Single Static Assignment

CSE 401 Section 10/10

Aaron Johnston & Nate Yazdani

Adapted from Laura Vonesson's Wi17 Slides

The Final Stretch

 You are here

SUN	MON	TUE	WED	THU	FRI	SAT
				 Compiler Additions		Report M501 Additions
M501 Report Evals!!			Review Session (4:30 EEB 045)	**Final Exam** (8:30)	**Eternal Mastery of Compilers** →	

Problem 1

(review of dataflow)

Single Static Assignment

- An intermediate representation where each variable has only one definition:

Original		**SSA Form**
a := x + y		a_1 := x + y
b := a – 1	⇨	b_1 := a_1 – 1
a := y + b		a_2 := y + b_1
b := x * 4		b_2 := x * 4
a := a + b		a_3 := a_2 + b_2

SSA: Why We Love It

- Without SSA, all definitions and uses of a variable get mixed together
  - Computing information about the definitions of a variable is an expensive but necessary part of many dataflow analyses

- Doing the work of converting to SSA once makes many analyses + optimizations more efficient
  - SSA can be thought of as an implicit representation of Definition/Use chains

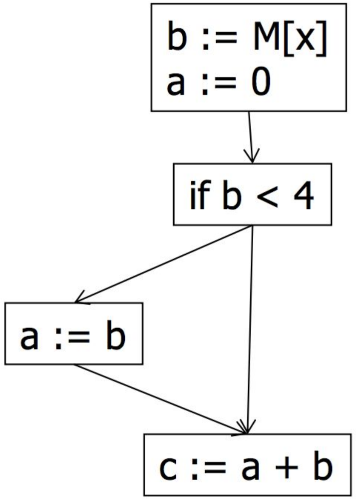
SSA: Why We Love It

- Ex: Dead Store Elimination
  - Without SSA: Compute live variables at every point, which requires working backwards and using the dataflow sets to check for *any path* that does not kill the variable, and eliminate any stores that are not to a live variable.
  - With SSA: Eliminate any store where the variable being assigned has 0 uses.

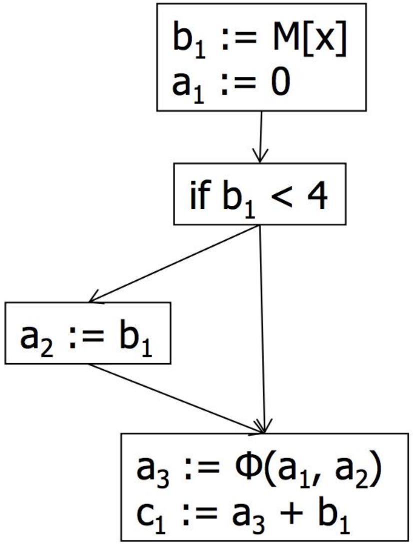
Phi-Functions

- A method of representing an uncertain value for a certain definition
  - Not a “real” instruction -- only a formality needed for SSA

Original



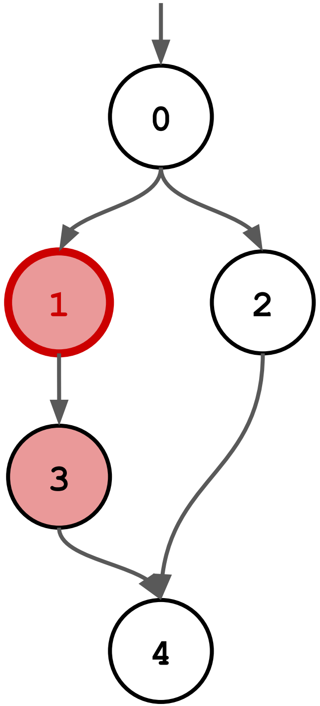
SSA Form



Dominators

- A node **X** ***dominates*** a node **Y** iff every path from the entry point of the control flow graph to **Y** includes **X**

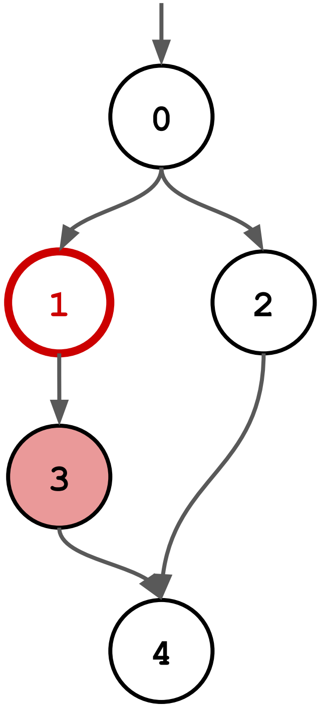
Node 1 dominates nodes 1 and 3.
It does not dominate 4 because there is another path that reaches it.



Strict Dominance

- A node **`X`** ***strictly dominates*** a node **`Y`** if **`X`** dominates **`Y`** and **`X`** ≠ **`Y`**.

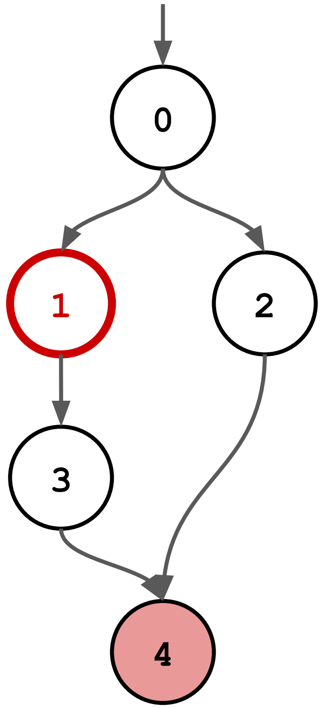
Node 1 only strictly dominates node 3 because it is the only dominated node that is not equal to 1.



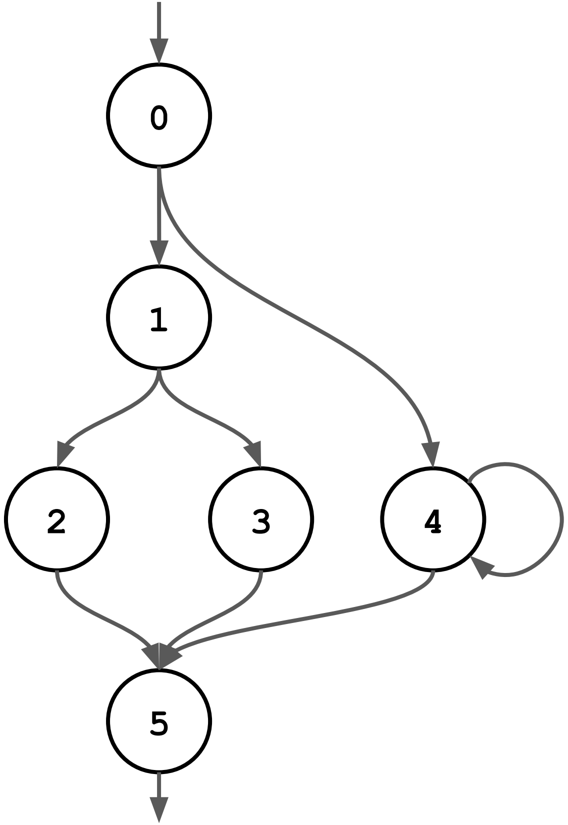
Dominance Frontiers

- A node **`Y`** is in the ***dominance frontier*** of node **`X`** if **`X`** dominates an immediate predecessor of **`Y`** but **`X`** does not strictly dominate **`Y`**.
- Essentially, the border between dominated and non-dominated nodes
    - Note: a node can be in its own dominance frontier
- This is where phi function merging is necessary

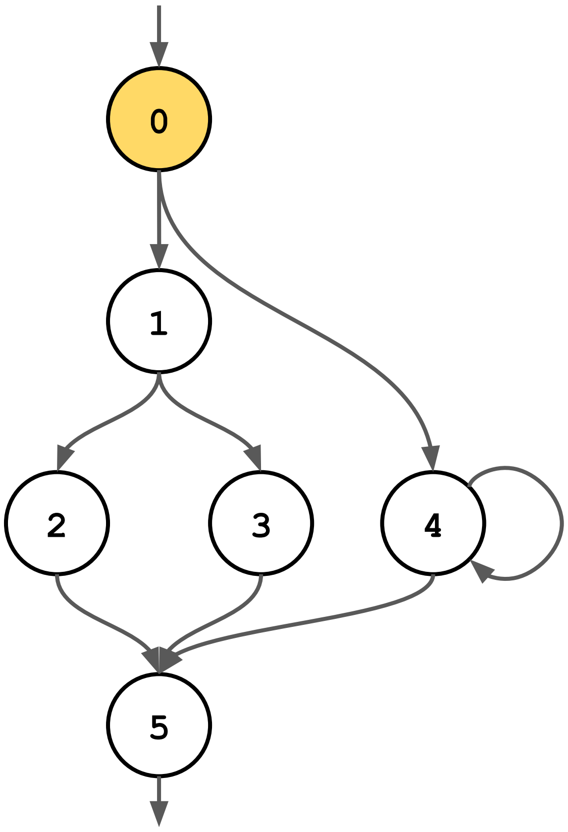
Node 4 is in the dominance frontier of node 1 because an immediate predecessor (node 3) is dominated by 1.



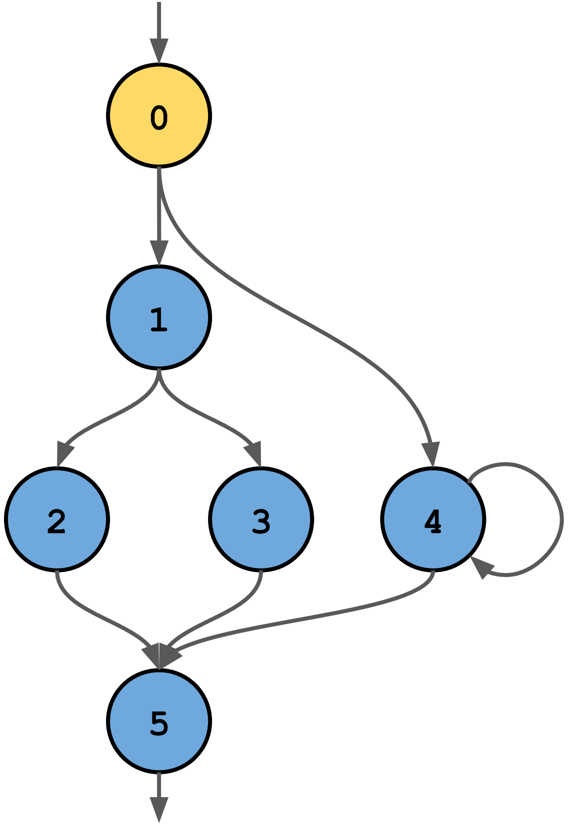
Problem 2



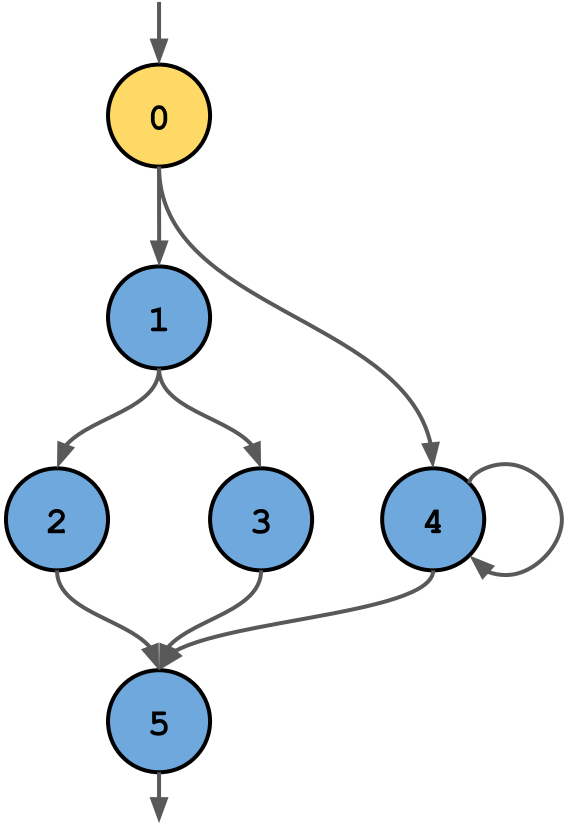
NODE	STRICTLY DOMINATES	DOMINANCE FRONTIER
0		
1		
2		
3		
4		
5		



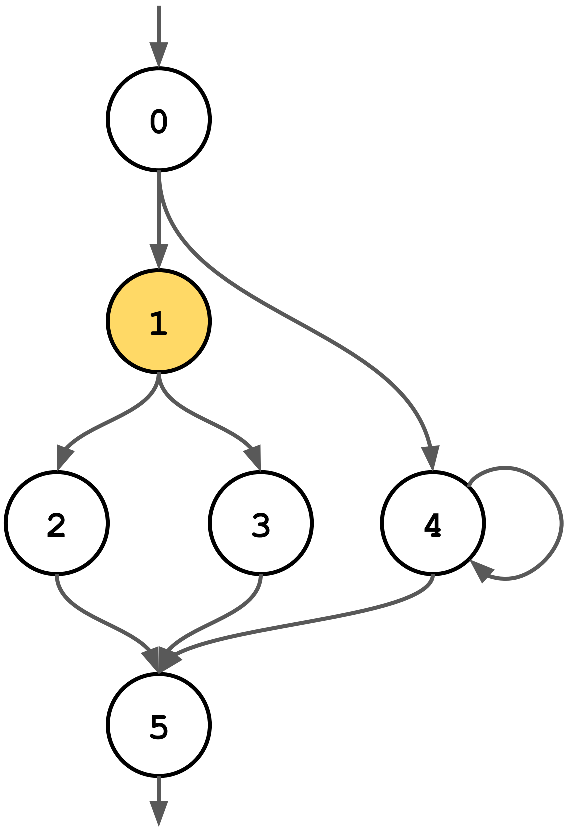
NODE	STRICTLY DOMINATES	DOMINANCE FRONTIER
0		
1		
2		
3		
4		
5		



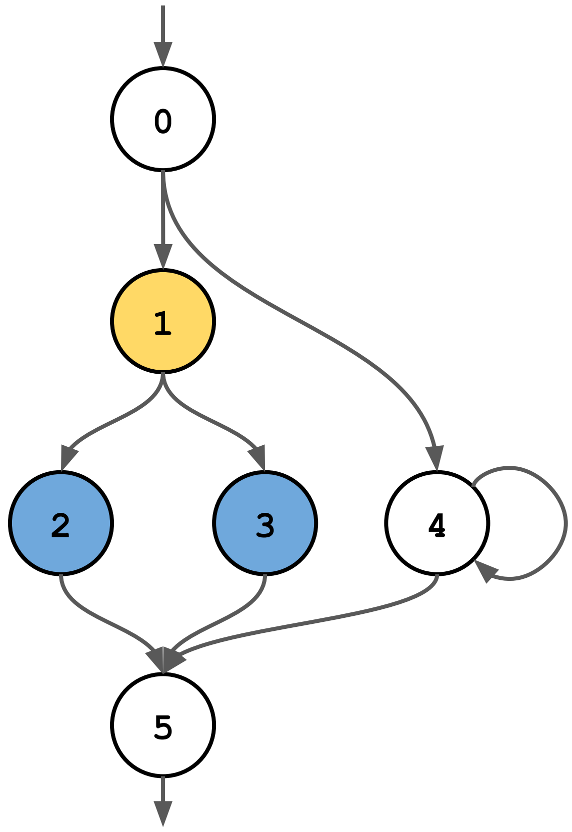
NODE	STRICTLY DOMINATES	DOMINANCE FRONTIER
0	1, 2, 3, 4, 5	
1		
2		
3		
4		
5		



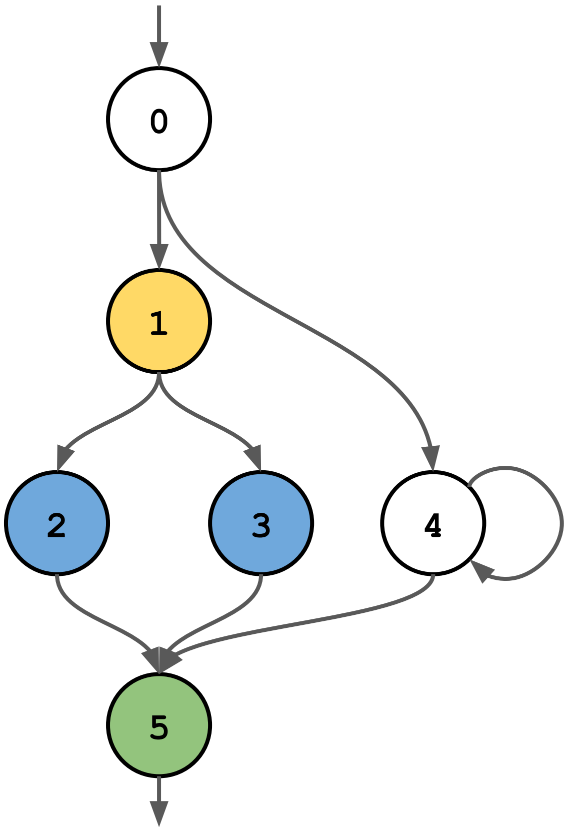
NODE	STRICTLY DOMINATES	DOMINANCE FRONTIER
0	1, 2, 3, 4, 5	∅
1		
2		
3		
4		
5		



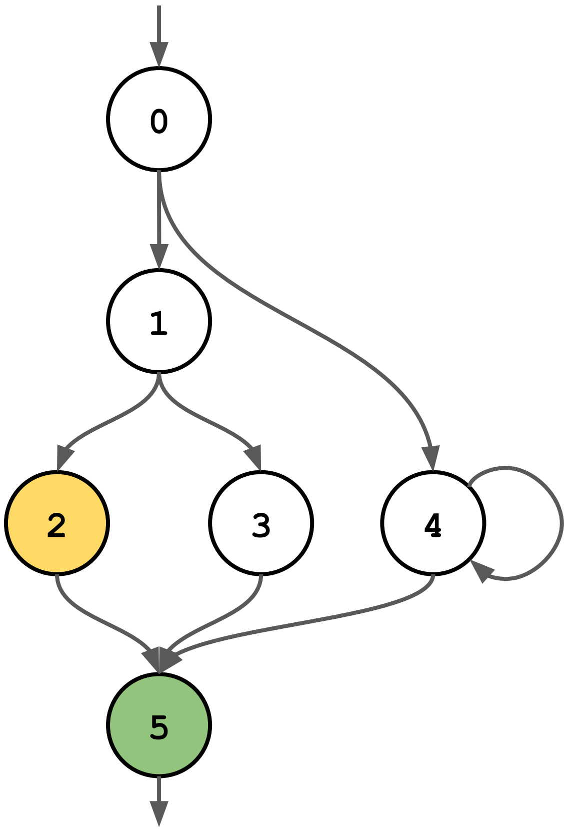
NODE	STRICTLY DOMINATES	DOMINANCE FRONTIER
0	1, 2, 3, 4, 5	∅
1		
2		
3		
4		
5		



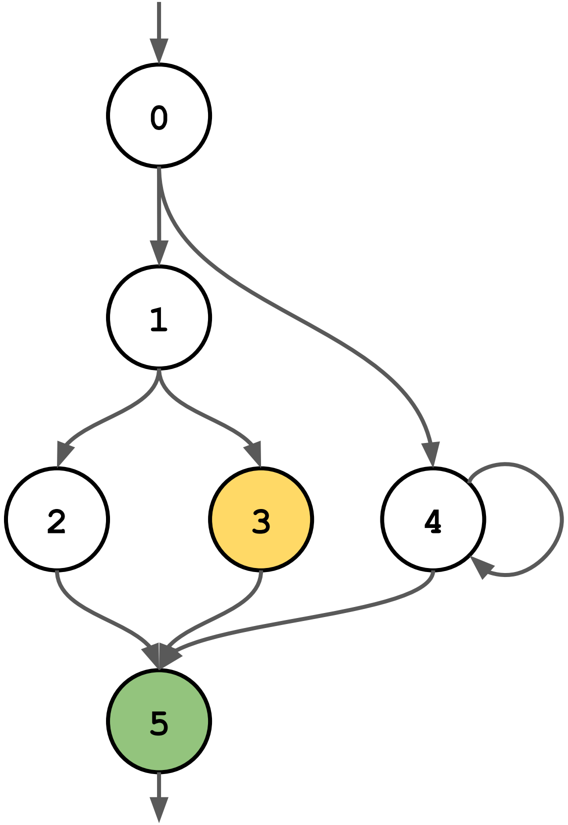
NODE	STRICTLY DOMINATES	DOMINANCE FRONTIER
0	1, 2, 3, 4, 5	∅
1	2, 3	
2		
3		
4		
5		



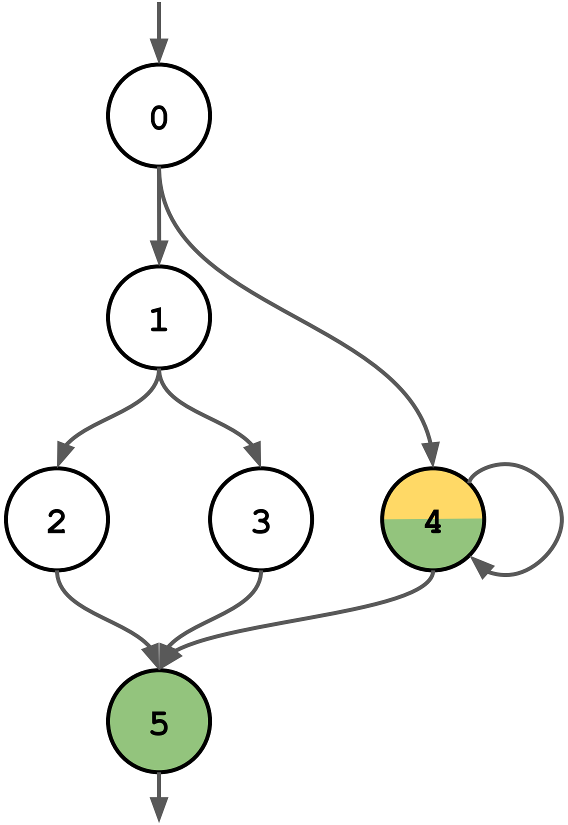
NODE	STRICTLY DOMINATES	DOMINANCE FRONTIER
0	1, 2, 3, 4, 5	∅
1	2, 3	5
2		
3		
4		
5		



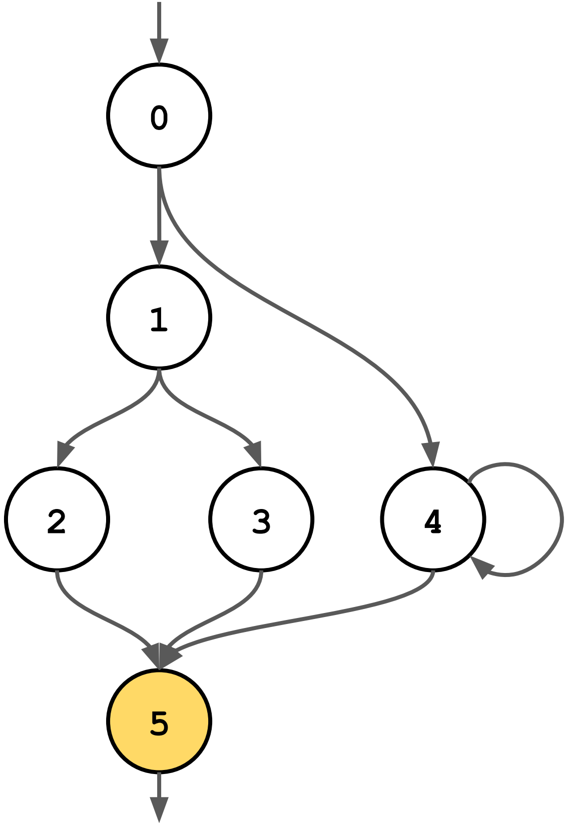
NODE	STRICTLY DOMINATES	DOMINANCE FRONTIER
0	1, 2, 3, 4, 5	∅
1	2, 3	5
2	∅	5
3		
4		
5		



NODE	STRICTLY DOMINATES	DOMINANCE FRONTIER
0	1, 2, 3, 4, 5	∅
1	2, 3	5
2	∅	5
3	∅	5
4		
5		

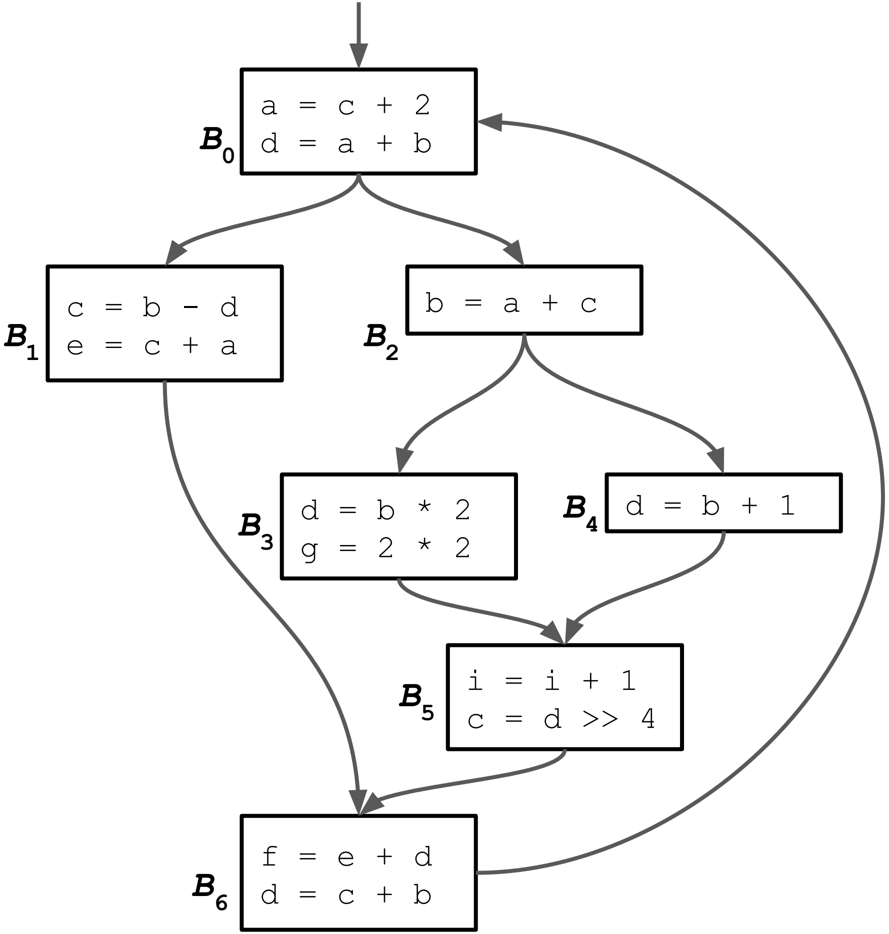


NODE	STRICTLY DOMINATES	DOMINANCE FRONTIER
0	1, 2, 3, 4, 5	∅
1	2, 3	5
2	∅	5
3	∅	5
4	∅	4, 5
5		

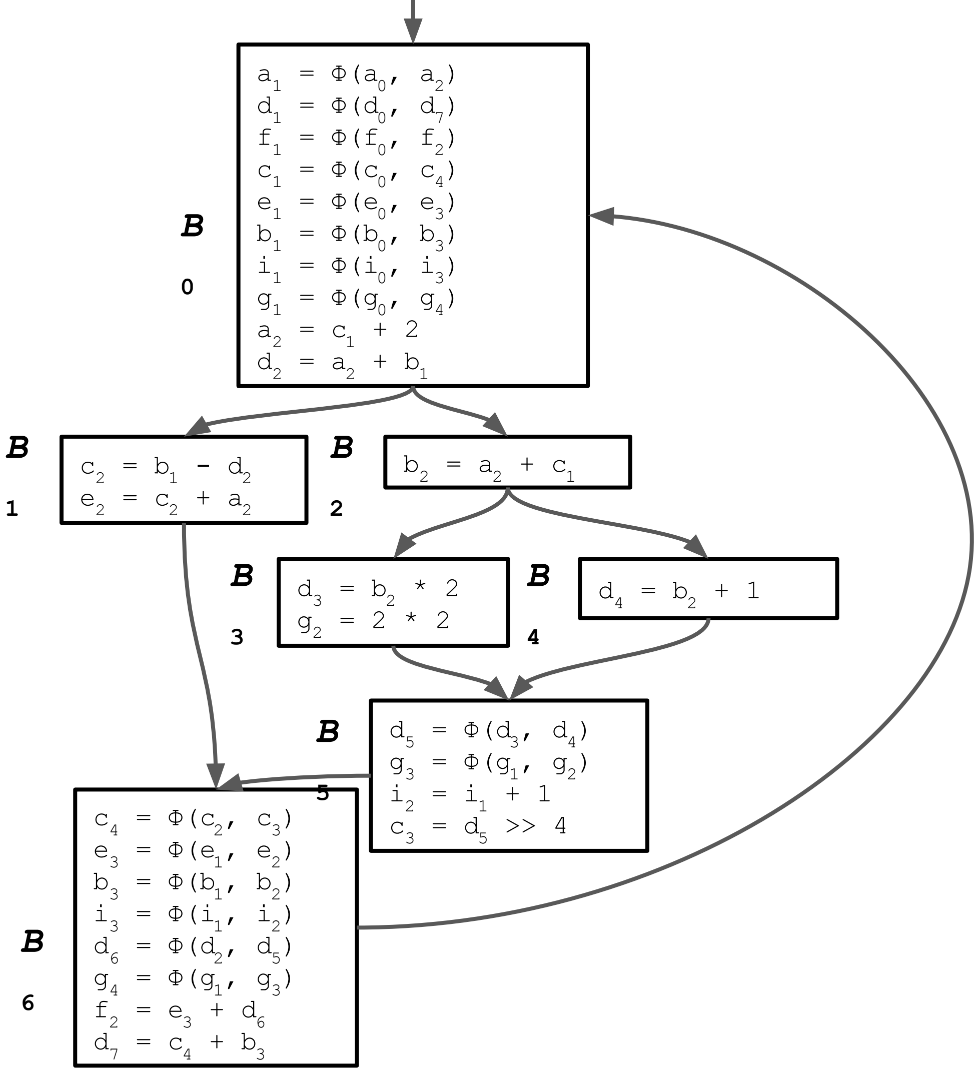


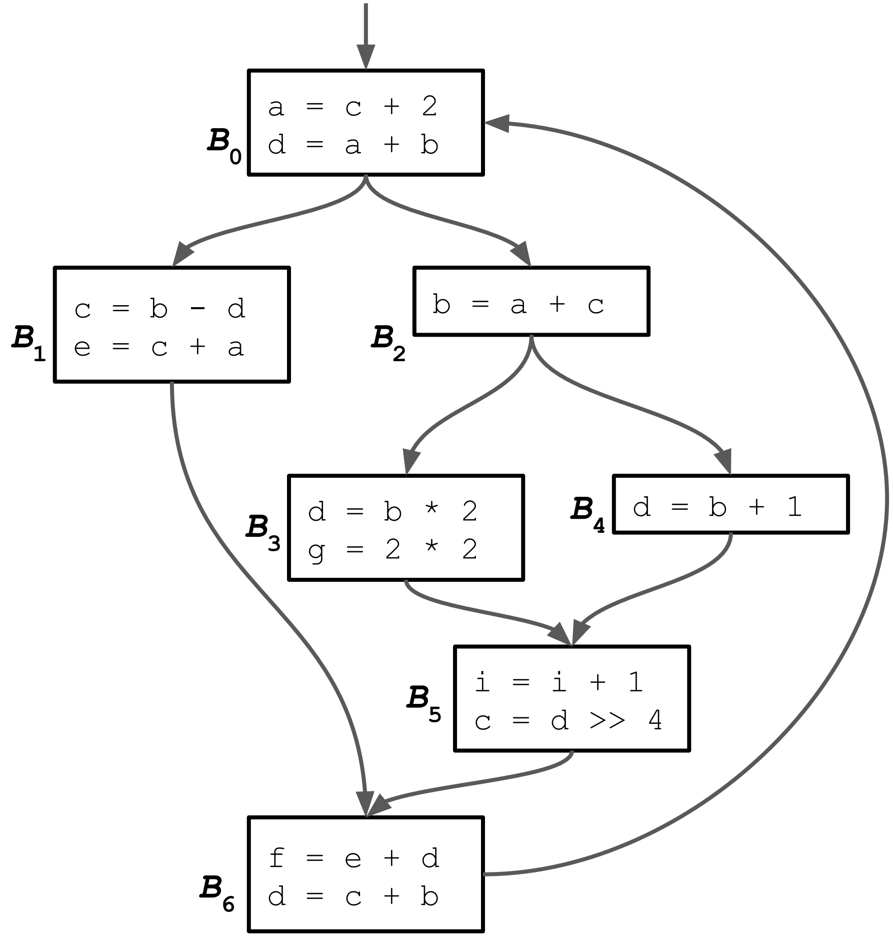
NODE	STRICTLY DOMINATES	DOMINANCE FRONTIER
0	1, 2, 3, 4, 5	∅
1	2, 3	5
2	∅	5
3	∅	5
4	∅	4, 5
5	∅	∅

Problem 3


B0
a = c + 2
d = a + b
B1
c = b - d
e = c + a
B2
b = a + c
B3
d = b * 2
g = 2 * 2
B4
d = b + 1
B5
i = i + 1
c = d >> 4
B6
f = e + d
d = c + b

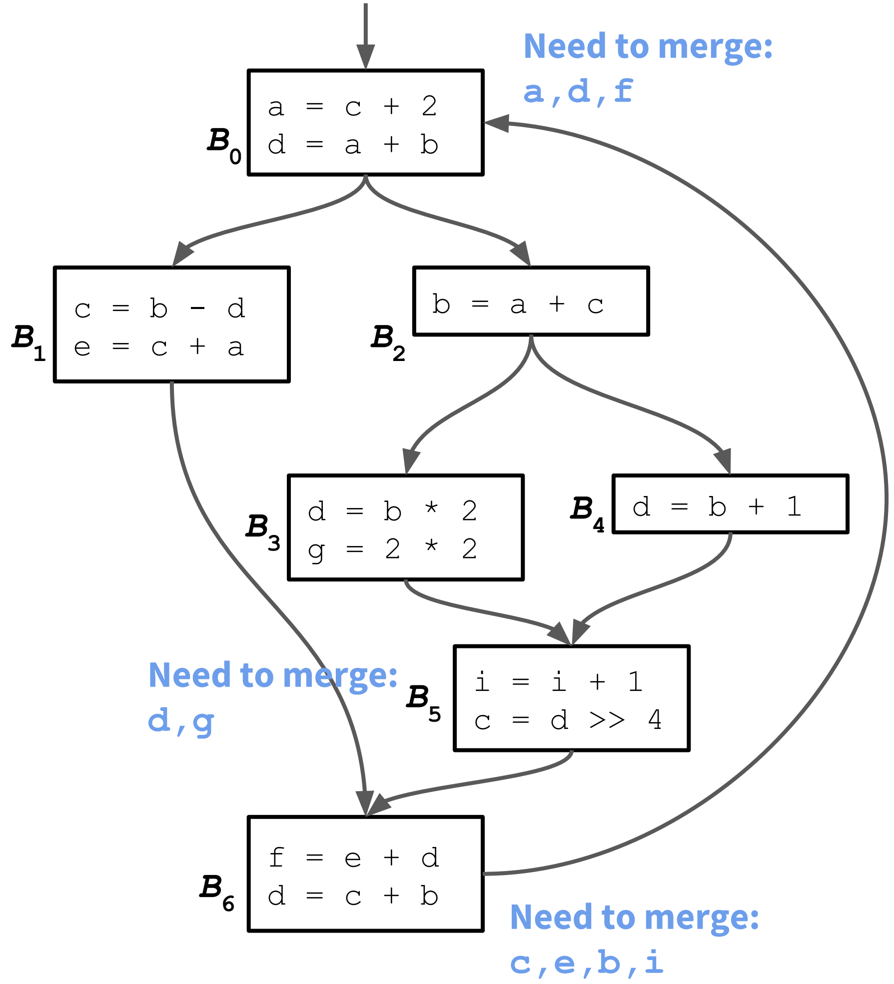
Solution


B
0
a1 = Φ(a0, a2)
d1 = Φ(d0, d7)
f1 = Φ(f0, f2)
c1 = Φ(c0, c4)
e1 = Φ(e0, e3)
b1 = Φ(b0, b3)
i1 = Φ(i0, i3)
g1 = Φ(g0, g4)
a2 = c1 + 2
d2 = a2 + b1
B
1
c2 = b1 - d2
e2 = c2 + a2
B
2
b2 = a2 + c1
B
3
d3 = b2 * 2
g2 = 2 * 2
B
4
d4 = b2 + 1
B
5
d5 = Φ(d3, d4)
g3 = Φ(g1, g2)
i2 = i1 + 1
c3 = d5 >> 4
B
6
c4 = Φ(c2, c3)
e3 = Φ(e1, e2)
b3 = Φ(b1, b2)
i3 = Φ(i1, i2)
d6 = Φ(d2, d5)
g4 = Φ(g1, g3)
f2 = e3 + d6
d7 = c4 + b3



Step 1: Compute Dominance Frontiers

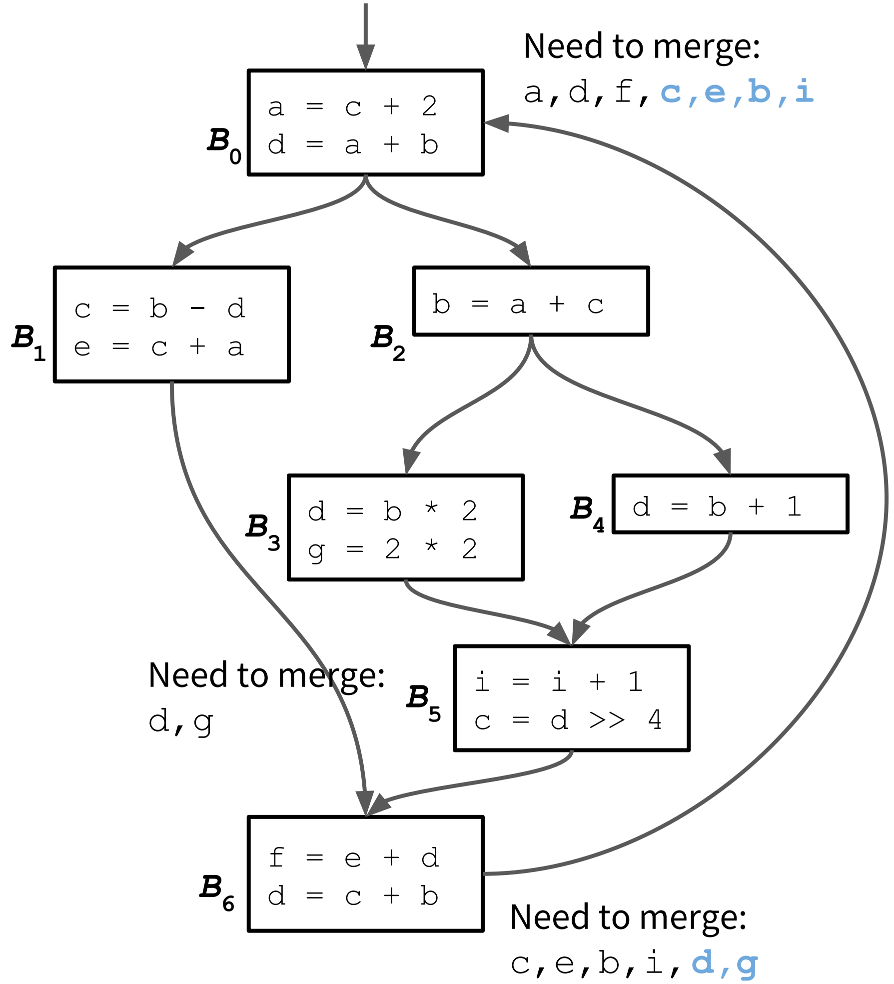
NODE	STRICTLY DOMINATES	DOMINANCE FRONTIER
0	1, 2, 3, 4, 5, 6	0
1	∅	6
2	3, 4, 5	6
3	∅	5
4	∅	5
5	∅	6
6	∅	0



Step 2: Determine Necessary Merges

Each node in the dominance frontier of node X will merge definitions created in node X

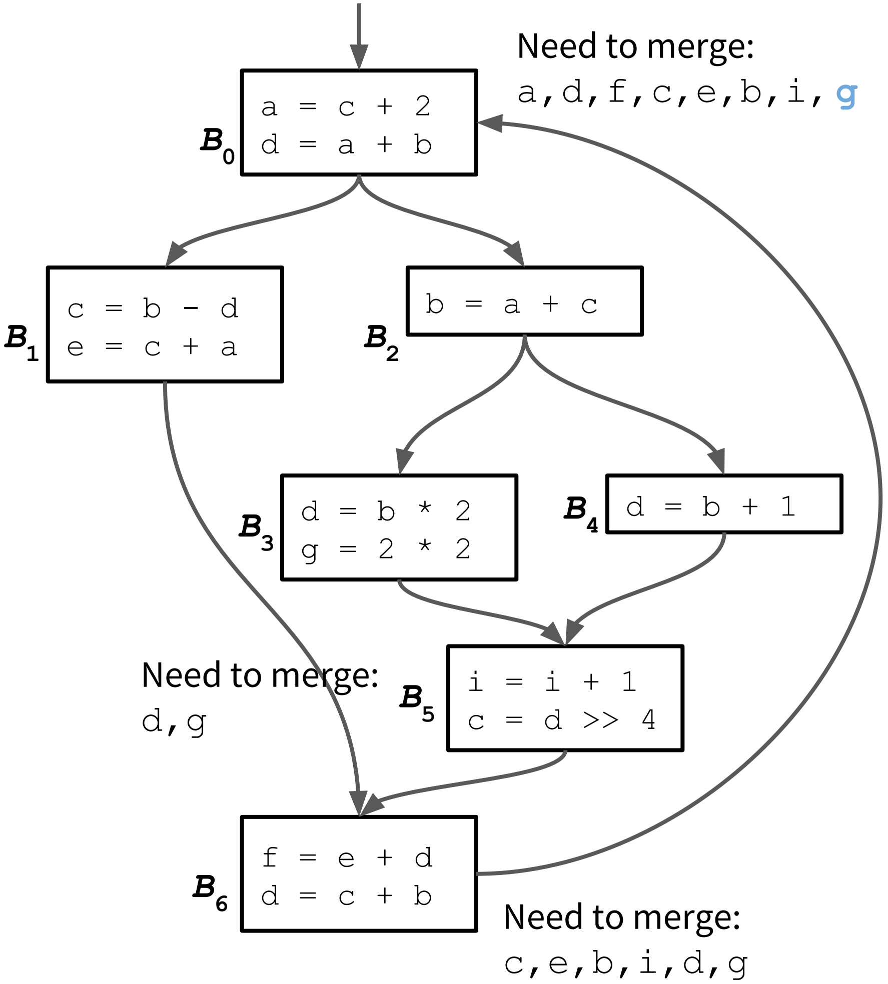
NODE	STRICTLY DOMINATES	DOMINANCE FRONTIER
0	1, 2, 3, 4, 5, 6	0
1	∅	6
2	3, 4, 5	6
3	∅	5
4	∅	5
5	∅	6
6	∅	0



Step 3: Continue Computing Merges

Each merge will create a new definition, and that definition may need to be merged again -- continue until there are no changes

NODE	STRICTLY DOMINATES	DOMINANCE FRONTIER
0	1, 2, 3, 4, 5, 6	0
1	∅	6
2	3, 4, 5	6
3	∅	5
4	∅	5
5	∅	6
6	∅	0



Step 3: Continue Computing Merges

Each merge will create a new definition, and that definition may need to be merged again -- continue until there are no changes

NODE	STRICTLY DOMINATES	DOMINANCE FRONTIER
0	1, 2, 3, 4, 5, 6	0
1	∅	6
2	3, 4, 5	6
3	∅	5
4	∅	5
5	∅	6
6	∅	0

Step 4: Write SSA Definitions

Merges go first, and each successive definition of a variable should increment its index by 1.

B_0

```
a = c + 2
d = a + b
```

Need to merge:

a,d,f,c,e,b,i,g

⇒

B_0

$a_1 = \Phi(a_0, a_2)$
$d_1 = \Phi(d_0, d_7)$
$f_1 = \Phi(f_0, f_2)$
$c_1 = \Phi(c_0, c_4)$
$e_1 = \Phi(e_0, e_3)$
$b_1 = \Phi(b_0, b_3)$
$i_1 = \Phi(i_0, i_3)$
$g_1 = \Phi(g_0, g_4)$
$a_2 = c_1 + 2$
$d_2 = a_2 + b_1$

Step 4: Write SSA Definitions

Merges go first, and each successive definition of a variable should increment its index by 1.

B_1

```
c = b - d
e = c + a
```

Nothing to merge

B_1

$c_2 = b_1 - d_2$

$e_2 = c_2 + a_2$

Step 4: Write SSA Definitions

Merges go first, and each successive definition of a variable should increment its index by 1.

B_2: `b = a + c`

⇒

B_2: $b_2 = a_2 + c_1$

Nothing to merge

Step 4: Write SSA Definitions

Merges go first, and each successive definition of a variable should increment its index by 1.

B_3

```
d = b * 2
g = 2 * 2
```

Nothing to merge

⇒

B_3

$d_3 = b_2 * 2$

$g_2 = 2 * 2$

Step 4: Write SSA Definitions

Merges go first, and each successive definition of a variable should increment its index by 1.

B_4	`d = b + 1`	⇒	B_4: $d_4 = b_2 + 1$

Nothing to merge

Step 4: Write SSA Definitions

Merges go first, and each successive definition of a variable should increment its index by 1.

B_5

```
i = i + 1
c = d >> 4
```

Need to merge:

d, g

⇒

B_5

$$
\begin{aligned}
d_5 &= \Phi(d_3,\ d_4) \\
g_3 &= \Phi(g_1,\ g_2) \\
i_2 &= i_1 + 1 \\
c_3 &= d_5 \gg 4
\end{aligned}
$$

Step 4: Write SSA Definitions

Merges go first, and each successive definition of a variable should increment its index by 1.

B_6

```
f = e + d
d = c + b
```

Need to merge:

c,e,b,i,d,g

⇒

B_6

$$
\begin{aligned}
c_4 &= \Phi(c_2,\ c_3)\\
e_3 &= \Phi(e_1,\ e_2)\\
b_3 &= \Phi(b_1,\ b_2)\\
i_3 &= \Phi(i_1,\ i_2)\\
d_6 &= \Phi(d_2,\ d_5)\\
g_4 &= \Phi(g_1,\ g_3)\\
f_2 &= e_3 + d_6\\
d_7 &= c_4 + b_3
\end{aligned}
$$

Thanks for a Great Quarter!

- **The 401 18sp Staff :)**