

Introduction to Program Analysis

14. Static Analysis by Equations

Kihong Heo



Specialized Frameworks

- Practical alternatives to the aforementioned general abstract interpretation framework
 - For simple languages and properties
 - Simple yet powerful enough
- Three specialized frameworks
 - Static analysis by equations (data-flow analysis)
 - Static analysis by monotonic closure (constraint-based analysis)
 - Static analysis by proof construction (type-based analysis)

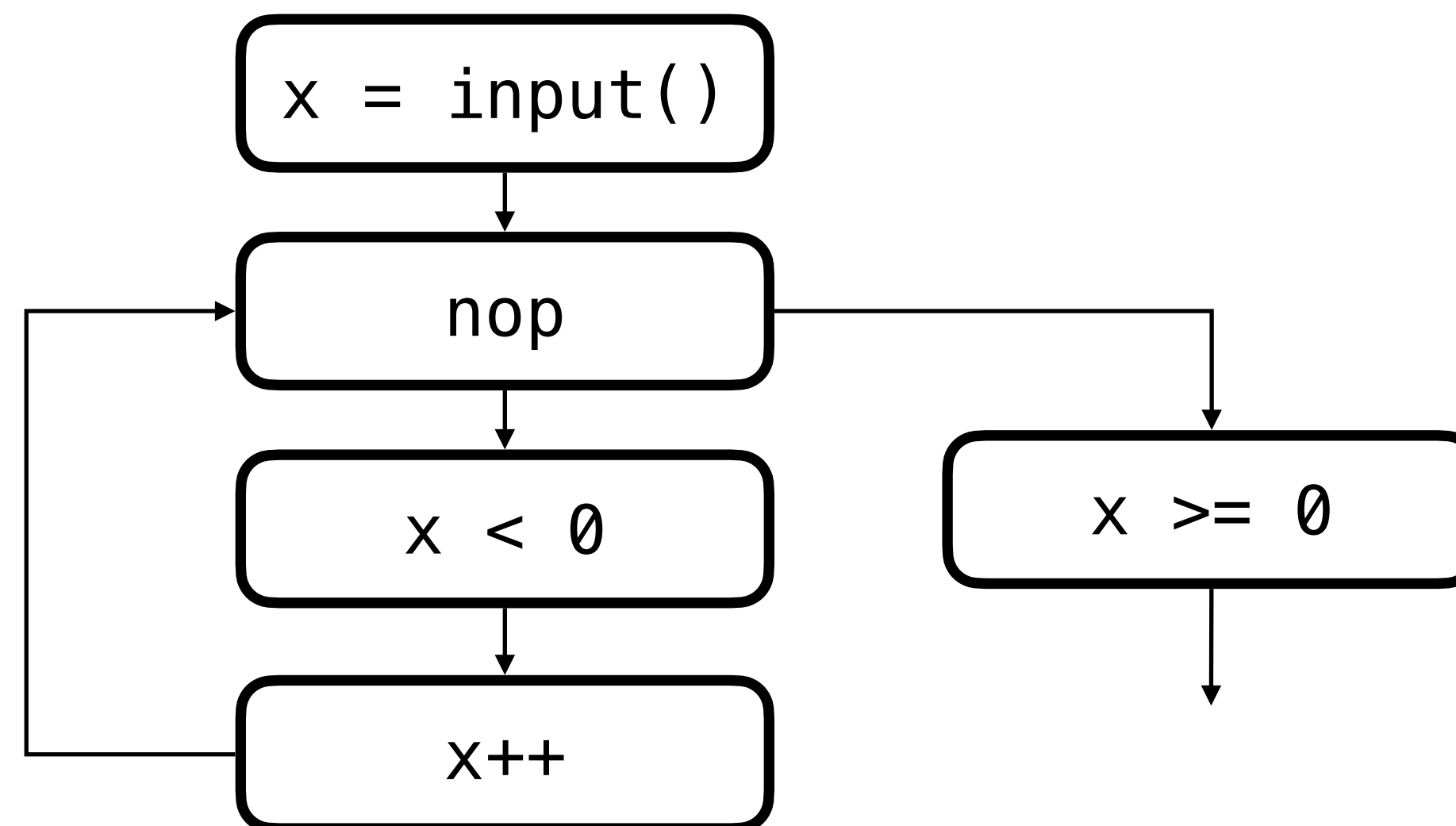
Dataflow Analysis

- A specialized framework: static analysis by equations
- Static analysis = equations setup + equations resolution
- Assumption: Program = control-flow graph (CFG)
 - Nodes: semantic functions (statements)
 - Edges: control flows
 - That is why this approach is called “data-flow” analysis
- Not true for modern languages (e.g., higher order functions, exceptions, etc)

Program as a Graph

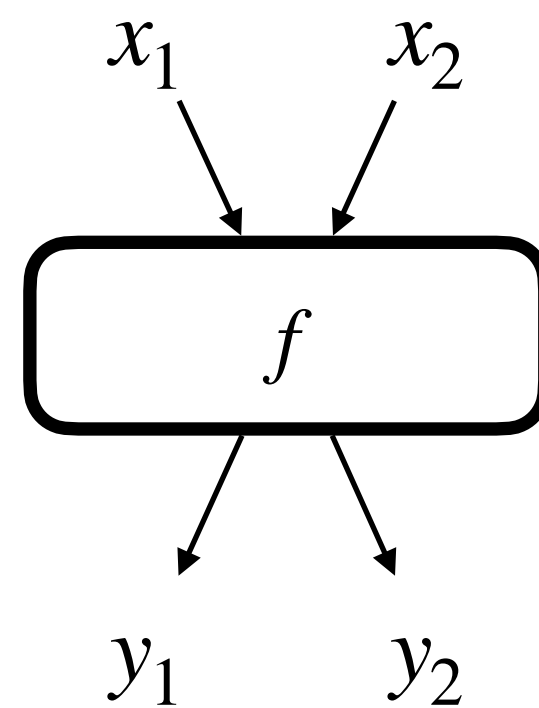
- Control-flow graph: a directed graph $G = (Node, \rightarrow)$
 - Nodes: atomic statements or conditions
 - Edges: execution order between the statements

```
x = input();  
while (x < 0) {  
    x++;  
}
```



Equations

- Describe the abstract states that flow at each edge of the CFG

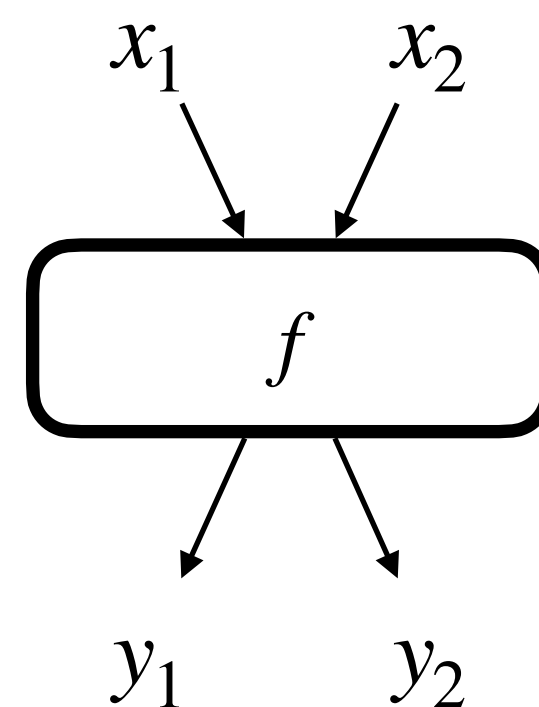


$$y_1 = f(x_1 \sqcup x_2)$$
$$y_2 = f(x_1 \sqcup x_2)$$

- f : a state transfer function for the corresponding statement (i.e., abstract semantics)
- x_i : incoming pre-states to the node (elements of an abstract domain)
- y_i : post-states flowing out along the edges (elements of an abstract domain)

Transfer Function

- A monotonic function that describes the behavior of each statement
 - Defined on abstract domain: $f \in D \rightarrow D$
 - Monotonic: $\forall x, y \in D. x \sqsubseteq y \implies f(x) \sqsubseteq f(y)$
- Larger (less precise) inputs will result in larger outputs
 - Larger = more information merged = less precise



f : “increase by 1”

x_1 : Pos
x_2 : Pos
y_1, y_2 : Pos

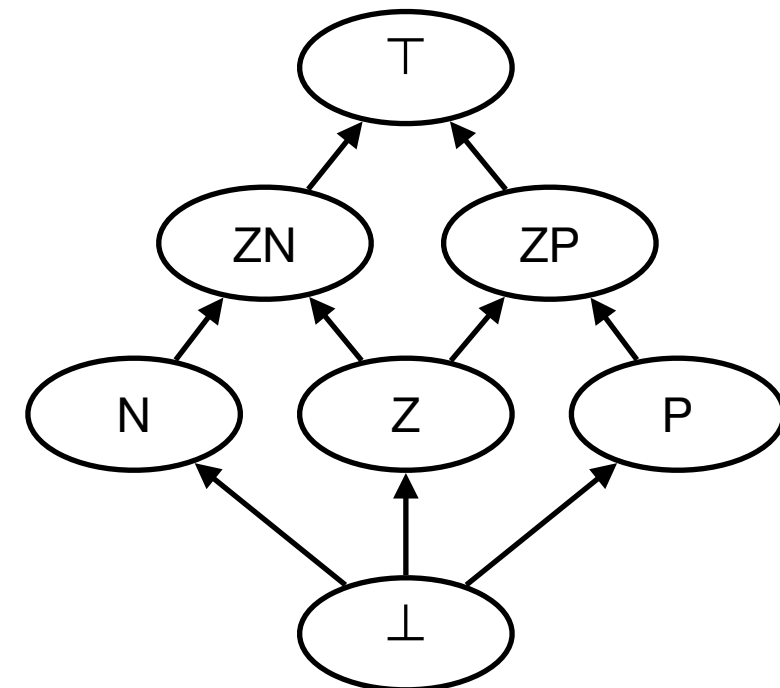
x_1 : Pos
x_2 : Neg
y_1, y_2 : Unknown

Example: Transfer Functions

- Consider a simple grammar with only one variable x
- Goal: estimate the value of x with the sign abstract domain

$C \rightarrow \text{nop}$
 $\quad | x := E$
 $\quad | x \ominus E$
 $E \rightarrow n$
 $\quad | x$
 $\quad | E \oplus E$
 $\quad | \text{input}()$

Grammar



Abstract Domain

$$f_n(x) = \begin{cases} N & \text{if } n < 0 \\ Z & \text{if } n = 0 \\ P & \text{if } n > 0 \end{cases}$$

$$f_x(x) = x$$

$$f_{E_1 + E_2}(x) = f_{E_1}(x) \oplus f_{E_2}(x)$$

$$f_{\text{input}()}(x) = \top$$

Transfer functions
for E

$$f_{\text{nop}}(x) = x$$

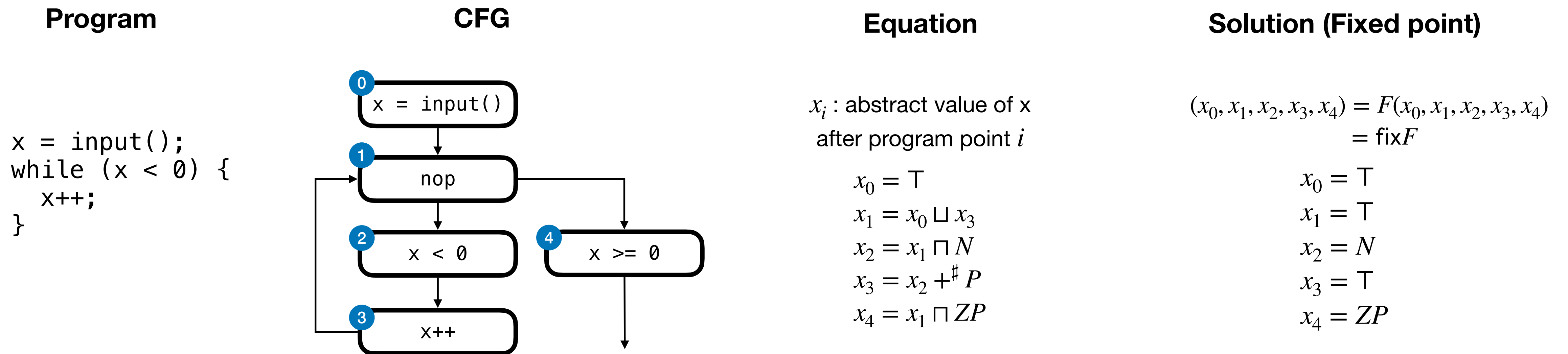
$$f_{x := E}(x) = f_E(x)$$

$$f_{x < E}(x) = \begin{cases} x \sqcap N & \text{if } f_E(x) = N, Z \text{ or } ZN \\ x & \text{otherwise} \end{cases}$$

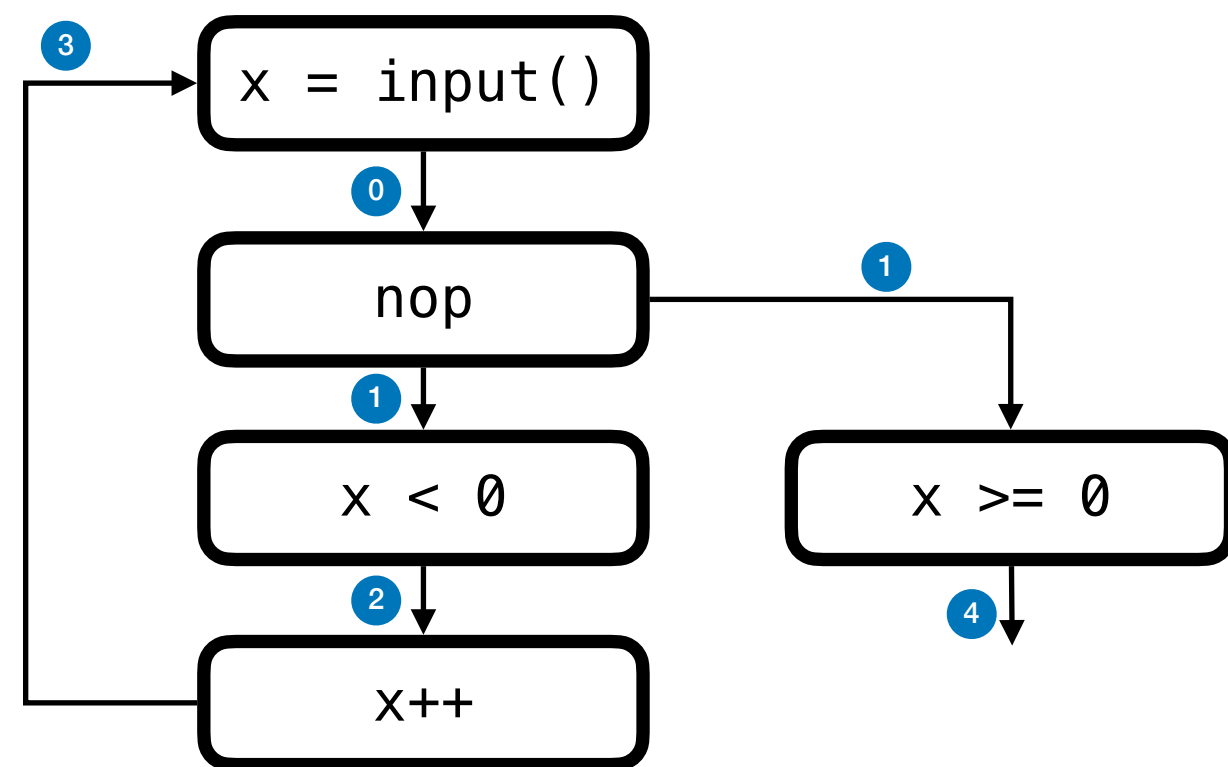
Transfer functions
for C

Static Analysis by Equation

- Static analysis = equation setup + equations resolution
- Semantics = solution of the equation = fixed point of the transfer function



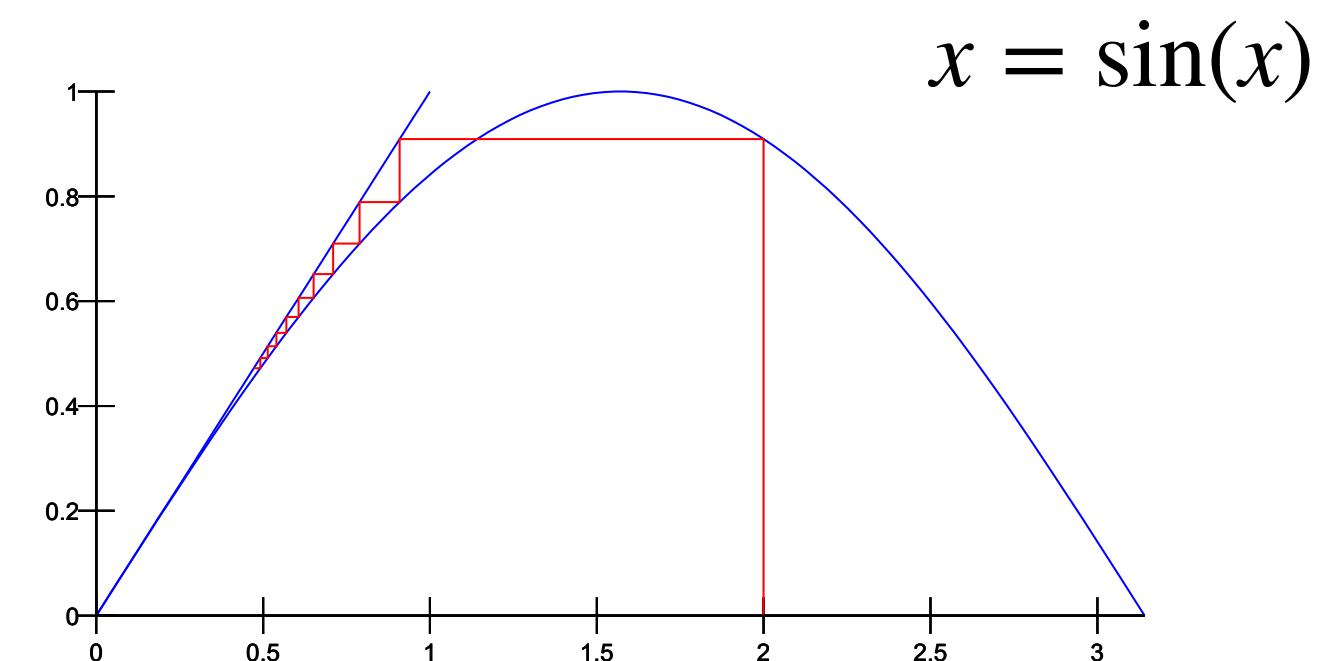
Equations Resolution



$$\begin{aligned}
 x_0 &= \top \\
 x_1 &= x_0 \sqcup x_3 \\
 x_2 &= x_1 \sqcap N \\
 x_3 &= x_2 +^\# P \\
 x_4 &= x_1 \sqcap ZP
 \end{aligned}$$

$$\begin{aligned}
 F &: (Node \rightarrow Z^\#) \rightarrow (Node \rightarrow Z^\#) \\
 F(X) &= \lambda c \in Node. f_c \left(\bigsqcup_{c' \rightarrow c} (X(c')) \right)
 \end{aligned}$$

- Static analysis result = a solution of the equations = a fixed point of F
- Q: How to solve the equations?
- A: A generic iterative algorithm!



Fixed-point Iteration

1. Start from the bottom element
2. Apply the transfer function
3. If the result is stable, return, otherwise goto 2 with the result

$$(x_0, x_1, x_2, x_3, x_4) = F(x_0, x_1, x_2, x_3, x_4)$$

$$x_0 = \top$$

$$x_1 = x_0 \sqcup x_3$$

$$x_2 = x_1 \sqcap N$$

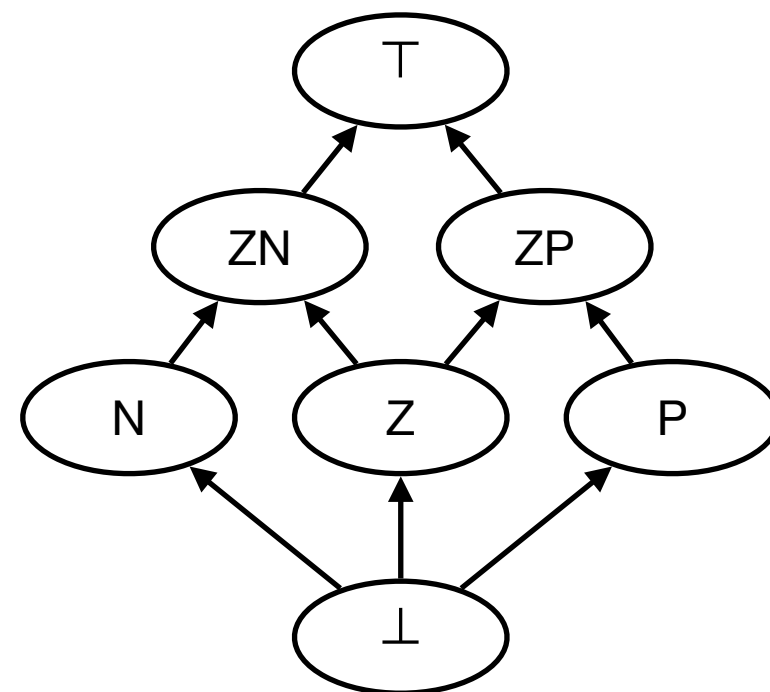
$$x_3 = x_2 +^\# P$$

$$x_4 = x_1 \sqcap ZP$$

x	Iter 0	Iter 1	Iter 2	Iter 3	Iter 4	Iter 5
0	\perp	\top	\top	\top	\top	\top
1	\perp	\perp	\top	\top	\top	\top
2	\perp	\perp	\perp	N	N	N
3	\perp	\perp	\perp	\perp	\top	\top
4	\perp	\perp	\perp	\perp	ZP	ZP

Termination

- Q: Does the fix-point iteration algorithm always terminate?
 - Static analysis must terminate even though the target program does not
- A: Yes! because,
 - Abstract domains have finite height + transfer functions are monotone



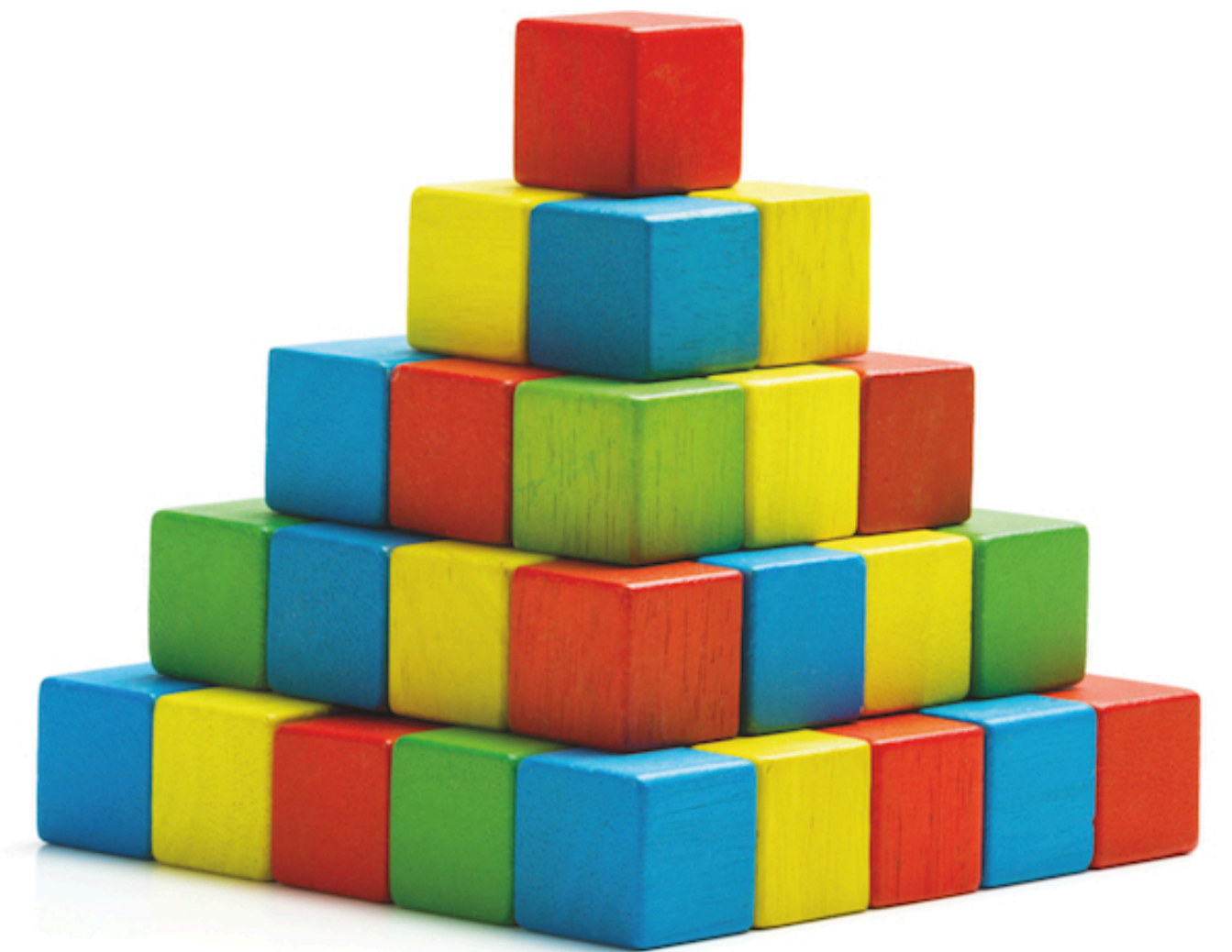
$$\forall x, y \in D . x \sqsubseteq y \implies f(x) \sqsubseteq f(y)$$

Extension to Handle Memory

- How to analyze memory state?
 - Abstract memory: a mapping from variables to abstract values
 - E.g., $Mem^\# = Var \rightarrow Z^\#$
- How to make a CPO for abstract memory?

Constructions on CPOs

- If S is a set, and D_1 and D_2 are CPOs, then the followings are CPOs
 - Lifted set : $D = S_{\perp}$
 - Cartesian product : $D = D_1 \times D_2$
 - Separated sum : $D = D_1 + D_2$
 - Function : $D = D_1 \rightarrow D_2$



Abstract Semantics with Memory

- Abstract domain: $Mem^\# = Var_\perp \rightarrow Z^\#$
 - Order: $m_1^\# \sqsubseteq m_2^\# \iff \forall x \in Var_\perp . m_1^\#(x) \sqsubseteq m_2^\#(x)$
 - Join: $m_1^\# \sqcup m_2^\# = \lambda x . m_1^\#(x) \sqcup m_2^\#(x)$
- Abstract semantics

$$F : (Node \rightarrow Mem^\#) \rightarrow (Node \rightarrow Mem^\#)$$

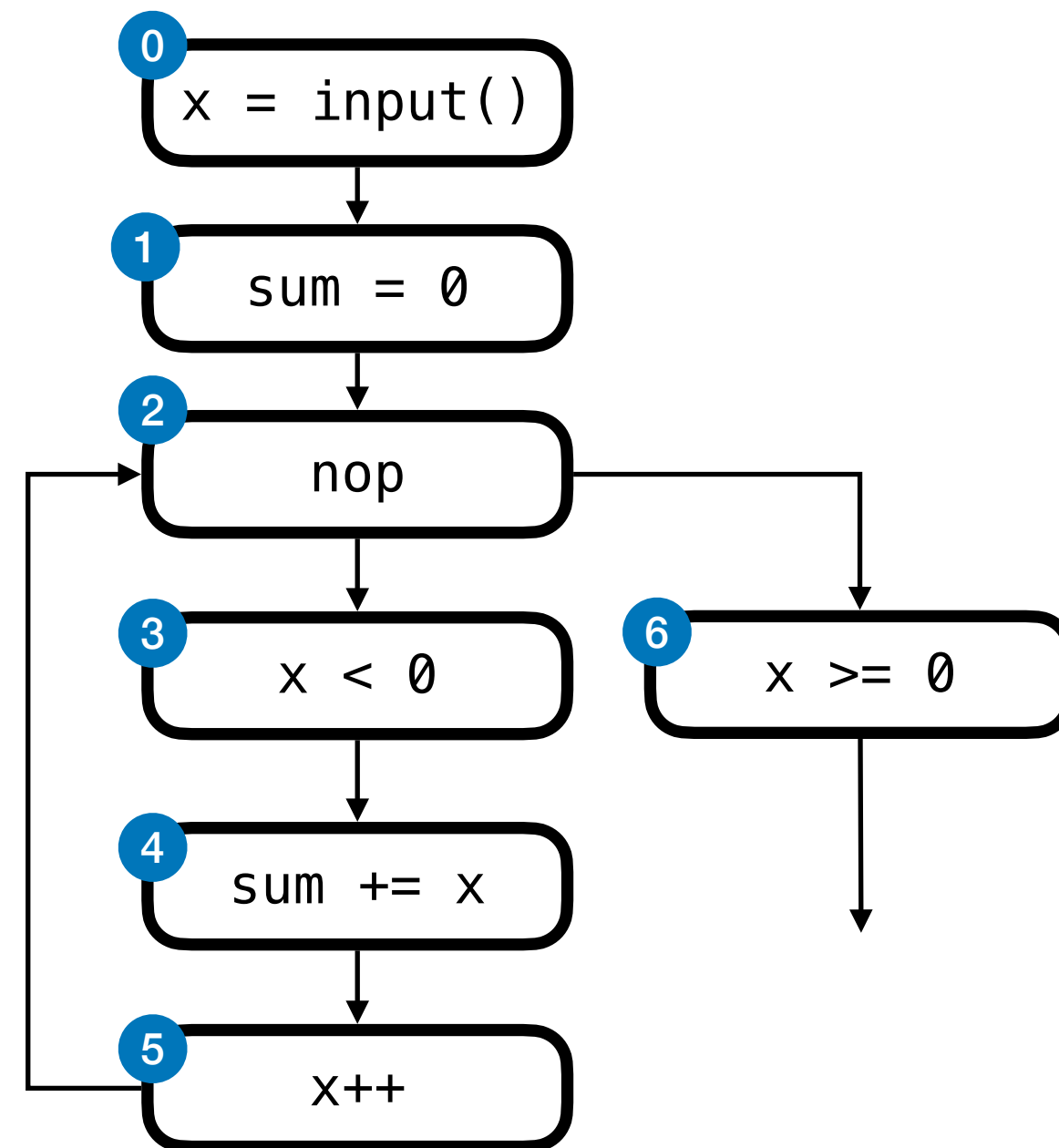
$$F(X) = \lambda c . f_n \left(\bigsqcup_{c' \rightarrow c} (X(c')) \right)$$

Example

Program

```
x = input();
sum = 0;
while (x < 0) {
    sum += x;
    x++;
}
```

CFG



Equation

m_i : abstract memory
after program point i

$$\begin{aligned}
 m_0 &= \{x \mapsto \top\} \\
 m_1 &= \{x \mapsto \top, \text{sum} \mapsto Z\} \\
 m_2 &= m_1 \sqcup m_5 \\
 m_3 &= m_2 \sqcap \top \{x \mapsto N\} \\
 m_4 &= m_3 \{ \text{sum} \mapsto m_3(\text{sum}) +^\# m_3(x) \} \\
 m_5 &= m_4 \{ x \mapsto m_4(x) +^\# P \} \\
 m_6 &= m_2 \sqcap \top \{x \mapsto ZP\}
 \end{aligned}$$

Solution (Fixed point)

$$\begin{aligned}
 (m_0, m_1, m_2, m_3, m_4, m_5) &= F(m_0, m_1, m_2, m_3, m_4, m_5) \\
 &= \text{fix} F
 \end{aligned}$$

$$\begin{aligned}
 m_0 &= \{x \mapsto \top\} \\
 m_1 &= \{x \mapsto \top, \text{sum} \mapsto Z\} \\
 m_2 &= \{x \mapsto \top, \text{sum} \mapsto ZP\} \\
 m_3 &= \{x \mapsto N, \text{sum} \mapsto ZP\} \\
 m_4 &= \{x \mapsto N, \text{sum} \mapsto ZP\} \\
 m_5 &= \{x \mapsto \top, \text{sum} \mapsto ZP\} \\
 m_6 &= \{x \mapsto P, \text{sum} \mapsto ZP\}
 \end{aligned}$$

Conclusion

- Data-flow analysis: a specialized framework of static analysis
- Static analysis by equation
 - Static analysis = equations setup + equations resolution
- Limited but powerful enough for simple properties
- Limitations:
 - Control-flow before analysis?
 - Sound transfer function?
 - Systematic approaches to prove the correctness or vary the accuracy?