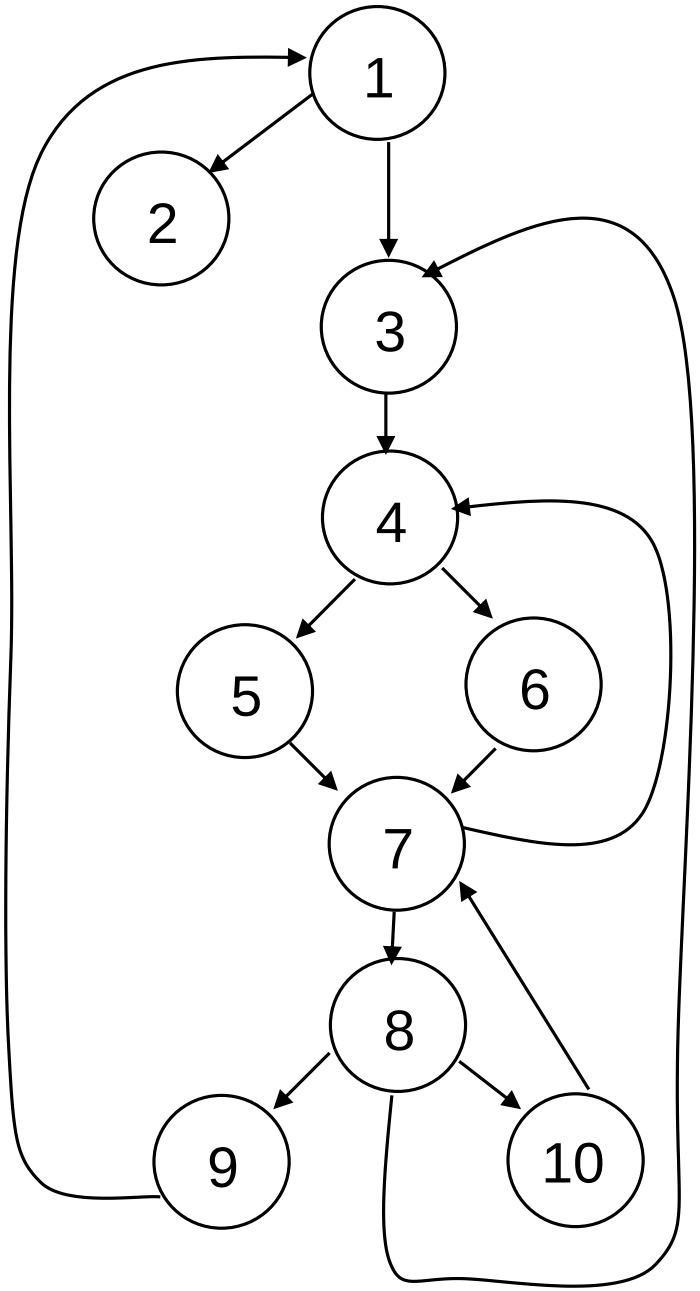


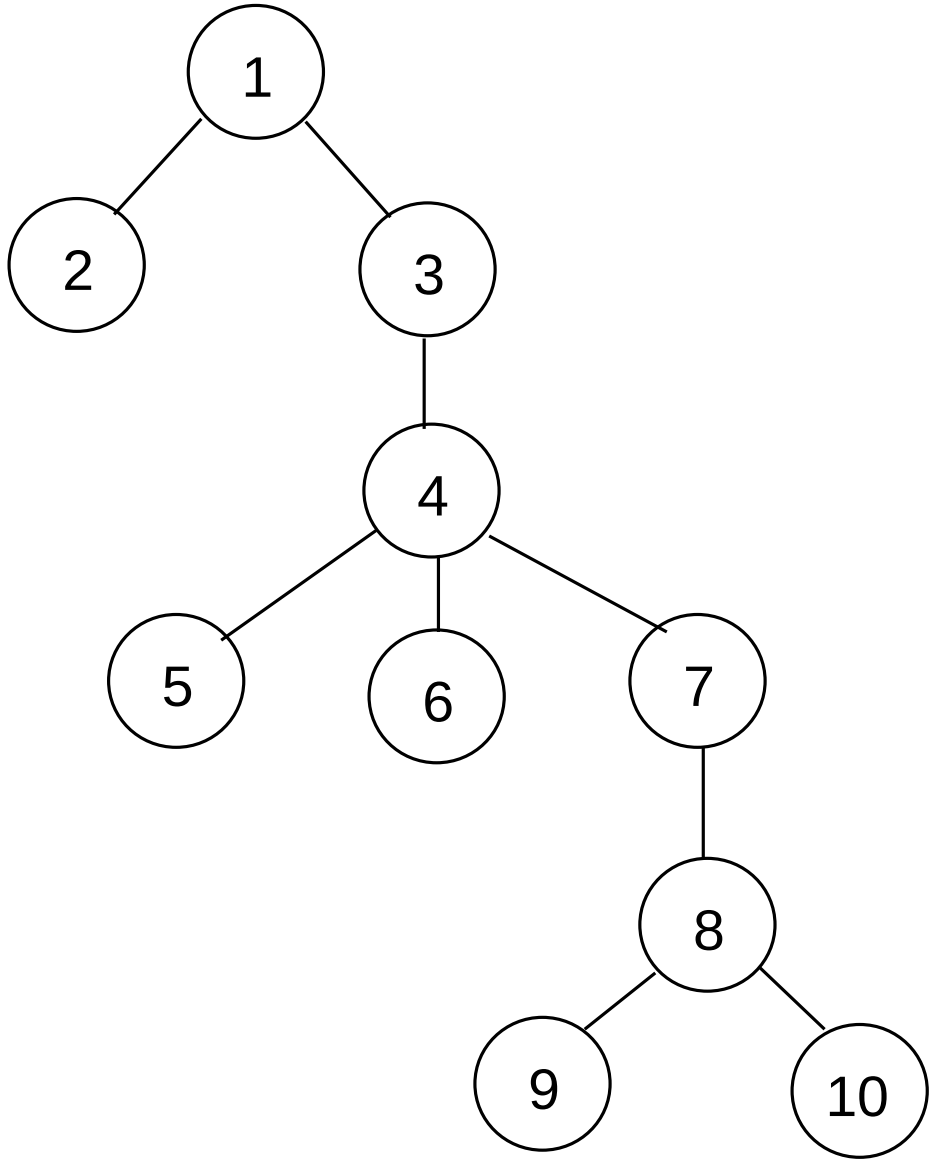
Definindo laços

- Um laço deve ter um único ponto de entrada (*header*)
  - O nó correspondente domina todos os nós do laço
- Deve existir pelo menos uma forma de iterar o laço
  - Através de um arco de retorno no grafo de fluxo (*back edge*)

- Algoritmo para encontrar laços
  - Buscar no grafo de fluxo arestas $a \rightarrow b$ cujo nó destino b domina o nó origem a

Encontrando laços





Movimentação de código

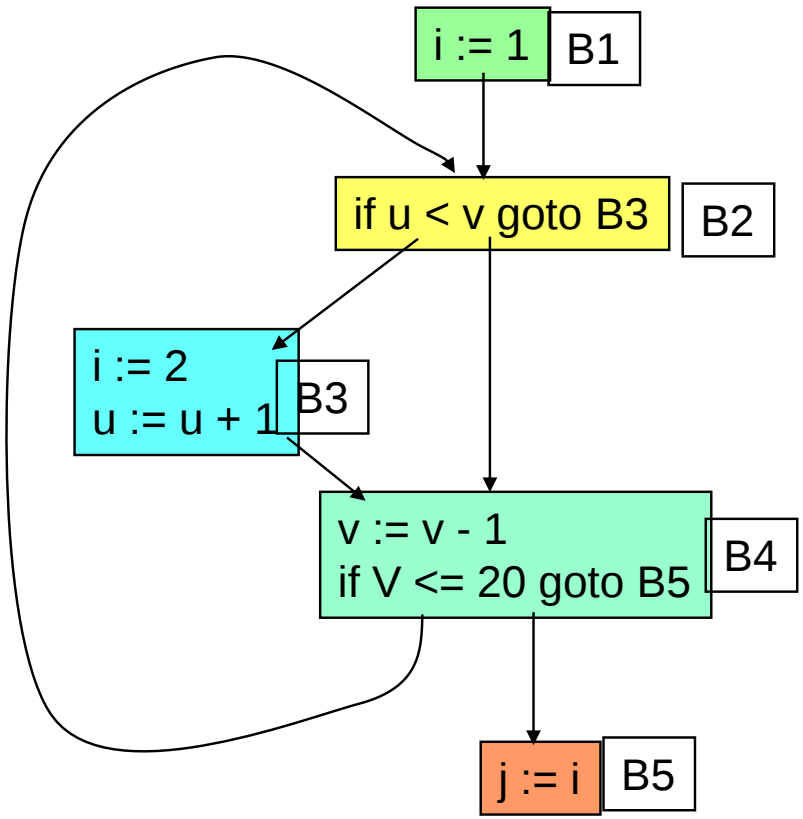
- Comando invariante: podem ser movidos para um bloco anterior ao laço porque seu valor não muda durante o laço

- Seja um comando invariante
  `s: x = y + z`
- Movimentação de `s` para fora do laço `L` é dita válida se
  - O bloco B que contém `s` domina todas as saídas de `L`
  - Nenhum outro comando no laço atribui um valor a `x`
  - Todos os usos de `x` dentro de `L` são feitos na definição em `s`

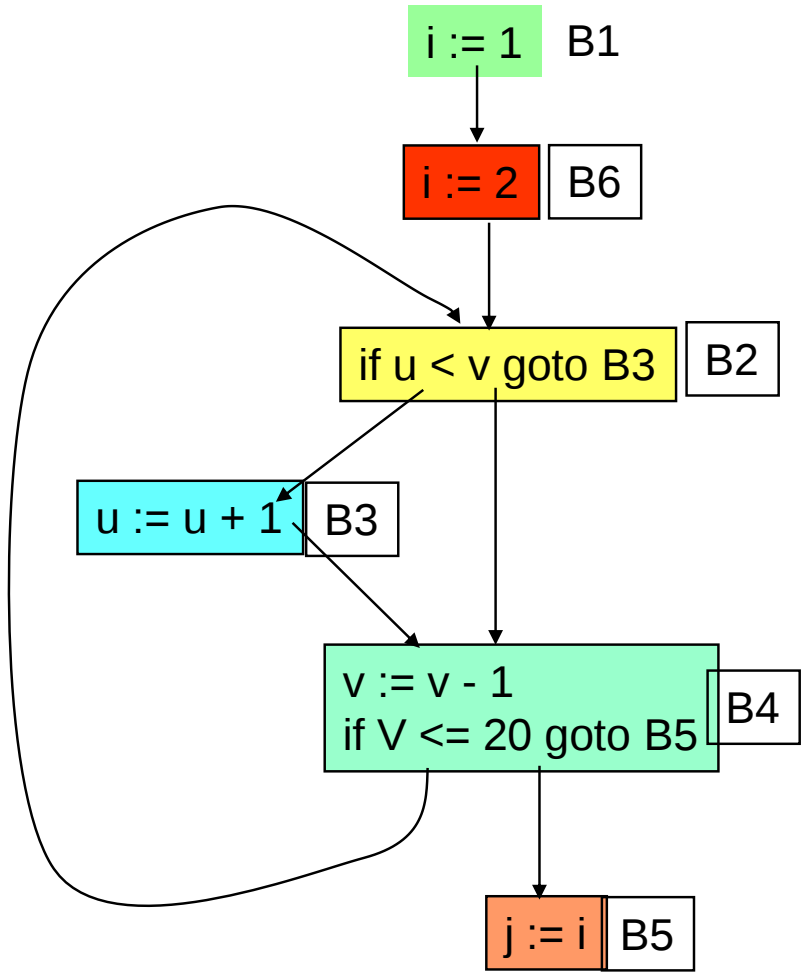
Movimentação ilegal de código

- O bloco B que contém s domina todas as saídas de L

- Original



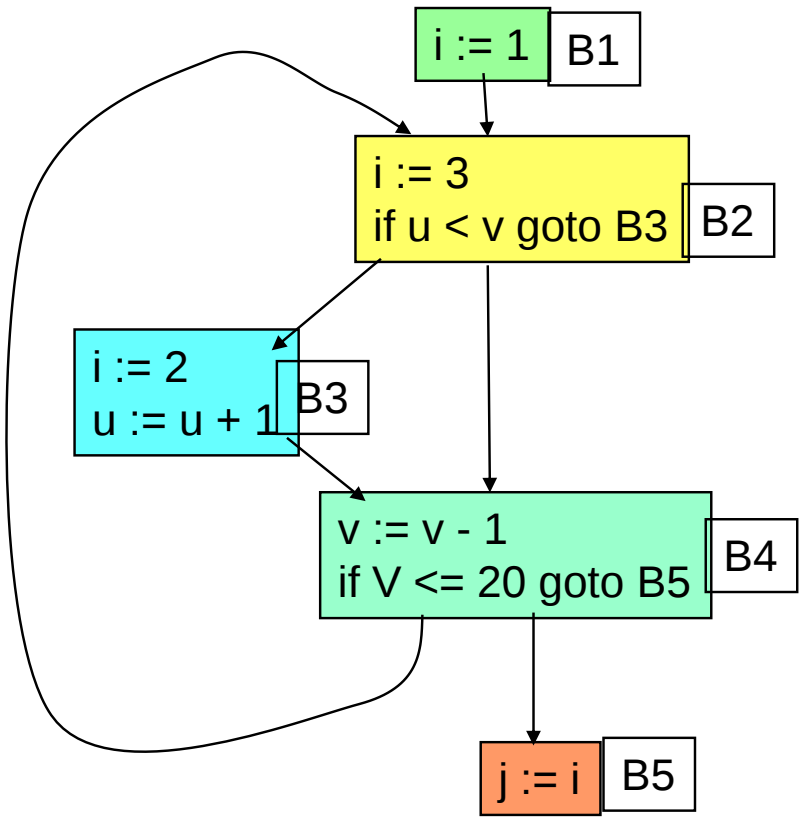
- Modificado



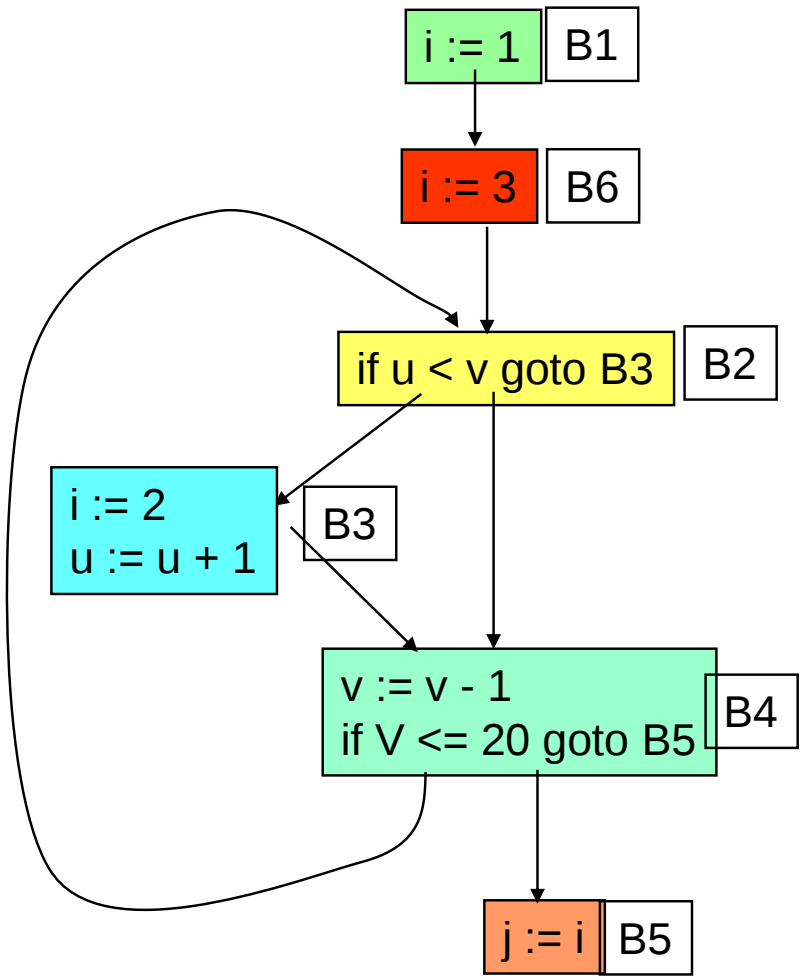
Movimentação ilegal de código

- Nenhum outro comando no laço atribui um valor a `x`

- Original



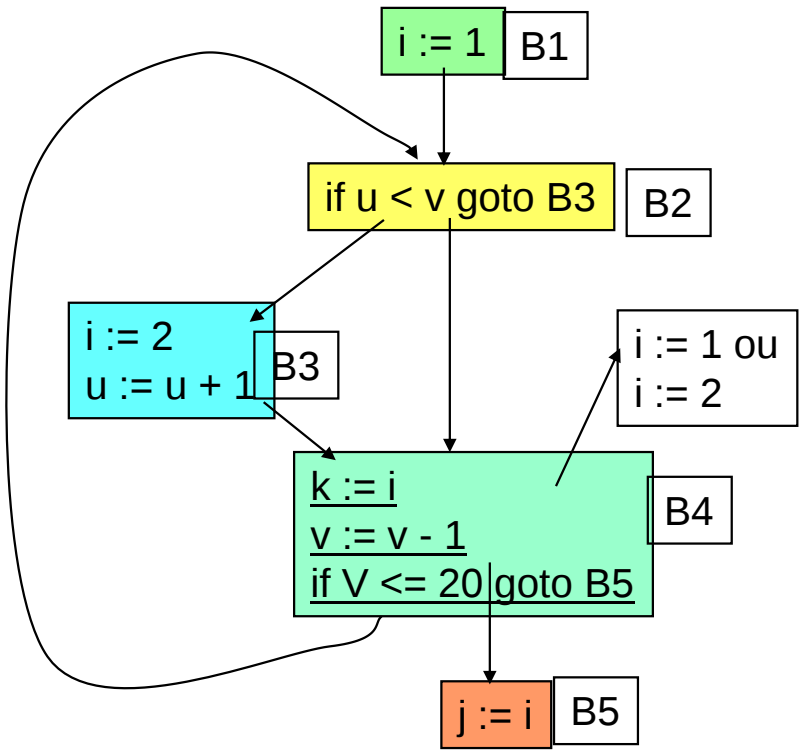
- Modificado



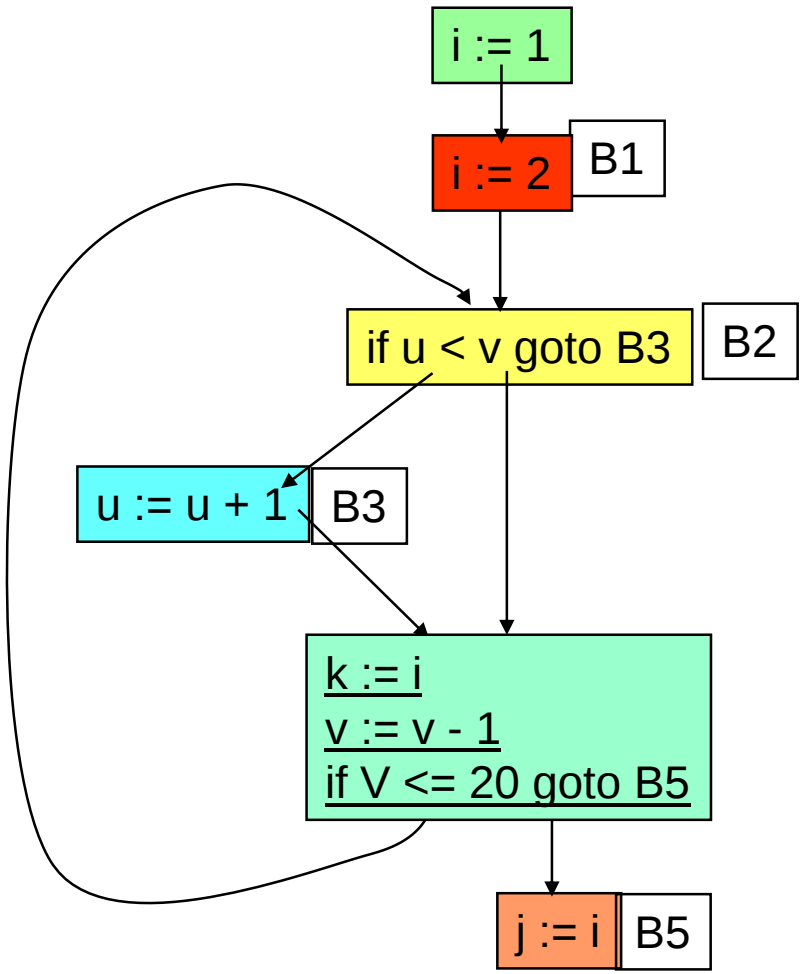
Movimentação ilegal de código

- Todos os usos de `x` dentro de `L` são feitos na definição em `s`

- Original

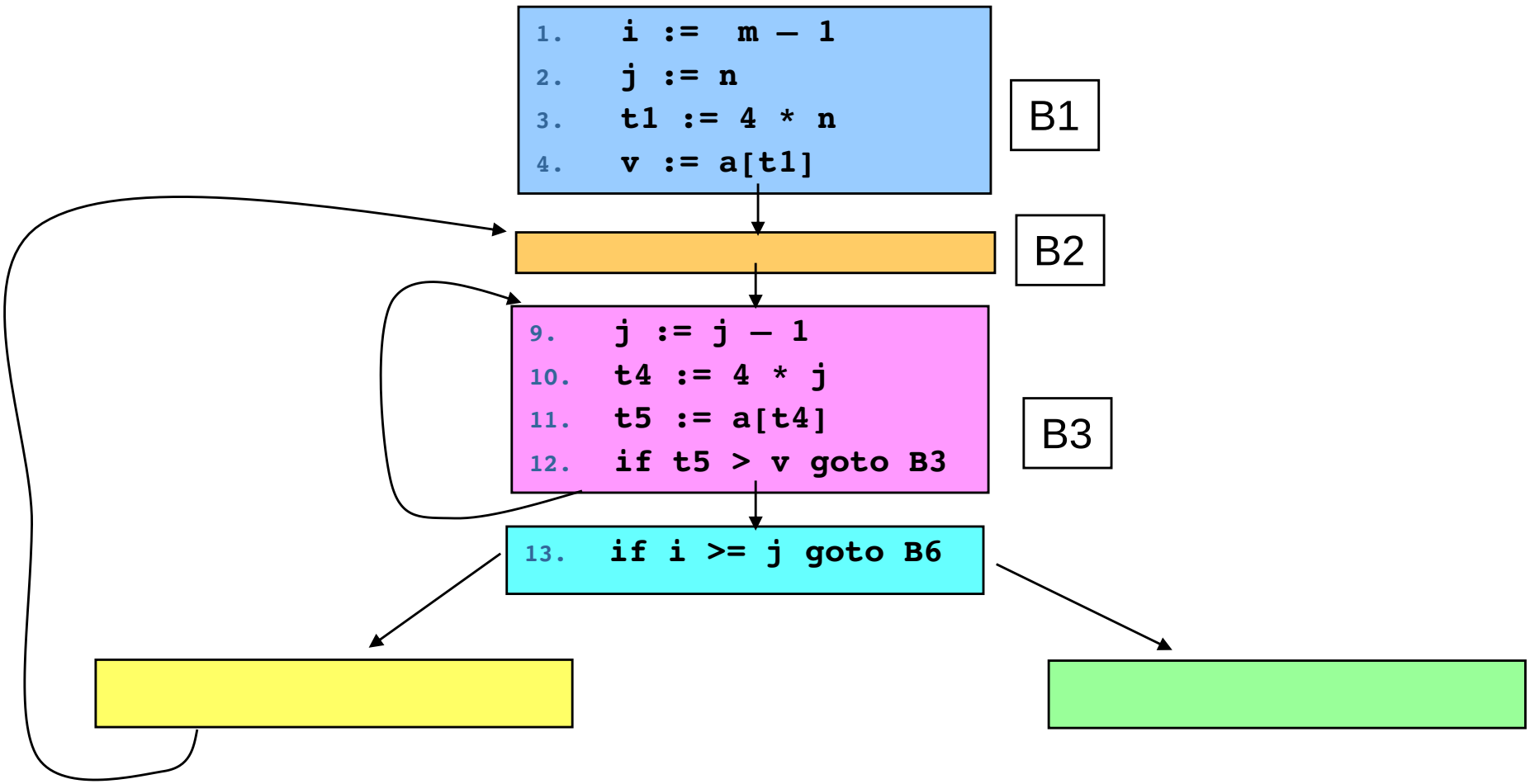


- Modificado



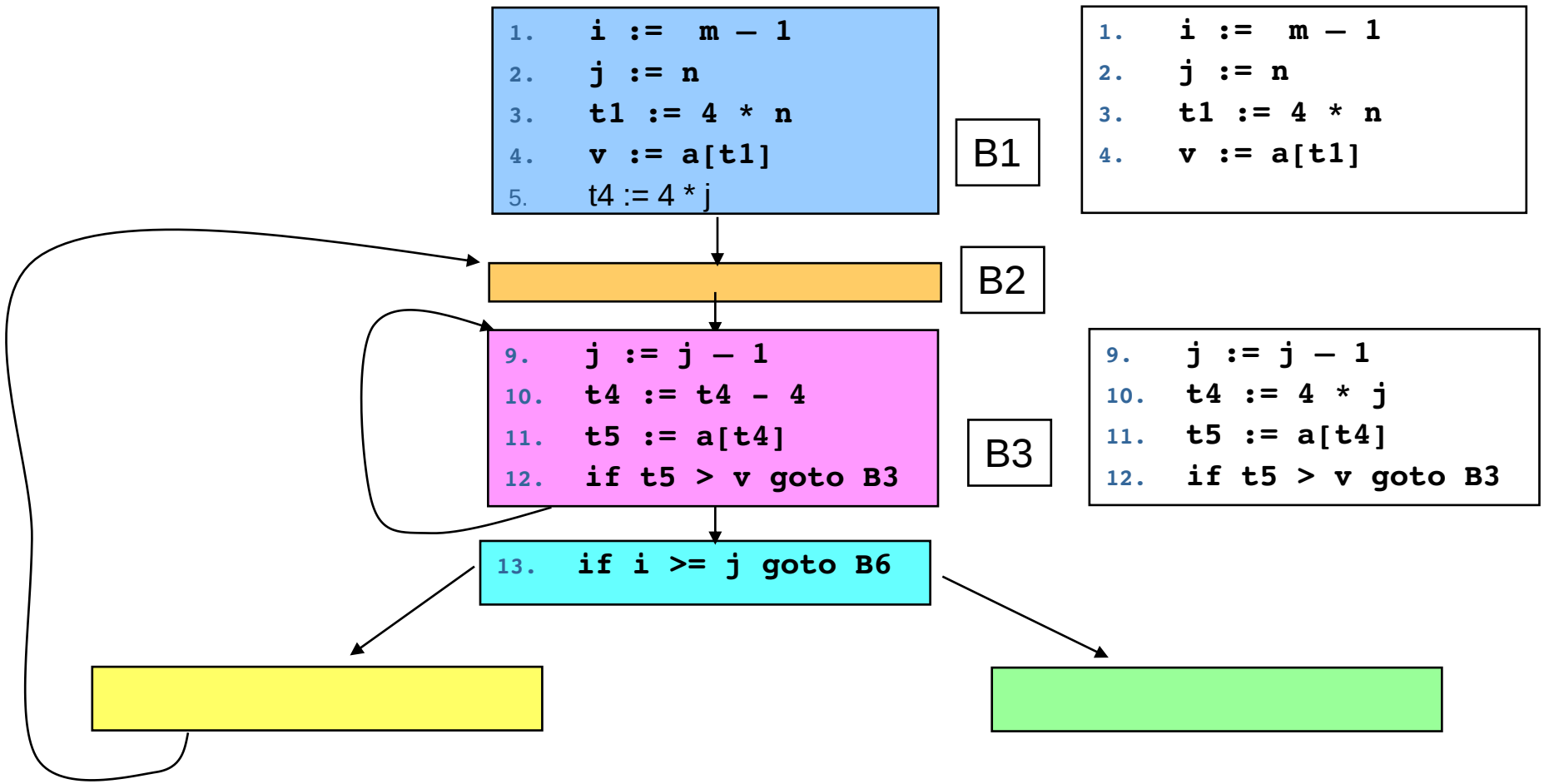
Variáveis de Indução

- São incrementadas/decrementadas por uma constante



Remover variáveis de indução – Redução de Força

- Elimina-se todas menos uma variável de indução
- Reduzir força $\rightarrow$ substituir operador por um mais barato



Exercício de Otimização de Laços

```
 1  n = 12
 2  m = 80
 3  j = 1
 4  if j > n goto 37
 5  i = 1
 6  if i > m goto 35
 7  t1 = 4 * j
 8  t2 = weight[t1]
 9  if i < t2 goto 33
10  t3 = 4 * i
11  t4 = cost[t3]
12  t5 = 4 * j
13  t6 = weight[t5]
14  t7 = i - t6
15  t8 = t7 * 4
16  t9 = cost[t8]
17  t10 = 4 * j
18  t11 = val[t10]
19  t12 = t9 + t11
20  if t4 >= t12 goto 33
21  t13 = 4 * j
22  t14 = weight[t13]
23  t15 = i - t14
24  t16 = t15 * 4
25  t17 = cost[t16]
26  t18 = 4 * j
27  t19 = val[t18]
28  t20 = t17 + t19
29  t21 = 4 * i
30  cost[t21] = t20
31  t22 = 4 * i
32  best[t22] = j
33  i = i + 1
34  goto 6
35  j = j + 1
36  goto 4
```

Conclusão

- Leituras Recomendadas para a aula de hoje
  - Livro do Dragão
    - Seções 8.4, 8.7
  - Livro do Keith
    - Capítulo 8 (introdução)
  - Série Didática
    - Capítulo 6

- Próxima Aula
  - Apresentação da etapa 6