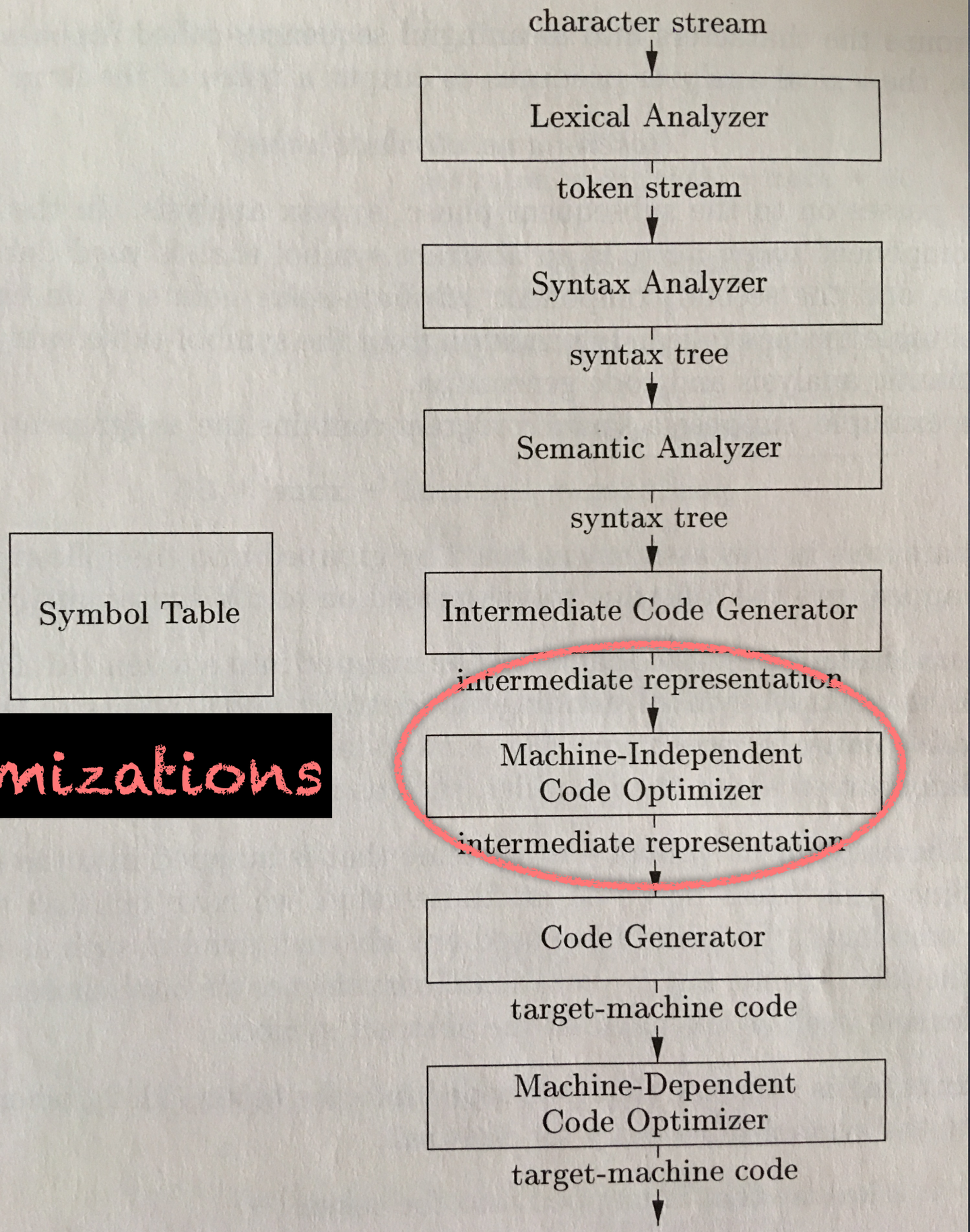
CSE443
Compilers

Dr. Carl Alphonce
alphonce@buffalo.edu
343 Davis Hall

Phases of a compiler



Optimizations

Figure 1.6, page 5 of text

Machine-Independent Optimizations

Chapter 8: local optimization
(within a basic block)

Chapter 9: global optimization
(across basic blocks)

Optimization

- The semantics of a program must be preserved by optimizations.
- The compiler does not know a programmer's intent - it can only reason about the program as written.

Why is there redundancy?
Why is there redundancy?

- May be addressable in source code.

  "a programmer may find it more direct and convenient to recalculate some result, leaving it the compiler to recognize that only one such calculation is necessary" [p. 584]

- Often cannot be resolved in HLL.

  e.g. "...programmers have no choice but to refer to elements of an array or fields in a structure through accesses like A[i][j]..." [p.584]

PLAN

- Discuss code transformations in context of a small code example (Sedgewick's Quicksort implmentation - see page 585)

Quicksort [p. 585]

```
void quicksort(int m, int n)
     /* recursively sorts a[m] through a[n] */
{
     int i, j;
     int v, x;
     if (n <= m) return;
     /* fragment begins here */
     i = m-1; j = n; v = a[n];
     while (1) {
          do i = i+1; while (a[i] < v);
          do j = j-1; while (a[j] > v);
          if (i >= j) break;
          x = a[i]; a[i] = a[j]; a[j] = x; /* swap a[i], a[j] */
     }
     /* fragment ends here */
     quicksort(m,j); quicksort(i+1,n);
}
```

```
01) i = m - 1
02) j = n
03) t1 = 4 * n
04) v = a[ t1 ]
05) i = i + 1
06) t2 = 4 * i
07) t3 = a[ t2 ]
08) if t3 < v goto 5 (=B2)
09) j = j - 1
10) t4 = 4 * j
11) t5 = a[ t4 ]
12) if t5 > v goto 9 (=B3)
13) if i >= j goto 23 (=B6)
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 (=B2)
23) t11 = 4 * i
24) x = a[ t11 ]
25) t12 = 4 * i
26) t13 = 4 * n
27) t14 = a[ t13 ]
28) a[ t12 ] = t14
29) t15 = 4 * n
30) a[ t15 ] = x
```

IR code for highlighted fragment of Quicksort

PLAN

- Original flow graph (9.3 - p. 587)
- Common Subexpression Elimination flow graph (9.5 - p. 589)
- Copy propagation flow graph (9.7 - p. 591)
- Dead code elimination (p. 591)
- Strength reduction flow graphs (9.8 - p. 593 and 9.9 - p. 594)

Flow graph for quicksort

B1

```
i = m - 1
j = n
t1 = 4 * n
v = a[ t1 ]
```

B2

```
i = i + 1
t2 = 4 * i
t3 = a[ t2 ]
if t3 < v goto B2
```

B3

```
j = j - 1
t4 = 4 * j
t5 = a[ t4 ]
if t5 > v goto B3
```

B4

```
if i >= j goto B6
```

B5

```
t6 = 4 * i
x = a[ t6 ]
t7 = 4 * i
t8 = 4 * j
t9 = a[ t8 ]
a[ t7 ] = t9
t10 = 4 * j
a[ t10 ] = x
goto B2
```

B6

```
t11 = 4 * i
x = a[ t11 ]
t12 = 4 * i
t13 = 4 * n
t14 = a[ t13 ]
a[ t12 ] = t14
t15 = 4 * n
a[ t15 ] = x
```

local common subexpressions in B5 (similarly in B6)

B1

```
i = m - 1
j = n
t1 = 4 * n
v = a[ t1 ]
```

B2

```
i = i + 1
t2 = 4 * i
t3 = a[ t2 ]
if t3 < v goto B2
```

B3

```
j = j - 1
t4 = 4 * j
t5 = a[ t4 ]
if t5 > v goto B3
```

B4

```
if i >= j goto B6
```

B5

```
t6 = 4 * i
x = a[ t6 ]
t7 = 4 * i
t8 = 4 * j
t9 = a[ t8 ]
a[ t7 ] = t9
t10 = 4 * j
a[ t10 ] = x
goto B2
```

B6

```
t11 = 4 * i
x = a[ t11 ]
t12 = 4 * i
t13 = 4 * n
t14 = a[ t13 ]
a[ t12 ] = t14
t15 = 4 * n
a[ t15 ] = x
```

local common subexpressions in B5 (similarly in B6)

B1

```
i = m - 1
j = n
t1 = 4 * n
v = a[ t1 ]
```

B2

```
i = i + 1
t2 = 4 * i
t3 = a[ t2 ]
if t3 < v goto B2
```

B3

```
j = j - 1
t4 = 4 * j
t5 = a[ t4 ]
if t5 > v goto B3
```

B4

```
if i >= j goto B6
```

B5

```
t6 = 4 * i
x = a[ t6 ]

t8 = 4 * j
t9 = a[ t8 ]
a[ t6 ] = t9

a[ t8 ] = x
goto B2
```

B6

```
t11 = 4 * i
x = a[ t11 ]
t12 = 4 * i
t13 = 4 * n
t14 = a[ t13 ]
a[ t12 ] = t14
t15 = 4 * n
a[ t15 ] = x
```

B1

```
i = m - 1
j = n
t1 = 4 * n
v = a[ t1 ]
```

B2

```
i = i + 1
t2 = 4 * i
t3 = a[ t2 ]
if t3 < v goto B2
```

B3

```
j = j - 1
t4 = 4 * j
t5 = a[ t4 ]
if t5 > v goto B3
```

B4

```
if i >= j goto B6
```

B5

local common subexpressions in B5 (similarly in B6)

```
t6 = 4 * i
x = a[ t6 ]

t8 = 4 * j
t9 = a[ t8 ]
a[ t6 ] = t9

a[ t8 ] = x
goto B2
```

B6

local common subexpressions in B6

```
t11 = 4 * i
x = a[ t11 ]
t12 = 4 * i
t13 = 4 * n
t14 = a[ t13 ]
a[ t12 ] = t14
t15 = 4 * n
a[ t15 ] = x
```

B1

```
i = m - 1
j = n
t1 = 4 * n
v = a[ t1 ]
```

B2

```
i = i + 1
t2 = 4 * i
t3 = a[ t2 ]
if t3 < v goto B2
```

B3

```
j = j - 1
t4 = 4 * j
t5 = a[ t4 ]
if t5 > v goto B3
```

B4

```
if i >= j goto B6
```

B5

```
t6 = 4 * i
x = a[ t6 ]
t8 = 4 * j
t9 = a[ t8 ]
a[ t6 ] = t9
a[ t8 ] = x
goto B2
```

B6

```
t11 = 4 * i
x = a[ t11 ]
t13 = 4 * n
t14 = a[ t13 ]
a[ t11 ] = t14
a[ t13 ] = x
```

local common subexpressions in B6

global common subexpressions

B1

```
i = m - 1
j = n
t1 = 4 * n
v = a[ t1 ]
```

B2

```
i = i + 1
t2 = 4 * i
t3 = a[ t2 ]
if t3 < v goto B2
```

B3

```
j = j - 1
t4 = 4 * j
t5 = a[ t4 ]
if t5 > v goto B3
```

B4

```
if i >= j goto B6
```

B5

```
t6 = 4 * i
x = a[ t6 ]
t8 = 4 * j
t9 = a[ t8 ]
a[ t6 ] = t9
a[ t8 ] = x
goto B2
```

B6

```
t11 = 4 * i
x = a[ t11 ]
t13 = 4 * n
t14 = a[ t13 ]
a[ t11 ] = t14
a[ t13 ] = x
```

global common subexpressions

B1

```
i = m - 1
j = n
t1 = 4 * n
v = a[ t1 ]
```

B2

```
i = i + 1
t2 = 4 * i
t3 = a[ t2 ]
if t3 < v goto B2
```

B3

```
j = j - 1
t4 = 4 * j
t5 = a[ t4 ]
if t5 > v goto B3
```

B4

```
if i >= j goto B6
```

B5

```
t6 = 4 * i
x = a[ t6 ]

t9 = a[ t4 ]
a[ t6 ] = t9
a[ t4 ] = x
goto B2
```

B6

```
t11 = 4 * i
x = a[ t11 ]
t13 = 4 * n
t14 = a[ t13 ]
a[ t11 ] = t14
a[ t13 ] = x
```

global common subexpressions

B1

```
i = m - 1
j = n
t1 = 4 * n
v = a[ t1 ]
```

B2

```
i = i + 1
t2 = 4 * i
t3 = a[ t2 ]
if t3 < v goto B2
```

B3

```
j = j - 1
t4 = 4 * j
t5 = a[ t4 ]
if t5 > v goto B3
```

B4

```
if i >= j goto B6
```

B5

```
t6 = 4 * i
x = a[ t6 ]

t9 = a[ t4 ]
a[ t6 ] = t9
a[ t4 ] = x
goto B2
```

B6

```
t11 = 4 * i
x = a[ t11 ]
t13 = 4 * n
t14 = a[ t13 ]
a[ t11 ] = t14
a[ t13 ] = x
```

global common subexpressions

B1

```
i = m - 1
j = n
t1 = 4 * n
v = a[ t1 ]
```

B2

```
i = i + 1
t2 = 4 * i
t3 = a[ t2 ]
if t3 < v goto B2
```

B3

```
j = j - 1
t4 = 4 * j
t5 = a[ t4 ]
if t5 > v goto B3
```

B4

```
if i >= j goto B6
```

B5

```
t6 = 4 * i
x = a[ t6 ]

a[ t6 ] = t5
a[ t4 ] = x
goto B2
```

B6

```
t11 = 4 * i
x = a[ t11 ]
t13 = 4 * n
t14 = a[ t13 ]
a[ t11 ] = t14
a[ t13 ] = x
```

global common subexpressions

B1

```
i = m - 1
j = n
t1 = 4 * n
v = a[ t1 ]
```

B2

```
i = i + 1
t2 = 4 * i
t3 = a[ t2 ]
if t3 < v goto B2
```

B3

```
j = j - 1
t4 = 4 * j
t5 = a[ t4 ]
if t5 > v goto B3
```

B4

```
if i >= j goto B6
```

B5

```
t6 = 4 * i
x = a[ t6 ]

a[ t6 ] = t5
a[ t4 ] = x
goto B2
```

B6

```
t11 = 4 * i
x = a[ t11 ]
t13 = 4 * n
t14 = a[ t13 ]
a[ t11 ] = t14
a[ t13 ] = x
```

global common subexpressions

B1

```
i = m - 1
j = n
t1 = 4 * n
v = a[ t1 ]
```

B2

```
i = i + 1
t2 = 4 * i
t3 = a[ t2 ]
if t3 < v goto B2
```

B3

```
j = j - 1
t4 = 4 * j
t5 = a[ t4 ]
if t5 > v goto B3
```

B4

```
if i >= j goto B6
```

B5

```
x = t3

a[ t2 ] = t5
a[ t4 ] = x
goto B2
```

B6

```
t11 = 4 * i
x = a[ t11 ]
t13 = 4 * n
t14 = a[ t13 ]
a[ t11 ] = t14
a[ t13 ] = x
```

global common subexpressions

B1

```
i = m - 1
j = n
t1 = 4 * n
v = a[ t1 ]
```

B2

```
i = i + 1
t2 = 4 * i
t3 = a[ t2 ]
if t3 < v goto B2
```

B3

```
j = j - 1
t4 = 4 * j
t5 = a[ t4 ]
if t5 > v goto B3
```

B4

```
if i >= j goto B6
```

B5

```
x = t3
a[ t2 ] = t5
a[ t4 ] = x
goto B2
```

B6

```
t11 = 4 * i
x = a[ t11 ]
t13 = 4 * n
t14 = a[ t13 ]
a[ t11 ] = t14
a[ t13 ] = x
```

B1

```
i = m - 1
j = n
t1 = 4 * n
v = a[ t1 ]
```

B2

```
i = i + 1
t2 = 4 * i
t3 = a[ t2 ]
if t3 < v goto B2
```

B3

```
j = j - 1
t4 = 4 * j
t5 = a[ t4 ]
if t5 > v goto B3
```

B4

```
if i >= j goto B6
```

B5

```
x = t3
a[ t2 ] = t5
a[ t4 ] = x
goto B2
```

B6

```
t11 = 4 * i
x = a[ t11 ]
t13 = 4 * n
t14 = a[ t13 ]
a[ t11 ] = t14
a[ t13 ] = x
```

global common subexpressions

B1

```
i = m - 1
j = n
t1 = 4 * n
v = a[ t1 ]
```

B2

```
i = i + 1
t2 = 4 * i
t3 = a[ t2 ]
if t3 < v goto B2
```

B3

```
j = j - 1
t4 = 4 * j
t5 = a[ t4 ]
if t5 > v goto B3
```

B4

```
if i >= j goto B6
```

B5

```
x = t3
a[ t2 ] = t5
a[ t4 ] = x
goto B2
```

B6

```
x = a[ t2 ]
t13 = 4 * n
t14 = a[ t13 ]
a[ t2 ] = t14
a[ t13 ] = x
```

global common subexpressions

B1

```
i = m - 1
j = n
t1 = 4 * n
v = a[ t1 ]
```

B2

```
i = i + 1
t2 = 4 * i
t3 = a[ t2 ]
if t3 < v goto B2
```

B3

```
j = j - 1
t4 = 4 * j
t5 = a[ t4 ]
if t5 > v goto B3
```

B4

```
if i >= j goto B6
```

B5

```
x = t3
a[ t2 ] = t5
a[ t4 ] = x
goto B2
```

B6

```
x = a[ t2 ]

t14 = a[ t1 ]
a[ t2 ] = t14
a[ t1 ] = x
```

global common subexpressions

B1

```
i = m - 1
j = n
t1 = 4 * n
v = a[ t1 ]
```

B2

```
i = i + 1
t2 = 4 * i
t3 = a[ t2 ]
if t3 < v goto B2
```

B3

```
j = j - 1
t4 = 4 * j
t5 = a[ t4 ]
if t5 > v goto B3
```

B4

```
if i >= j goto B6
```

B5

```
x = t3
a[ t2 ] = t5
a[ t4 ] = x
goto B2
```

B6

```
x = t3

t14 = a[ t1 ]
a[ t2 ] = t14
a[ t1 ] = x
```

global common subexpressions

B1

```
i = m - 1
j = n
t1 = 4 * n
v = a[ t1 ]
```

B2

```
i = i + 1
t2 = 4 * i
t3 = a[ t2 ]
if t3 < v goto B2
```

B3

```
j = j - 1
t4 = 4 * j
t5 = a[ t4 ]
if t5 > v goto B3
```

B4

```
if i >= j goto B6
```

B5

```
x = t3
a[ t2 ] = t5
a[ t4 ] = x
goto B2
```

B6

```
x = t3
t14 = a[ t1 ]
a[ t2 ] = t14
a[ t1 ] = x
```

global common subexpressions

Copy propagation

B1

```
i = m - 1
j = n
t1 = 4 * n
v = a[ t1 ]
```

B2

```
i = i + 1
t2 = 4 * i
t3 = a[ t2 ]
if t3 < v goto B2
```

B3

```
j = j - 1
t4 = 4 * j
t5 = a[ t4 ]
if t5 > v goto B3
```

B4

```
if i >= j goto B6
```

B5

```
x = t3
a[ t2 ] = t5
a[ t4 ] = x
goto B2
```

B6

```
x = t3
t14 = a[ t1 ]
a[ t2 ] = t14
a[ t1 ] = x
```

Copy propagation

B1

```
i = m - 1
j = n
t1 = 4 * n
v = a[ t1 ]
```

B2

```
i = i + 1
t2 = 4 * i
t3 = a[ t2 ]
if t3 < v goto B2
```

B3

```
j = j - 1
t4 = 4 * j
t5 = a[ t4 ]
if t5 > v goto B3
```

B4

```
if i >= j goto B6
```

B5

```
x = t3
a[ t2 ] = t5
a[ t4 ] = t3
goto B2
```

B6

```
x = t3
t14 = a[ t1 ]
a[ t2 ] = t14
a[ t1 ] = t3
```

Dead code elimination

B1

```
i = m - 1
j = n
t1 = 4 * n
v = a[ t1 ]
```

B2

```
i = i + 1
t2 = 4 * i
t3 = a[ t2 ]
if t3 < v goto B2
```

B5

```
x = t3
a[ t2 ] = t5
a[ t4 ] = t3
goto B2
```

B3

```
j = j - 1
t4 = 4 * j
t5 = a[ t4 ]
if t5 > v goto B3
```

B6

```
x = t3
t14 = a[ t1 ]
a[ t2 ] = t14
a[ t1 ] = t3
```

B4

```
if i >= j goto B6
```

Dead code elimination

B1

```
i = m - 1
j = n
t1 = 4 * n
v = a[ t1 ]
```

B2

```
i = i + 1
t2 = 4 * i
t3 = a[ t2 ]
if t3 < v goto B2
```

B3

```
j = j - 1
t4 = 4 * j
t5 = a[ t4 ]
if t5 > v goto B3
```

B4

```
if i >= j goto B6
```

B5

```
a[ t2 ] = t5
a[ t4 ] = t3
goto B2
```

B6

```
t14 = a[ t1 ]
a[ t2 ] = t14
a[ t1 ] = t3
```

Dead code elimination

B1

```
i = m - 1
j = n
t1 = 4 * n
v = a[ t1 ]
```

B2

```
i = i + 1
t2 = 4 * i
t3 = a[ t2 ]
if t3 < v goto B2
```

B3

```
j = j - 1
t4 = 4 * j
t5 = a[ t4 ]
if t5 > v goto B3
```

B4

```
if i >= j goto B6
```

B5

```
a[ t2 ] = t5
a[ t4 ] = t3
goto B2
```

B6

```
t14 = a[ t1 ]
a[ t2 ] = t14
a[ t1 ] = t3
```

Induction variables and reduction in strength

B1

```
i = m - 1
j = n
t1 = 4 * n
v = a[ t1 ]
```

B2

```
i = i + 1
t2 = 4 * i
t3 = a[ t2 ]
if t3 < v goto B2
```

B3

```
j = j - 1
t4 = 4 * j
t5 = a[ t4 ]
if t5 > v goto B3
```

B4

```
if i >= j goto B6
```

B5

```
a[ t2 ] = t5
a[ t4 ] = t3
goto B2
```

B6

```
t14 = a[ t1 ]
a[ t2 ] = t14
a[ t1 ] = t3
```

Whenever j changes, so does t4.
* not necessary!

B1

```
i = m - 1
j = n
t1 = 4 * n
v = a[ t1 ]
t4 = 4 * j
```

Establish relationship between t4 and j

B2

```
i = i + 1
t2 = 4 * i
t3 = a[ t2 ]
if t3 < v goto B2
```

Induction variables and reduction in strength

B3

```
j = j - 1
t4 = t4 - 4
t5 = a[ t4 ]
if t5 > v goto B3
```

Decrement t4 by 4 each time j decrements by 1

B4

```
if i >= j goto B6
```

B5

```
a[ t2 ] = t5
a[ t4 ] = t3
goto B2
```

B6

```
t14 = a[ t1 ]
a[ t2 ] = t14
a[ t1 ] = t3
```

Induction variables and reduction in strength

B1

```
i = m - 1
j = n
t1 = 4 * n
v = a[ t1 ]
t4 = 4 * j
```

B2

```
i = i + 1
t2 = 4 * i
t3 = a[ t2 ]
if t3 < v goto B2
```

B3

```
j = j - 1
t4 = t4 - 4
t5 = a[ t4 ]
if t5 > v goto B3
```

B4

```
if i >= j goto B6
```

B5

```
a[ t6 ] = t5
a[ t8 ] = t3
goto B2
```

B6

```
t14 = a[ t1 ]
a[ t2 ] = t14
a[ t1 ] = t3
```

Induction variables and reduction in strength

B1

```
i = m - 1
j = n
t1 = 4 * n
v = a[ t1 ]
t4 = 4 * j
t2 = 4 * i
```

Establish relationship between t2 and I

B2

```
i = i + 1
t2 = t2 + 4
t3 = a[ t2 ]
if t3 < v goto B2
```

Increment t2 by 4 each time i increments by 1

B5

```
a[ t2 ] = t5
a[ t4 ] = t3
goto B2
```

B6

```
t14 = a[ t1 ]
a[ t2 ] = t14
a[ t1 ] = t3
```

B3

```
j = j - 1
t4 = t4 - 4
t5 = a[ t4 ]
if t5 > v goto B3
```

B4

```
if i >= j goto B6
```

B1

```
i = m - 1
j = n
t1 = 4 * n
v = a[ t1 ]
t4 = 4 * j
t2 = 4 * i
```

B2

```
i = i + 1
t2 = t2 + 4
t3 = a[ t2 ]
if t3 < v goto B2
```

B5

```
a[ t2 ] = t5
a[ t4 ] = t3
goto B2
```

B3

```
j = j - 1
t4 = t4 - 4
t5 = a[ t4 ]
if t5 > v goto B3
```

B4

```
if i >= j goto B6
```

B6

```
t14 = a[ t1 ]
a[ t2 ] = t14
a[ t1 ] = t3
```

Compare to initial graph

B1

```
i = m - 1
j = n
t1 = 4 * n
v = a[ t1 ]
```

B2

```
i = i + 1
t2 = 4 * i
t3 = a[ t2 ]
if t3 < v goto B2
```

B3

```
j = j - 1
t4 = 4 * j
t5 = a[ t4 ]
if t5 > v goto B3
```

B4

```
if i >= j goto B6
```

B5

```
t6 = 4 * i
x = a[ t6 ]
t7 = 4 * i
t8 = 4 * j
t9 = a[ t8 ]
a[ t7 ] = t9
t10 = 4 * j
a[ t10 ] = x
goto B2
```

B6

```
t11 = 4 * i
x = a[ t11 ]
t12 = 4 * i
t13 = 4 * n
t14 = a[ t13 ]
a[ t12 ] = t14
t15 = 4 * n
a[ t15 ] = x
```