

Lecture 4: Data-Flow Analysis & Abstract Interpretation Framework

17-355/17-655/17-819: Program Analysis

Rohan Padhye

September 9, 2025

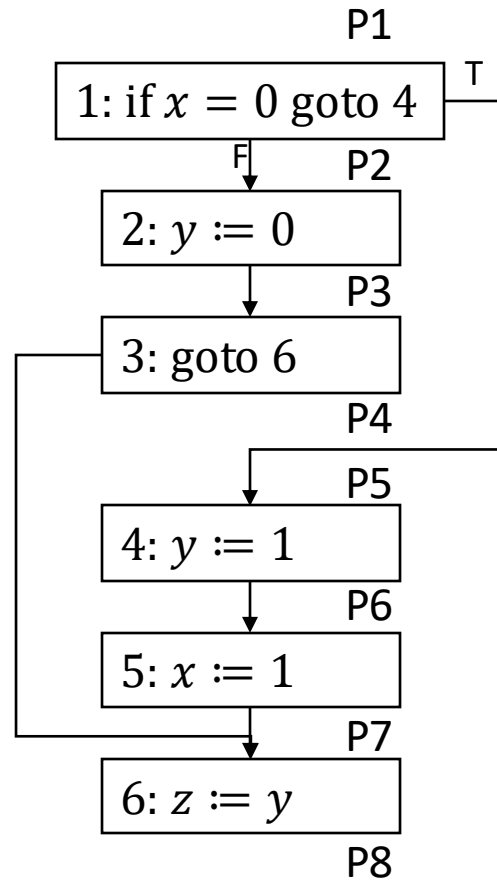
* Course materials developed with Jonathan Aldrich Claire Le Goues

Review: Zero Analysis with Branching

```

1 :  if  $x = 0$  goto 4
2 :   $y := 0$ 
3 :  goto 6
4 :   $y := 1$ 
5 :   $x := 1$ 
6 :   $z := y$ 

```



	x	y	z
P1	?	?	?
P2	Z_T, N_F	?	?
P3	N	Z	?
P4	N	Z	?
P5	Z	?	?
P6	Z	N	?
P7	N	T	?
P8	N	T	T

Partial Order & Join on set L

$l_1 \sqsubseteq l_2$: l_1 is at least as precise as l_2

reflexive: $\forall l : l \sqsubseteq l$

transitive: $\forall l_1, l_2, l_3 : l_1 \sqsubseteq l_2 \wedge l_2 \sqsubseteq l_3 \Rightarrow l_1 \sqsubseteq l_3$

anti-symmetric: $\forall l_1, l_2 : l_1 \sqsubseteq l_2 \wedge l_2 \sqsubseteq l_1 \Rightarrow l_1 = l_2$

$l_1 \sqcup l_2$: **join** or *least-upper-bound*... “most precise generalization”

L is a *join-semilattice* iff: $l_1 \sqcup l_2$ always exists and is unique $\forall l_1, l_2 \in L$

\top (“top”) is the maximal element

Lattice for Zero Analysis

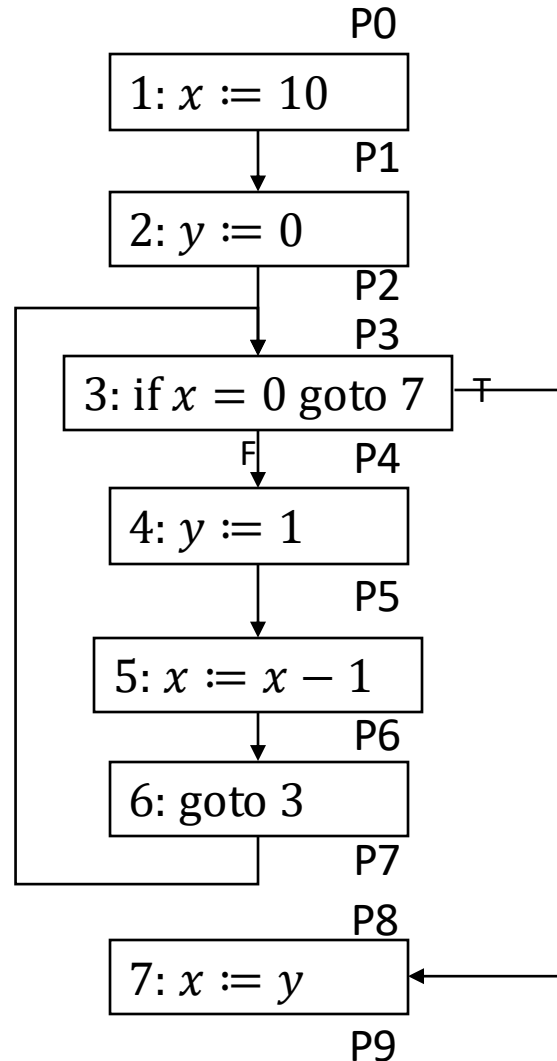
What would this look like?

Data-Flow Analysis

- a lattice (L, \sqsubseteq)
- an abstraction function α
- a flow function f
- initial dataflow analysis assumptions, σ_0

Example of Zero Analysis: Looping Code

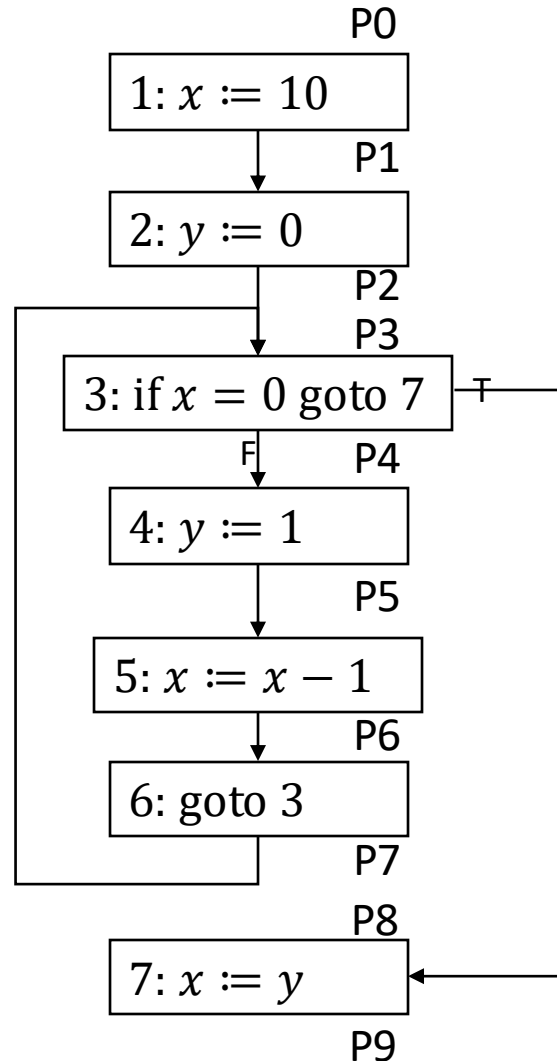
```
1 :  x := 10
2 :  y := 0
3 :  if x = 0 goto 7
4 :  y := 1
5 :  x := x - 1
6 :  goto 3
7 :  x := y
```



Example of Zero Analysis: Looping Code

```

1 :  x := 10
2 :  y := 0
3 :  if x = 0 goto 7
4 :  y := 1
5 :  x := x - 1
6 :  goto 3
7 :  x := y
    
```

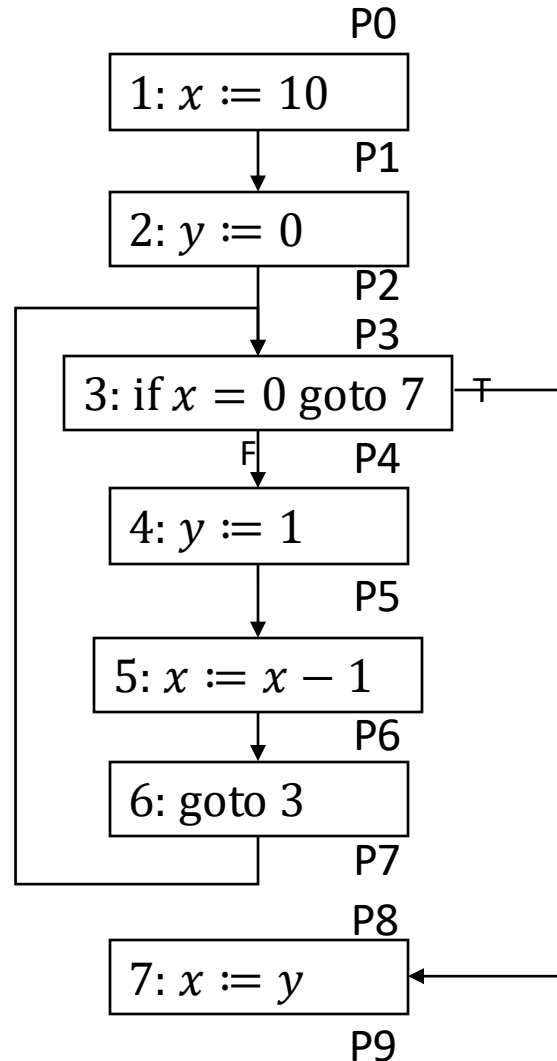


	x	y	
P0	T	T	
P1	N	T	
P2	N	Z	
P3	N	Z	<i>first time through...</i>
P4	N_F	Z	
P5	N	N	
P6	T	N	
P7	T	N	
P8	Z_t	N	<i>first time through...</i>
P9	N	N	<i>first time through...</i>

Example of Zero Analysis: Looping Code

```

1 :  x := 10
2 :  y := 0
3 :  if x = 0 goto 7
4 :  y := 1
5 :  x := x - 1
6 :  goto 3
7 :  x := y
    
```

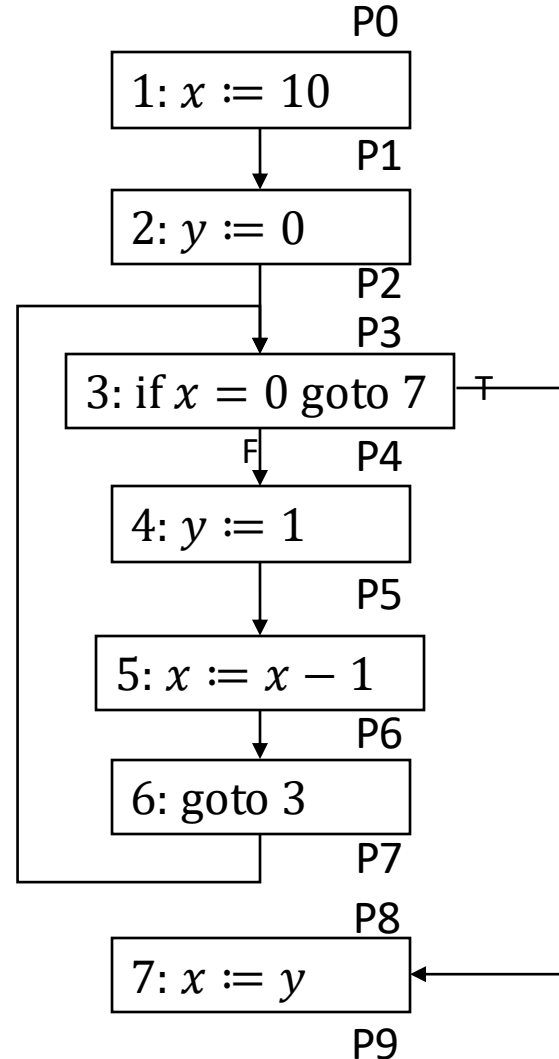


	x	y	
P0	\top	\top	
P1	N	\top	
P2	N	Z	
P3	\top	\top	<i>join</i>
P4	N_F	\top	<i>updated</i>
P5	N	N	<i>already at fixed point</i>
P6	\top	N	<i>already at fixed point</i>
P7	\top	N	<i>already at fixed point</i>
P8	Z_T	\top	<i>updated</i>
P9	\top	\top	<i>updated</i>

Fixed point of Flow Functions

```

1 :  x := 10
2 :  y := 0
3 :  if x = 0 goto 7
4 :  y := 1
5 :  x := x - 1
6 :  goto 3
7 :  x := y
    
```



$$(\sigma_0, \sigma_1, \sigma_2, \dots, \sigma_n) \xrightarrow{f_Z} (\sigma'_0, \sigma'_1, \sigma'_2, \dots, \sigma'_n)$$

$$\sigma'_0 = \sigma_0$$

$$\sigma'_1 = f_Z \llbracket x := 10 \rrbracket (\sigma_0)$$

$$\sigma'_2 = f_Z \llbracket y := 0 \rrbracket (\sigma_1)$$

$$\sigma'_3 = \sigma_2 \sqcup \sigma_7$$

$$\sigma'_4 = f_Z \llbracket \text{if } x = 10 \text{ goto } 7 \rrbracket_F (\sigma_3)$$

$$\vdots$$

$$\sigma'_8 = f_Z \llbracket \text{if } x = 10 \text{ goto } 7 \rrbracket_T (\sigma_3)$$

$$\sigma'_9 = f_Z \llbracket x := y \rrbracket (\sigma_8)$$

Fixed point of Flow Functions

$$(\sigma_0, \sigma_1, \sigma_2, \dots, \sigma_n) \xrightarrow{f_Z} (\sigma'_0, \sigma'_1, \sigma'_2, \dots, \sigma'_n)$$

Fixed point!

$$(\sigma_0, \sigma_1, \sigma_2, \dots, \sigma_n) = f_Z(\sigma_0, \sigma_1, \sigma_2, \dots, \sigma_n)$$

Correctness theorem:

If data-flow analysis is well designed*, then any fixed point of the analysis is sound.

* we will define these properties and prove this theorem in two weeks!

$$\sigma'_0 = \sigma_0$$

$$\sigma'_1 = f_Z[\![x := 10]\!](\sigma_0)$$

$$\sigma'_2 = f_Z[\![y := 0]\!](\sigma_1)$$

$$\sigma'_3 = \sigma_2 \sqcup \sigma_7$$

$$\sigma'_4 = f_Z[\![\text{if } x = 10 \text{ goto } 7]\!]_F(\sigma_3)$$

$$\vdots$$

$$\sigma'_8 = f_Z[\![\text{if } x = 10 \text{ goto } 7]\!]_T(\sigma_3)$$

$$\sigma'_9 = f_Z[\![x := y]\!](\sigma_8)$$

More on joins and lattices

$$(\sigma_0, \sigma_1, \sigma_2, \dots, \sigma_n) \xrightarrow{f_Z} (\sigma'_0, \sigma'_1, \sigma'_2, \dots, \sigma'_n)$$

Hold up! How do you

$$\sigma'_0 = \sigma_0$$

$$\sigma'_1 = f_Z \llbracket x := 10 \rrbracket (\sigma_0)$$

$$\sigma'_2 = f_Z \llbracket y := 0 \rrbracket (\sigma_1)$$

$$\sigma'_3 = \sigma_2 \sqcup \sigma_7$$

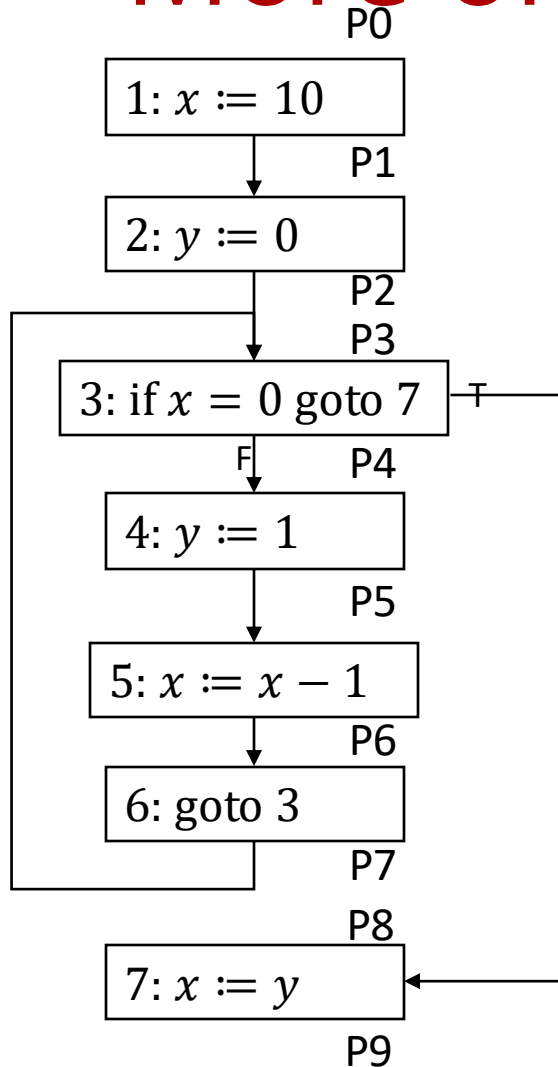
$$\sigma'_4 = f_Z \llbracket \text{if } x = 10 \text{ goto } 7 \rrbracket_F (\sigma_3)$$

$$\vdots$$

$$\sigma'_8 = f_Z \llbracket \text{if } x = 10 \text{ goto } 7 \rrbracket_T (\sigma_3)$$

$$\sigma'_9 = f_Z \llbracket x := y \rrbracket (\sigma_8)$$

More on joins and lattices



	x	y	
P0	T	T	
P1	N	T	
P2	N	Z	
P3	N	Z	<i>first time through...</i>
P4			
P5			$\sigma'_3 = \sigma_2$
P6			
P7			What should be the initial value for σ_7 ???
P8			
P9			

$$\sigma'_3 = \sigma_2 \sqcup \sigma_7$$

What should be the initial value for σ_7 ????

More on joins and lattices

Enter: \perp (“bottom”)

What would the **complete lattice**
for Zero Analysis look like?

for all $l \in L$:

$$\perp \sqsubseteq l \qquad l \sqsubseteq \top$$

$$\perp \sqcup l = l \qquad l \sqcup \top = \top$$

A lattice with both \perp and \top defined is called a ***Complete Lattice***

More on joins and lattices

$\sigma: Var \rightarrow L$ where $L = \{Z, N, \perp, \top\}$ and $Var = \{x, y\}$

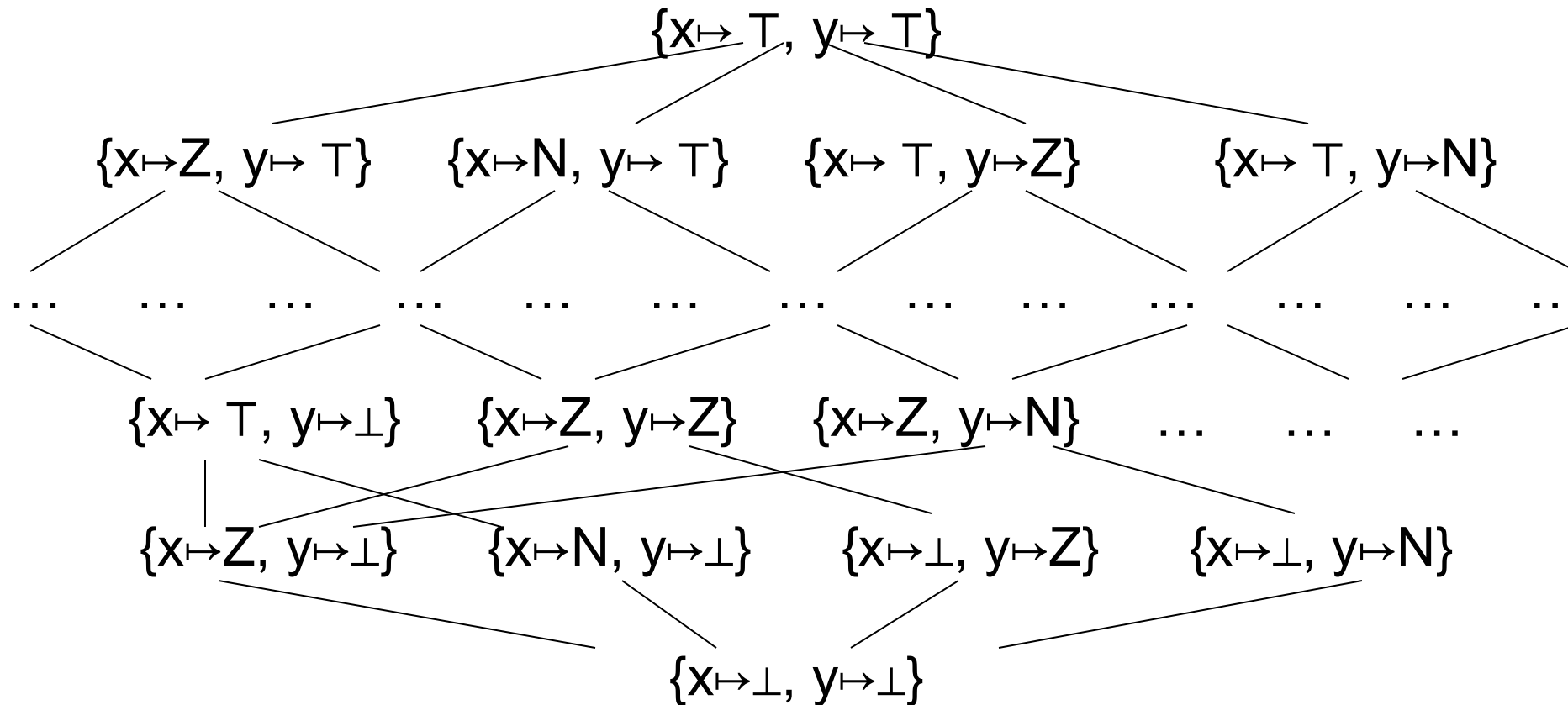
$$\sigma_1 \sqcup \sigma_2 = \{ x \mapsto \sigma_1(x) \sqcup \sigma_2(x), \quad y \mapsto \sigma_1(y) \sqcup \sigma_2(y) \}$$

Exercise: Define lifted \sqsubseteq in terms of ordering on L

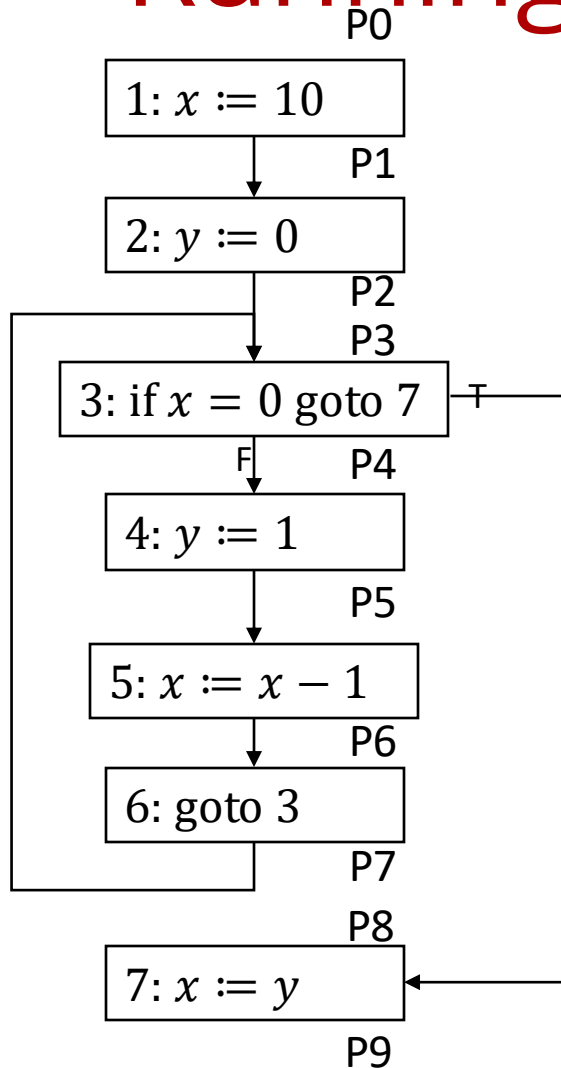
$$\sigma_1 \sqsubseteq \sigma_2 = ???$$

More on joins and lattices

Lifting a complete lattice gives another complete lattice

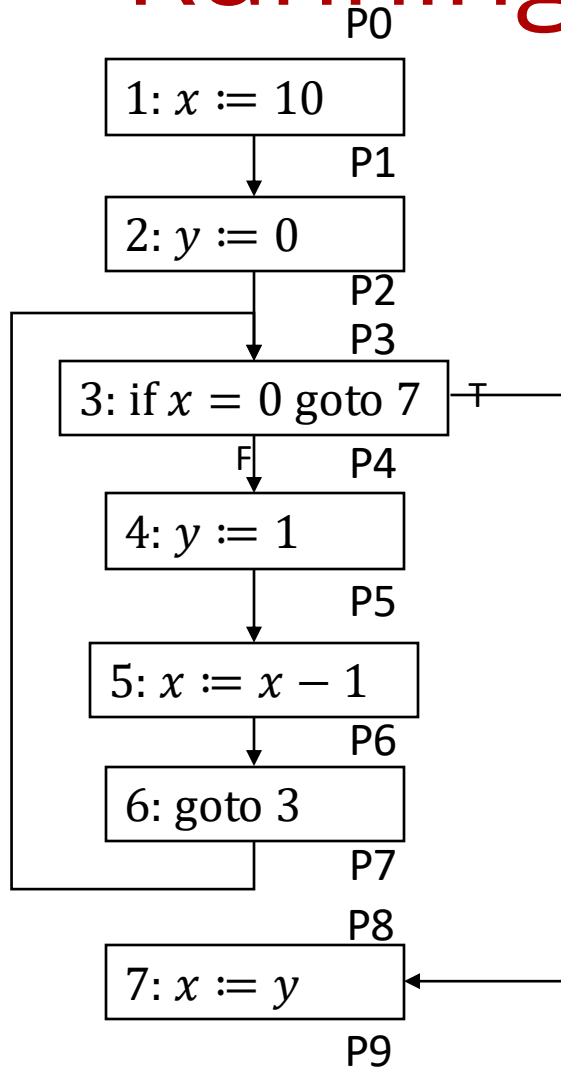


Running a Data Flow Analysis



	x	y
P0	\top	\top
P1	\perp	\perp
P2	\perp	\perp
P3	\perp	\perp
P4	\perp	\perp
P5	\perp	\perp
P6	\perp	\perp
P7	\perp	\perp
P8	\perp	\perp
P9	\perp	\perp

Running a Data Flow Analysis

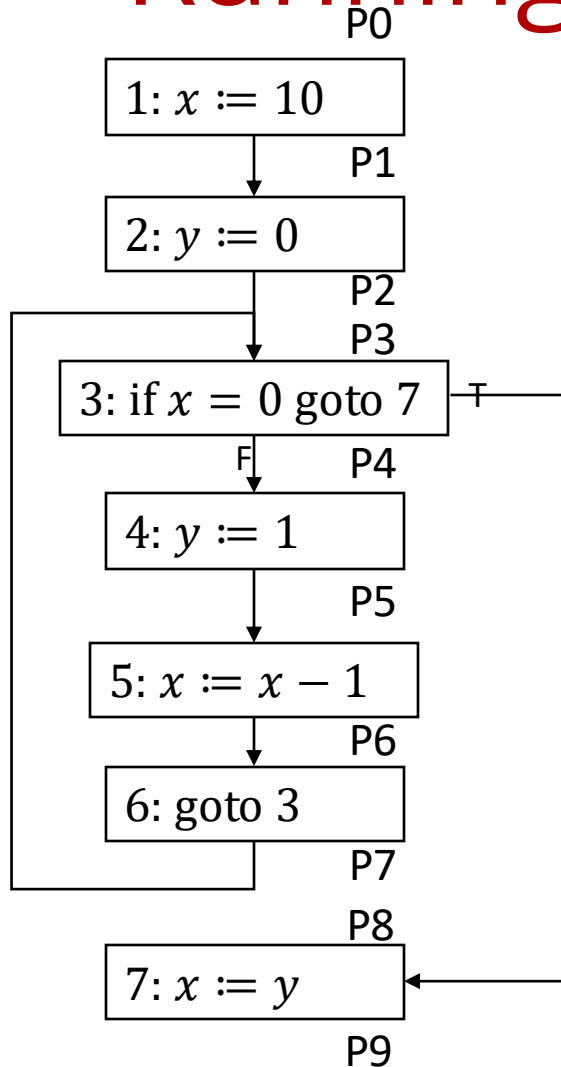


	x	y
P0	\top	\top
P1	N	\top
P2	N	Z
P3	N	Z
P4	\perp	\perp
P5	\perp	\perp
P6	\perp	\perp
P7	\perp	\perp
P8	\perp	\perp
P9	\perp	\perp

first time through...

$$\sigma'_3 = \sigma_2 \sqcup \sigma_7$$

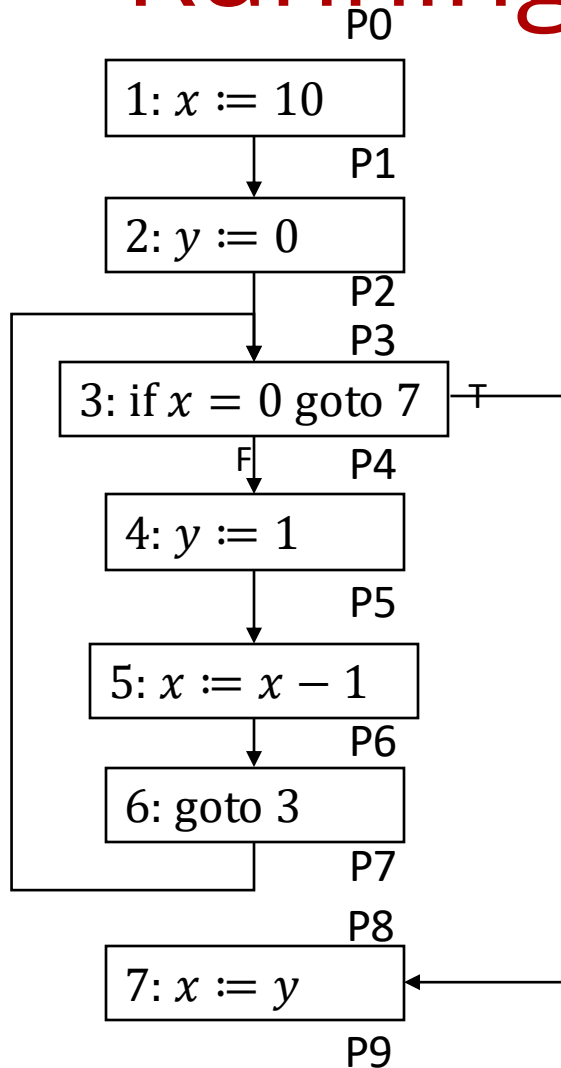
Running a Data Flow Analysis



	x	y	
P0	T	T	
P1	N	T	
P2	N	Z	
P3	N	Z	<i>first time through...</i>
P4	N_F	Z	
P5	N	N	$\sigma'_3 = \sigma_2$
P6	T	N	
P7	T	N	
P8	Z_t	N	<i>first time through...</i>
P9	N	N	<i>first time through...</i>

$$\sigma'_3 = \sigma_2 \sqcup \sigma_7$$

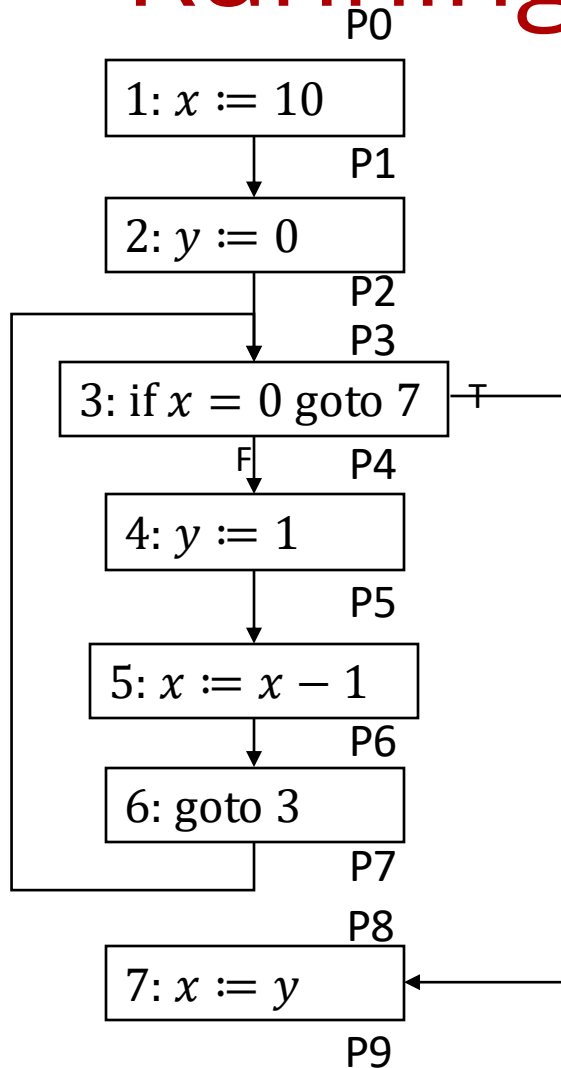
Running a Data Flow Analysis



	x	y	
P0	T	T	
P1	N	T	
P2	N	Z	
P3	T	T	<i>join</i>
P4	N_F	Z	
P5	N	N	
P6	T	N	
P7	T	N	
P8	Z_t	N	<i>first time through...</i>
P9	N	N	<i>first time through...</i>

$$\sigma'_3 = \sigma_2 \sqcup \sigma_7$$

Running a Data Flow Analysis



	x	y	
P0	T	T	
P1	N	T	
P2	N	Z	
P3	T	T	<i>join</i>
P4	N_F	T	<i>updated</i>
P5	N	N	<i>already at fixed point</i>
P6	T	N	<i>already at fixed point</i>
P7	T	N	<i>already at fixed point</i>
P8	Z_T	T	<i>updated</i>
P9	T	T	<i>updated</i>

What's the Algorithm?

Analysis Execution Strategy

```
for Node n in cfg
    input[n] =  $\perp$ 
input[0] = initialDataflowInformation

while not at fixed point
    pick a node n in program
    output = flow(n, input[n])
    for Node j in successors(n)
        input[j] = input[j]  $\sqcup$  output
```

Kildall's Algorithm

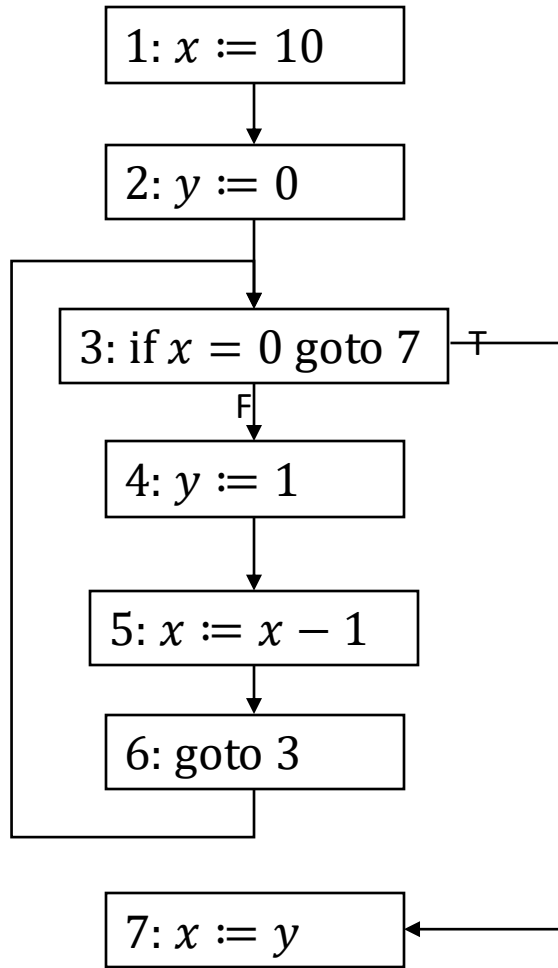
```
worklist =  $\emptyset$ 
for Node n in cfg
    input[n] = output[n] =  $\perp$ 
    add n to worklist
input[0] = initialDataflowInformation

while worklist is not empty
    take a Node n off the worklist
    output[n] = flow(n, input[n])
    for Node j in succs(n)
        newInput = input[j]  $\sqcup$  output[n]
        if newInput  $\neq$  input[j]
            input[j] = newInput
            add j to worklist
```

What order to process worklist nodes in?

- Random? Queue? Stack?
- Any order is valid (!!)
- Some orders are better in practice
 - Topological sorts are nice
 - Explore loops inside out
 - Reverse postorder!

Exercise: Apply Kildall's Worklist Algorithm for Zero Analysis



Performance of Kildall's Algorithm

- Why is it guaranteed to terminate?
- What is its complexity?