Adventures in

Dataflow Analysis

CSE 401 Section 9-ish
Aaron Johnston & Nate Yazdani

Announcements

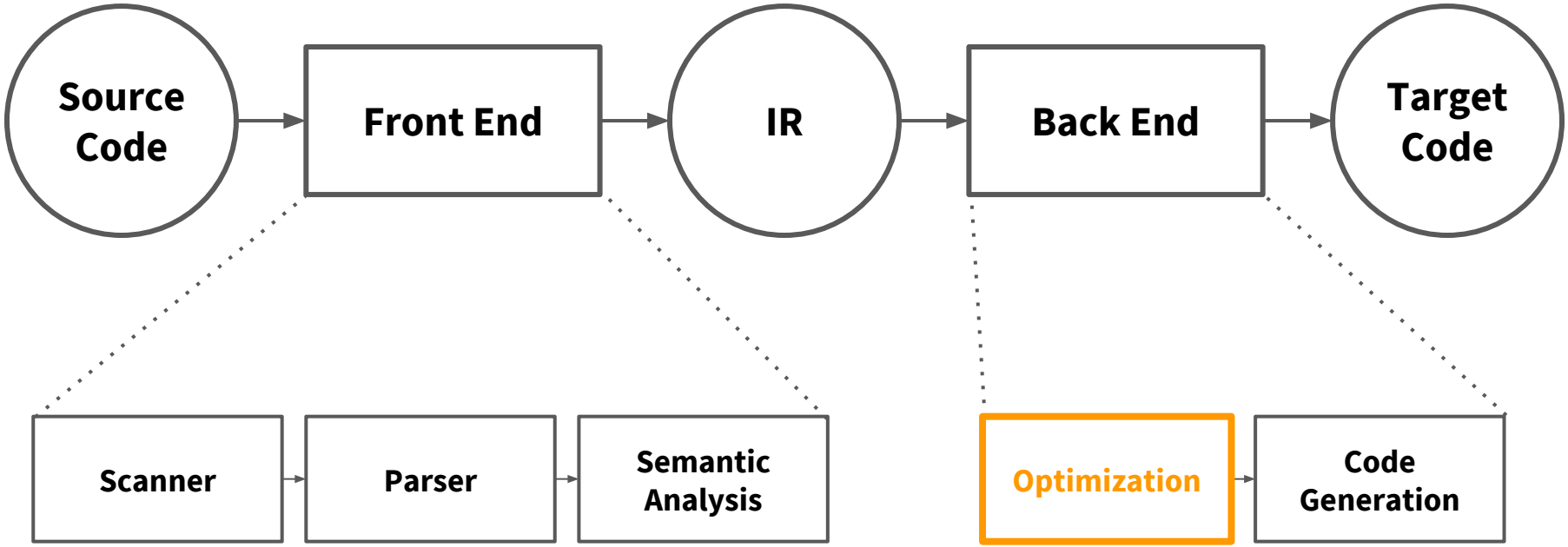
- Compiler Additions due next Thursday, 5/31
    - Involves revisiting all parts of the compiler

Announcements

- Compiler Additions due next Thursday, 5/31
  - Involves revisiting all parts of the compiler

- Final Report due the following Saturday, 6/2
  - Ideally, also involves revisiting all parts of the compiler

Review of Optimizations



Review of Optimizations

Peephole

Local

Intraprocedural / Global

Interprocedural

Review of Optimizations

Peephole	A few Instructions
Local	
Intraprocedural / Global	
Interprocedural	

Review of Optimizations

Peephole	A few Instructions
Local	A Basic Block
Intraprocedural / Global	
Interprocedural	

Review of Optimizations

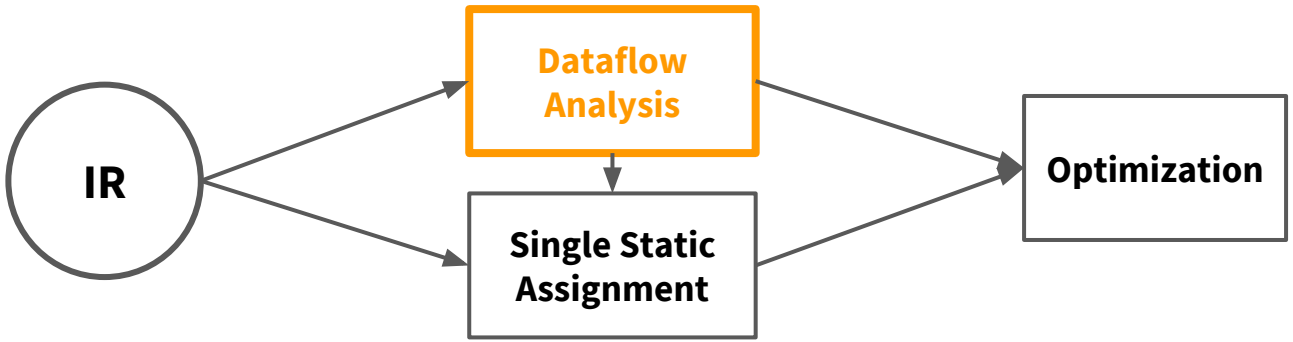
Peephole	A few Instructions
Local	A Basic Block
Intraprocedural / Global	A Function/Method
Interprocedural	

Review of Optimizations

Peephole	A few Instructions
Local	A Basic Block
Intraprocedural / Global	A Function/Method
Interprocedural	A Program

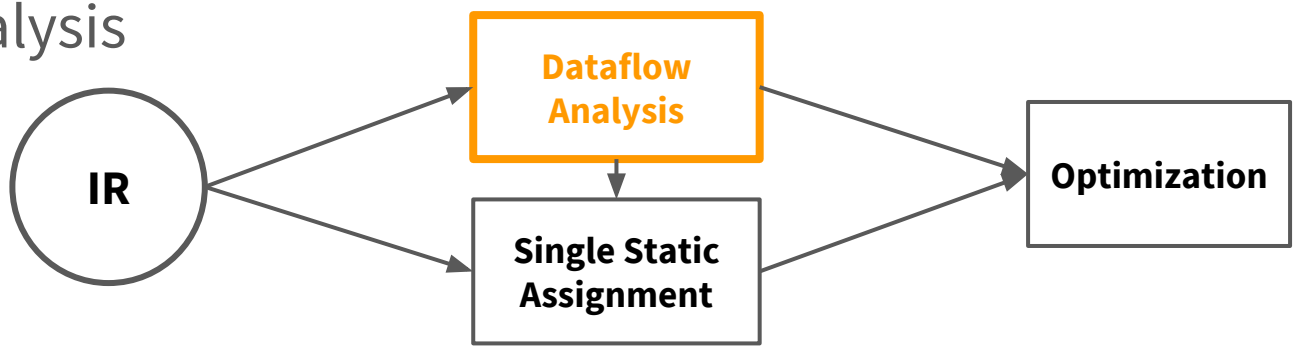
Overview of Dataflow Analysis

- A framework for exposing properties about programs
- Operates using sets of “facts”



Overview of Dataflow Analysis

- A framework for exposing properties about programs
- Operates using sets of “facts”
- Just the initial discovery phase
  - Changes can then be made to optimize based on the analysis



Overview of Dataflow Analysis

- Basic Set Definitions for a Basic Block *b*:
  - `IN(`*b*`)`: facts true on entry to *b*
  - `OUT(`*b*`)`: facts true on exit from *b*
  - `GEN(`*b*`)`: facts created (and not killed) in *b*
  - `KILL(`*b*`)`: facts killed in *b*

1 (a) & (b)

Equations for Reaching Definitions

- Sets:
  - DEFOUT(b): set of definitions in b that reach the end of b (i.e., not subsequently redefined in b)
  - SURVIVED(b): set of all definitions not obscured by a definition in b
  - REACHES(b): set of definitions that reach b
- Equations:
  - $\text{REACHES}(b) = \bigcup_{p \in \text{preds}(b)} \text{DEFOUT}(p) \cup (\text{REACHES}(p) \cap \text{SURVIVED}(p))$

Overview of Dataflow Analysis

- Basic Set Definitions for a Basic Block *b*:
  - IN(*b*): facts true on entry to *b*
  - OUT(*b*): facts true on exit from *b*
  - GEN(*b*): facts created (and not killed) in *b*
  - KILL(*b*): facts killed in *b*

$$OUT(b) = GEN(b) \cup (IN(b) - KILL(b))$$

1 (c) & (d)

```
L0:  a = 0
L1:  b = a + 1
L2:  c = c + b
L3:  a = b * 2
L4:  if a < N goto L1
L5:  return c
```

Block	GEN	KILL	IN (1)	OUT (1)	IN (2)	OUT (2)
L0	L0					
L1	L1					
L2	L2					
L3	L3					
L4						
L5						

```
L0:  a = 0
L1:  b = a + 1
L2:  c = c + b
L3:  a = b * 2
L4:  if a < N goto L1
L5:  return c
```

Block	GEN	KILL	IN (1)	OUT (1)	IN (2)	OUT (2)
L0	L0	L3				
L1	L1					
L2	L2					
L3	L3	L0				
L4						
L5						

```
L0:  a = 0
L1:  b = a + 1
L2:  c = c + b
L3:  a = b * 2
L4:  if a < N goto L1
L5:  return c
```

Block	GEN	KILL	IN (1)	OUT (1)	IN (2)	OUT (2)
L0	L0	L3				
L1	L1		L0			
L2	L2		L0, L1			
L3	L3	L0	L0, L1, L2			
L4			L1, L2, L3			
L5			L1, L2, L3			

```
L0:  a = 0
L1:  b = a + 1
L2:  c = c + b
L3:  a = b * 2
L4:  if a < N goto L1
L5:  return c
```

Block	GEN	KILL	IN (1)	OUT (1)	IN (2)	OUT (2)
L0	L0	L3		L0		
L1	L1		L0	L0, L1		
L2	L2		L0, L1	L0, L1, L2		
L3	L3	L0	L0, L1, L2	L1, L2, L3		
L4			L1, L2, L3	L1, L2, L3		
L5			L1, L2, L3	L1, L2, L3		

```
L0:  a = 0
L1:  b = a + 1
L2:  c = c + b
L3:  a = b * 2
L4:  if a < N goto L1
L5:  return c
```

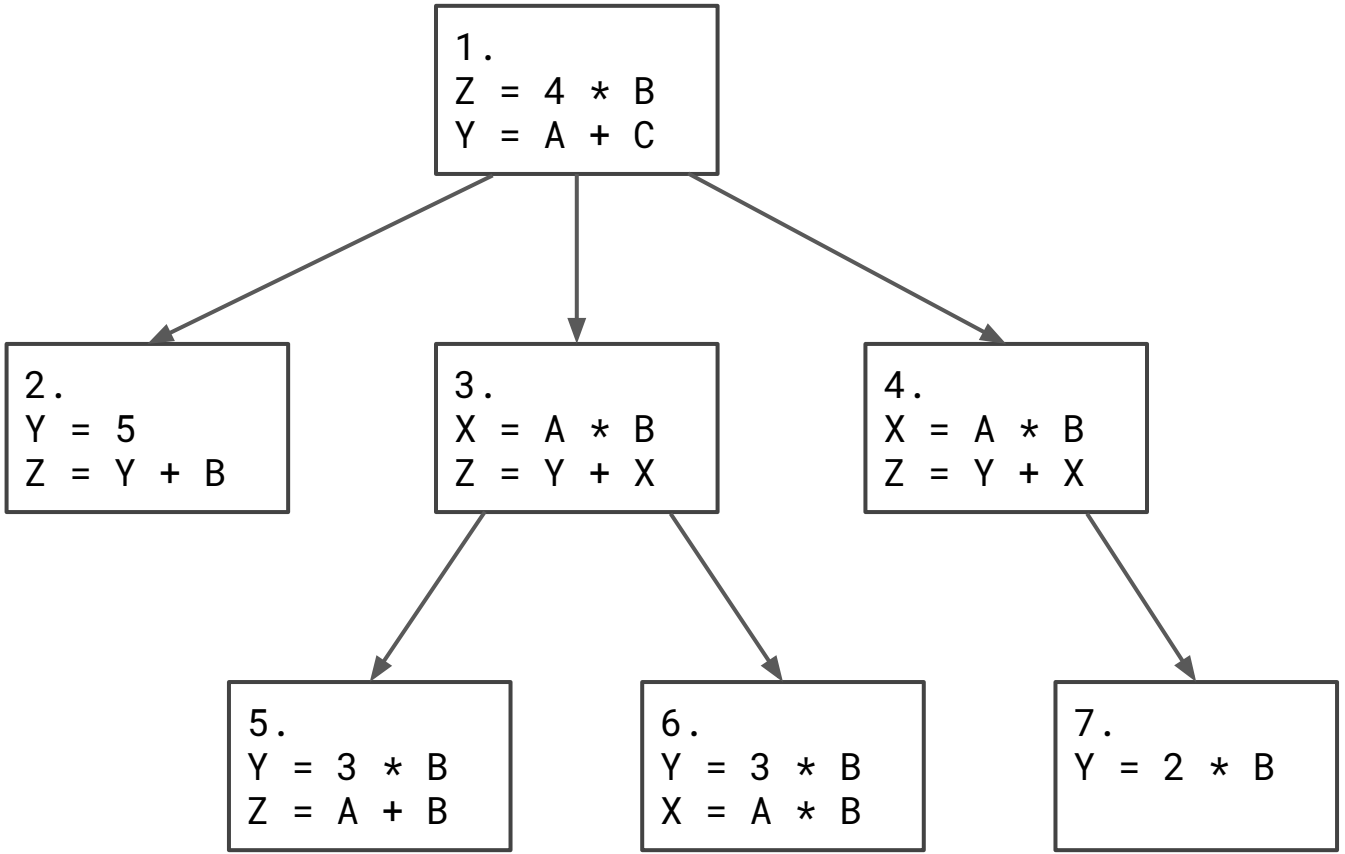
Block	GEN	KILL	IN (1)	OUT (1)	IN (2)	OUT (2)
L0	L0	L3		L0		L0
L1	L1		L0	L0, L1	L0, L1, L2, L3	L0, L1, L2, L3
L2	L2		L0, L1	L0, L1, L2	L0, L1, L2, L3	L0, L1, L2, L3
L3	L3	L0	L0, L1, L2	L1, L2, L3	L0, L1, L2, L3	L1, L2, L3
L4			L1, L2, L3	L1, L2, L3	L1, L2, L3	L1, L2, L3
L5			L1, L2, L3	L1, L2, L3	L1, L2, L3	L1, L2, L3

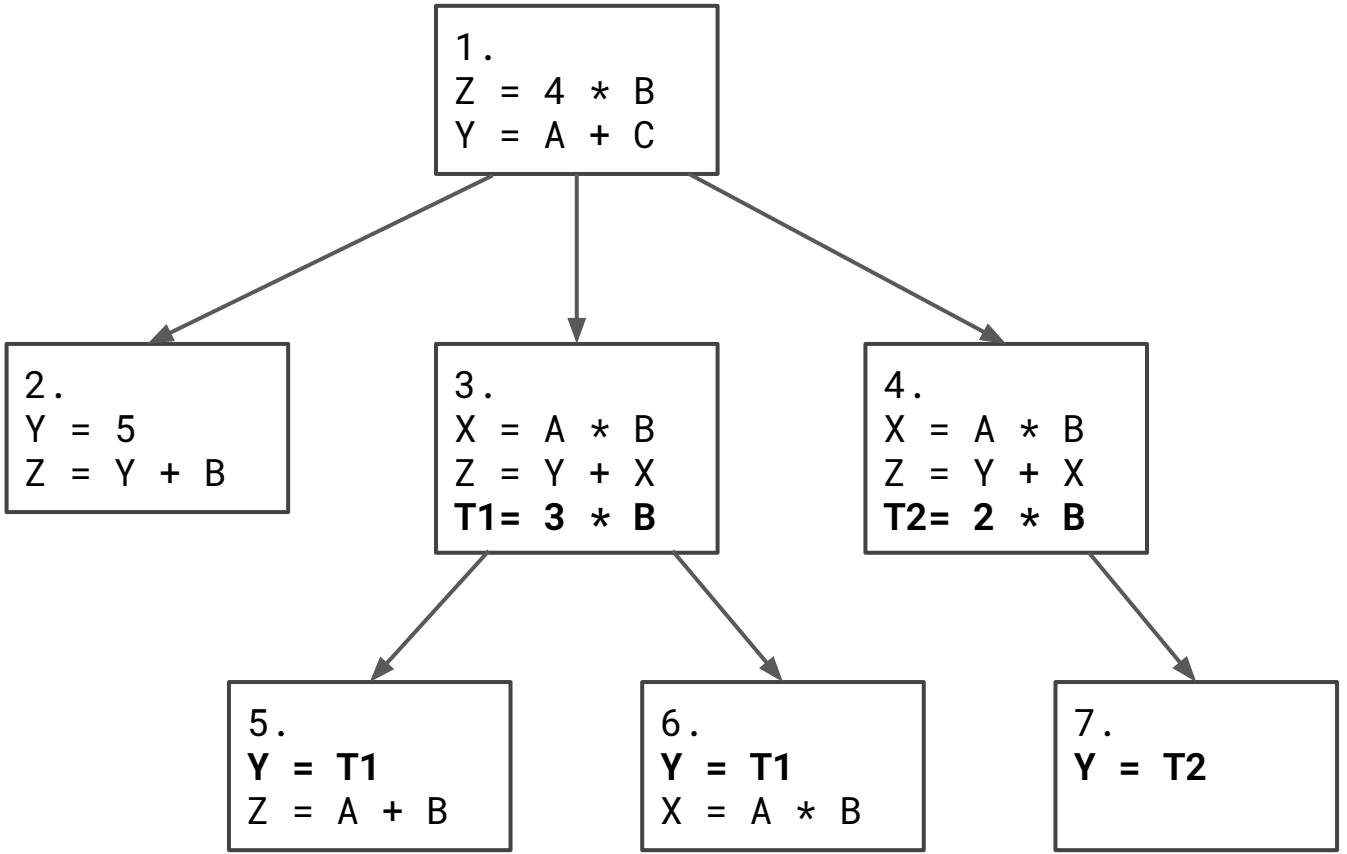
```
L0:  a = 0
L1:  b = a + 1
L2:  c = c + b
L3:  a = b * 2
L4:  if a < N goto L1
L5:  return c
```

Convergence!

Block	GEN	KILL	IN (1)	OUT (1)	IN (2)	OUT (2)
L0	L0	L3		L0		L0
L1	L1		L0	L0, L1	L0, L1, L2, L3	L0, L1, L2, L3
L2	L2		L0, L1	L0, L1, L2	L0, L1, L2, L3	L0, L1, L2, L3
L3	L3	L0	L0, L1, L2	L1, L2, L3	L0, L1, L2, L3	L1, L2, L3
L4			L1, L2, L3	L1, L2, L3	L1, L2, L3	L1, L2, L3
L5			L1, L2, L3	L1, L2, L3	L1, L2, L3	L1, L2, L3

2 (a) & (b)


1.
Z = 4 * B
Y = A + C
2.
Y = 5
Z = Y + B
3.
X = A * B
Z = Y + X
4.
X = A * B
Z = Y + X
5.
Y = 3 * B
Z = A + B
6.
Y = 3 * B
X = A * B
7.
Y = 2 * B


1.
Z = 4 * B
Y = A + C
2.
Y = 5
Z = Y + B
3.
X = A * B
Z = Y + X
T1= 3 * B
4.
X = A * B
Z = Y + X
T2= 2 * B
5.
Y = T1
Z = A + B
6.
Y = T1
X = A * B
7.
Y = T2