

# IS593: Language-based Security

## 3. Denotational Semantics

Kihong Heo



# Different Styles of Semantics (Revisited)

- **Operational Semantics:** “**How** to compute the execution result”
  - so-called transitional style
  - $3 * (2 + 1) : 3 * (2 + 1) \rightarrow 3 * 3 \rightarrow 9$
- **Denotational Semantics:** “**What** the execution result is”
  - so-called compositional style
  - $3 * (2 + 1) : 9$
- ...

*Different approaches for different purposes and languages*

# Denotation Semantics

- Mathematical meaning of a program (i.e., no state or transition)
- Program semantics is a function from input states to output states
- The semantics of a program is determined by that of each subcomponents (i.e., **compositional**)
- Notation:  $\llbracket P \rrbracket : \mathbb{M} \rightarrow \mathbb{M}$

# A Simple Imperative Language

$E ::=$		arithmetic expressions
	$n$	integer constants
	$x$	variable
	$E \odot E$	binary operation
$B ::=$		boolean expression
	$\text{true} \mid \text{false}$	boolean constants
	$E \oslash E$	comparison expressions
$C ::=$		commands
	$\text{skip}$	command that does nothing
	$C; C$	sequence
	$x := E$	assignment
	$\text{input}(x)$	command reading of a value
	$\text{if } B \text{ then } C \text{ else } C$	conditional command
	$\text{while } B \text{ } C$	loop command

# Semantic Domains

- Sets of semantic objects
- Memory is a mapping  $\mathbb{M} = \mathbb{X} \rightarrow \mathbb{V}$ 
  - $\mathbb{X}$  : the set of variables
  - $\mathbb{V}$  : the set of integers ( $\mathbb{Z}$ ) and booleans ( $\mathbb{B}$ )
- Example:  $\llbracket x := 7 ; y := 3 \rrbracket \{\} = \{x \mapsto 7, y \mapsto 3\}$

# Semantics of Expressions

$$\llbracket E \rrbracket : \mathbb{M} \rightarrow \mathbb{Z}$$

$$\llbracket n \rrbracket(m) = n$$

$$\llbracket x \rrbracket(m) = m(x)$$

$$\llbracket E_1 \odot E_2 \rrbracket(m) = \llbracket E_1 \rrbracket(m) \odot \llbracket E_2 \rrbracket(m)$$

$$\llbracket B \rrbracket : \mathbb{M} \rightarrow \mathbb{B}$$

$$\llbracket \text{true} \rrbracket(m) = \text{true}$$

$$\llbracket \text{false} \rrbracket(m) = \text{false}$$

$$\llbracket E_1 \oslash E_2 \rrbracket(m) = \llbracket E_1 \rrbracket(m) \oslash \llbracket E_2 \rrbracket(m)$$

# Semantics of Commands (1)

$$\llbracket C \rrbracket : \mathbb{M} \rightarrow \mathbb{M}$$

$$\llbracket \text{skip} \rrbracket(m) = m$$

$$\llbracket C_0 ; C_1 \rrbracket(m) = \llbracket C_1 \rrbracket(\llbracket C_0 \rrbracket(m))$$

$$\llbracket x := E \rrbracket(m) = m\{x \mapsto \llbracket E \rrbracket(m)\}$$

$$\llbracket \text{input}(x) \rrbracket(m) = m\{x \mapsto n\}$$

$$\llbracket \text{if } B \text{ then } C_1 \text{ else } C_2 \rrbracket(m) = \begin{cases} \llbracket C_1 \rrbracket(m) & \text{if } \llbracket B \rrbracket(m) = \text{true} \\ \llbracket C_2 \rrbracket(m) & \text{if } \llbracket B \rrbracket(m) = \text{false} \end{cases}$$

**Compositional? Yes!**

# Semantics of Commands (2)

- Semantics of While loop

$$\llbracket \text{while } B \ C \rrbracket(m) = \begin{cases} \llbracket \text{while } B \ C \rrbracket(\llbracket C \rrbracket(m)) & \text{if } \llbracket B \rrbracket(m) = \text{true} \\ m & \text{if } \llbracket B \rrbracket(m) = \text{false} \end{cases}$$

**Compositional?**    **NO!**

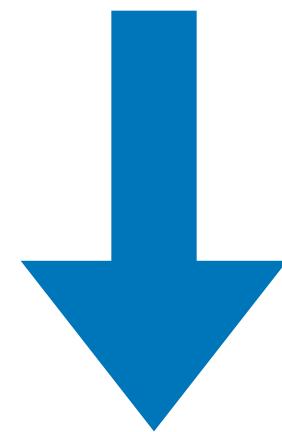
- C.f) inductive vs compositional definition of the Fibonacci function

$$fib(n) = fib(n - 1) + fib(n - 2) \quad \text{vs} \quad fib(n) = \frac{1}{\sqrt{5}} \left( \left( \frac{1 + \sqrt{5}}{2} \right)^n - \left( \frac{1 - \sqrt{5}}{2} \right)^n \right)$$

# Semantics as Fixed Point (1)

- Consider this inductive definition as an equation

$$\llbracket \text{while } B \ C \rrbracket(m) = \begin{cases} \llbracket \text{while } B \ C \rrbracket(\llbracket C \rrbracket(m)) & \text{if } \llbracket B \rrbracket(m) = \text{true} \\ m & \text{if } \llbracket B \rrbracket(m) = \text{false} \end{cases}$$



$$\llbracket \text{while } B \ C \rrbracket = \mathcal{F}_{B,C}(\llbracket \text{while } B \ C \rrbracket)$$

where  $\mathcal{F}_{B,C}(X) = \lambda m. \begin{cases} X(\llbracket C \rrbracket(m)) & \text{if } \llbracket B \rrbracket(m) = \text{true} \\ m & \text{if } \llbracket B \rrbracket(m) = \text{false} \end{cases}$

$$F(X) = \dots$$

$$\iff F = \lambda X. \dots$$

# Semantics as Fixed Point (2)

- What is the semantics of loops? A solution of this equation!

$$\llbracket \text{while } B \text{ } C \rrbracket = \mathcal{F}_{B,C}(\llbracket \text{while } B \text{ } C \rrbracket) \quad \longleftrightarrow \quad X = \mathcal{F}_{B,C}(X) \text{ where } X = \llbracket \text{while } B \text{ } C \rrbracket$$

- Solution: a fixed point of  $\mathcal{F}_{B,C}$
- Especially, we define the semantics as the least fixed point

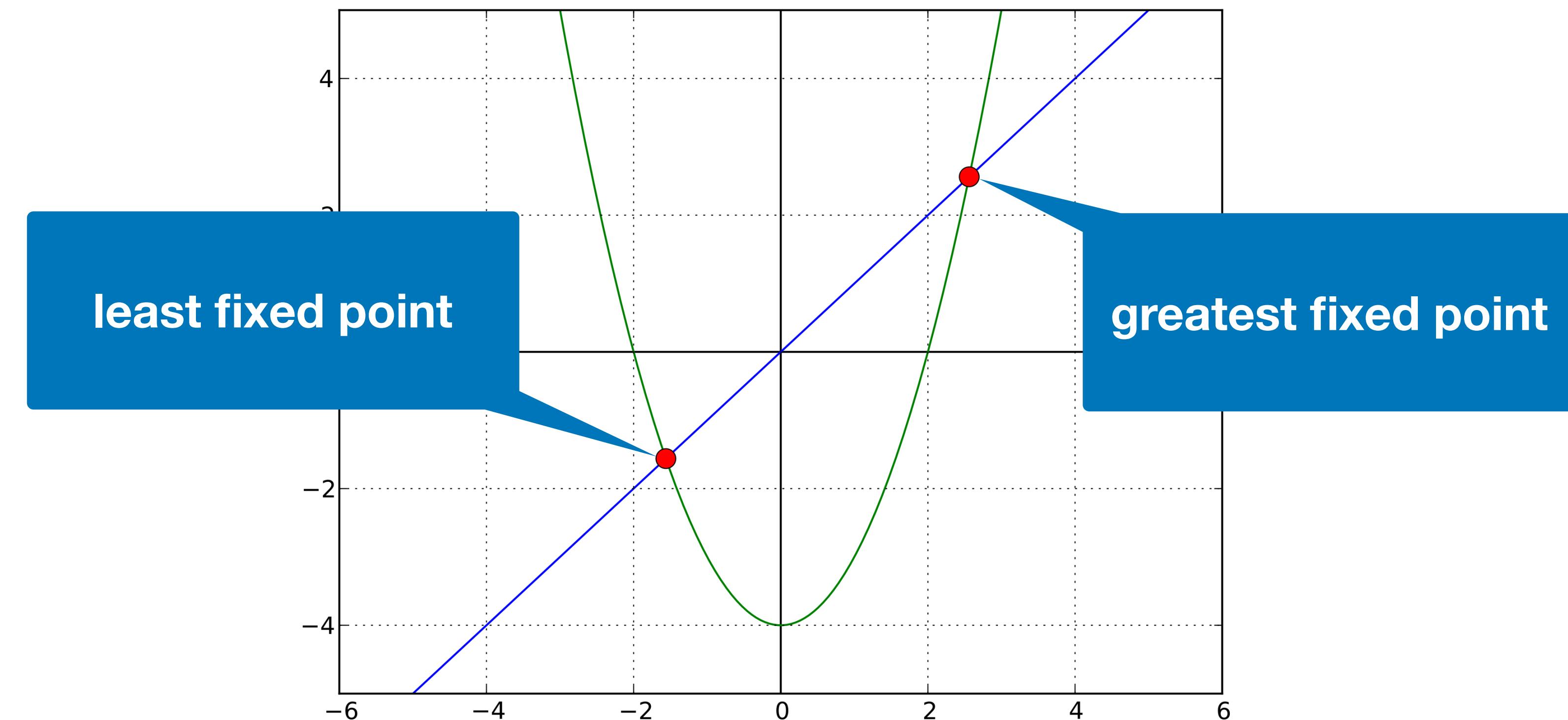
$$\llbracket \text{while } B \text{ } C \rrbracket = \text{lfp} \mathcal{F}_{B,C}$$

$$\text{where } \mathcal{F}_{B,C}(X)(m) = \begin{cases} X(\llbracket C \rrbracket(m)) & \text{if } \llbracket B \rrbracket(m) = \text{true} \\ m & \text{if } \llbracket B \rrbracket(m) = \text{false} \end{cases}$$

Compositional? Yes!

# Exercise

- $f(x) = x^2 - 4$



\*[https://en.wikipedia.org/wiki/Least\\_fixed\\_point](https://en.wikipedia.org/wiki/Least_fixed_point)

# Exercise

- $\mathbb{N} = \{0\} \cup \{n + 1 \mid n \in \mathbb{N}\}$

$\mathbb{N}$  is the least fixed point of  $F$  where  $F(X) = \{0\} \cup \{n + 1 \mid n \in X\}$

$$\therefore \mathbb{N} = \text{lfp}(\lambda X. \{0\} \cup \{n + 1 \mid n \in X\})$$

# Exercise

- $\mathbb{L} = \{\text{nil}\} \cup \{0 :: l \mid l \in \mathbb{L}\}$

$\mathbb{L}$  is the least fixed point of  $F$  where  $F(X) = \{\text{nil}\} \cup \{0 :: l \mid l \in X\}$

$$\therefore \mathbb{L} = \text{lfp}(\lambda X. \{\text{nil}\} \cup \{0 :: l \mid l \in X\})$$

# Exercise

- $\text{reach}(N) = N \cup \text{reach}(\text{next}(N))$

$$\text{reach} = \lambda N. N \cup \text{reach}(\text{next}(N))$$

$\text{reach}$  is the least fixed point of  $F$  where  $F(X) = \lambda N. N \cup X(\text{next}(N))$

$$\therefore \text{reach} = \text{lfp}(\lambda X. (\lambda N. N \cup X(\text{next}(N))))$$

# Exercise

- $\text{fact}(N) = \text{if } N = 0 ? 1 : N * \text{fact}(N - 1)$

$$\text{fact} = \lambda N. \text{if } N = 0 ? 1 : N * \text{fact}(N - 1)$$

$\text{fact}$  is the least fixed point of  $F$  where  $F(X) = \lambda N. \text{if } N = 0 ? 1 : N * X(N - 1)$

$$\therefore \text{fact} = \text{lfp}(\lambda X. (\lambda N. \text{if } N = 0 ? 1 : N * X(N - 1)))$$

# Questions

- Why does the semantic equation have a solution?
- Does the equation have a unique solution?
- How to compute the solution?



# Domain Theory

- Semantics of a program is an element of a domain called **CPO** (complete partial order set)
  - Example:  $\llbracket C \rrbracket \in \mathbb{M} \rightarrow \mathbb{M}$
- Semantics of a program is the **least fixed point** of a **continuous function**
  - Example:  $\mathcal{F}_{B,C} : (\mathbb{M} \rightarrow \mathbb{M}) \rightarrow (\mathbb{M} \rightarrow \mathbb{M})$
  - Established by Dana Scott
    - *Outline of a Mathematical Theory of Computation*, 1970
    - *Mathematical Concepts in Programming Language Semantics*, 1972

# Partial Order

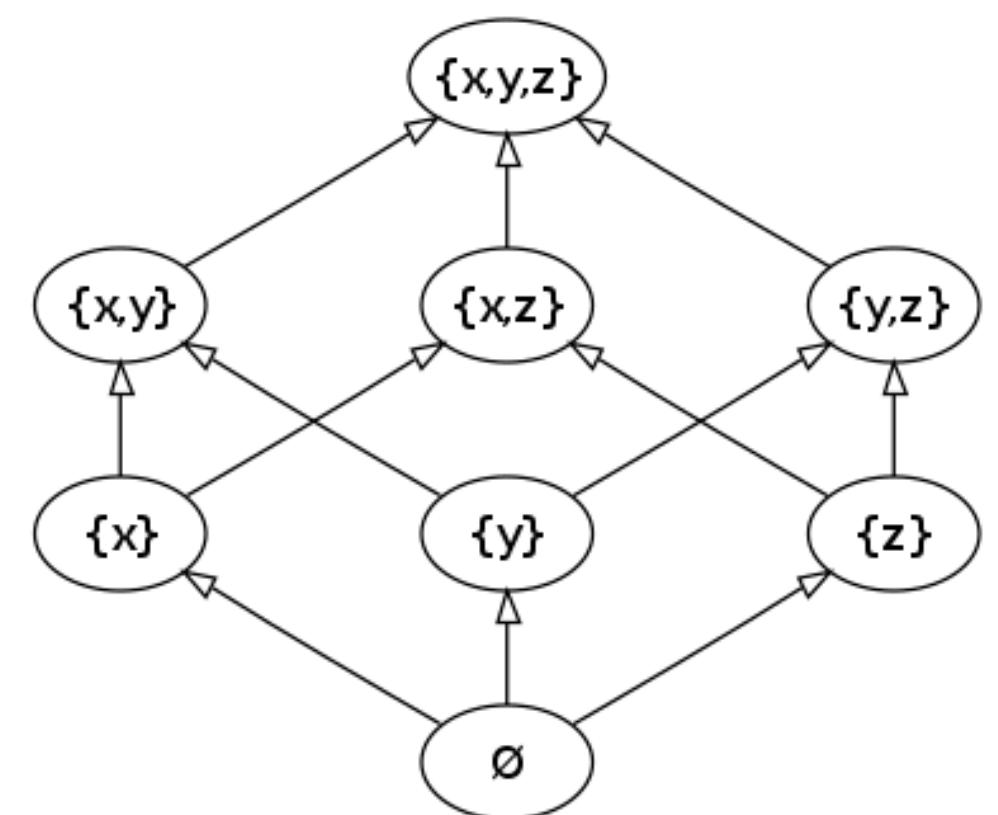
**Definition (Partial Order).** A binary relation  $\sqsubseteq$  is a **partial order** on a set  $D$  if it has:

1. reflexivity:  $a \sqsubseteq a$  for all  $a \in D$
2. Antisymmetry:  $a \sqsubseteq b$  and  $b \sqsubseteq a$  implies  $a = b$
3. Transitivity:  $a \sqsubseteq b$  and  $b \sqsubseteq c$  implies  $a \sqsubseteq c$

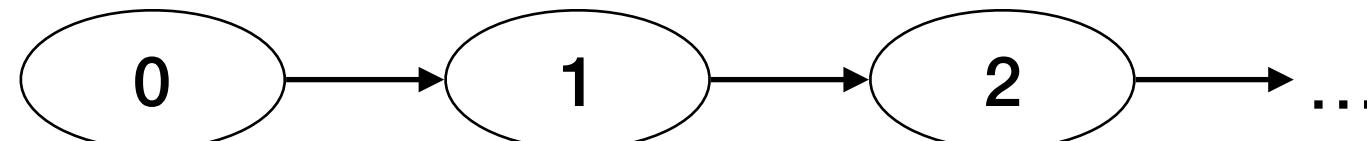
A set  $D$  with a partial order  $\sqsubseteq$  is called a **partially ordered set** ( $(D, \sqsubseteq)$ ), or simply **poset**.

# Partial Order

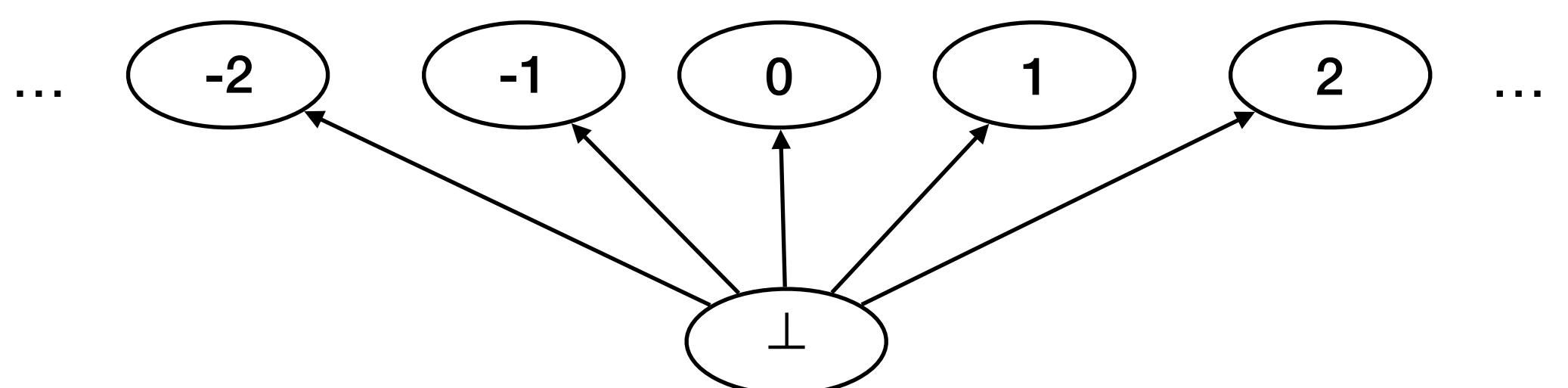
- Example 1:  $(\wp(\{x, y, z\}), \subseteq)$



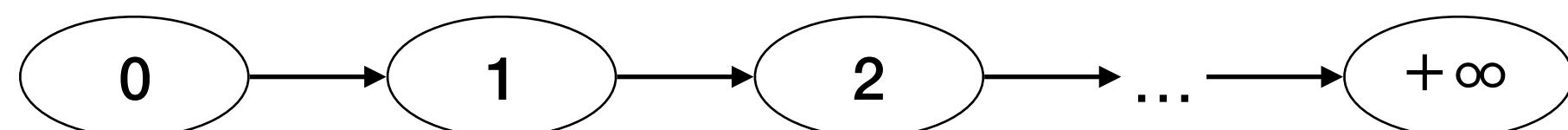
- Example 3:  $(\mathbb{N}, \leq)$



- Example 2:  $(\mathbb{Z}_\perp, \sqsubseteq)$



- Example 4:  $(\mathbb{N} + \{\infty\}, \leq)$



# Least Upper Bound

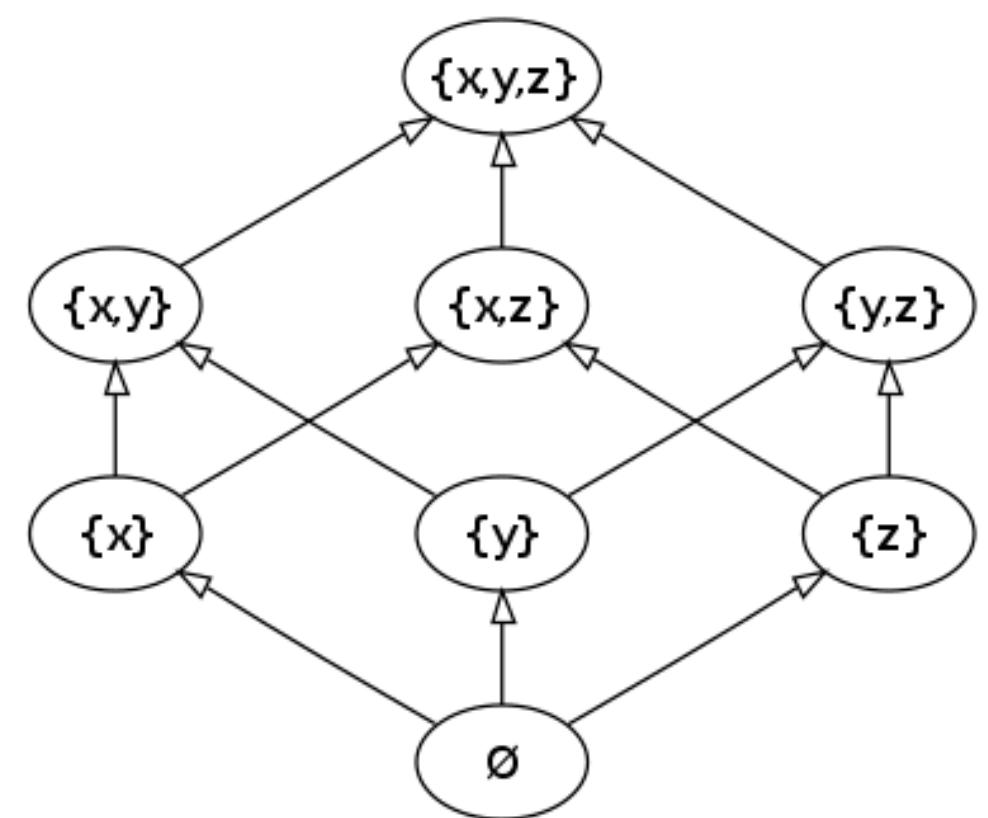
**Definition (Least Upper Bound).** For a partial ordered set  $(D, \sqsubseteq)$  and subset  $X \subseteq D$ ,  $d \in D$  is an **upper bound** of  $X$  iff

$$\forall x \in X. x \sqsubseteq d.$$

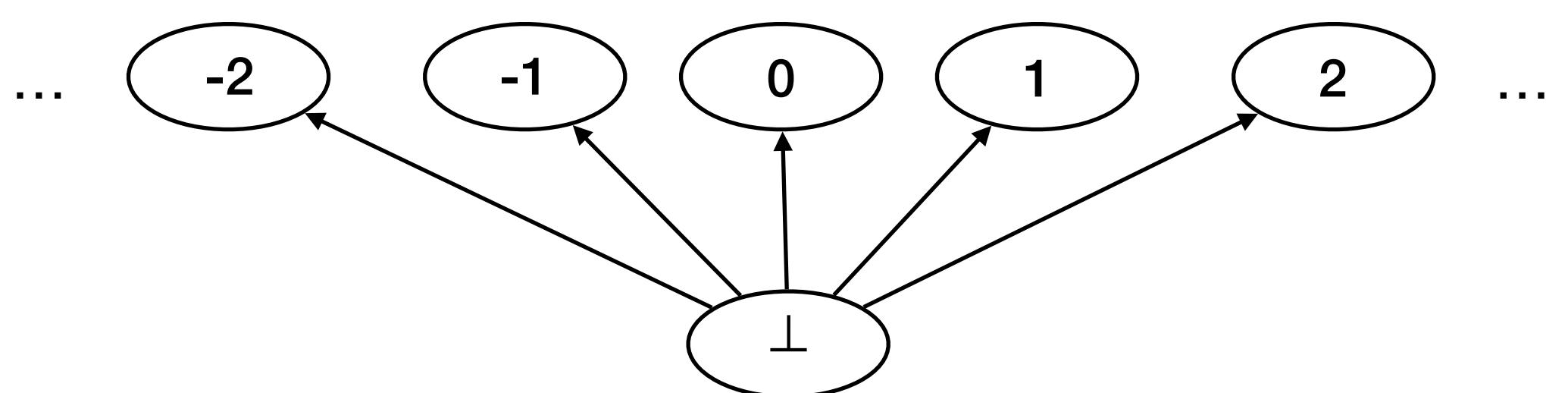
An upper bound  $d$  is the **least upper bound** of  $X$  iff for all upper bounds  $y$  of  $X$ ,  $d \sqsubseteq y$ . The least upper bound of  $X$  is denoted by  $\sqcup X$ .

# Least Upper Bound

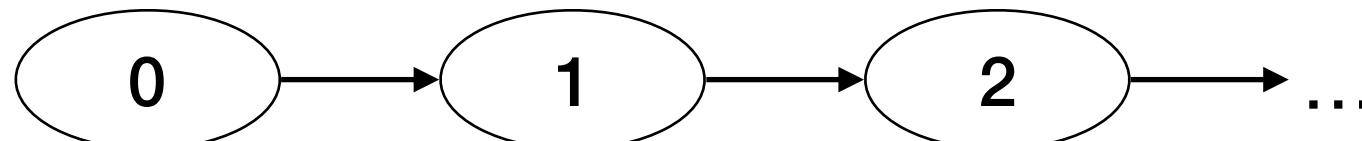
- Example 1:  $(\wp(\{x, y, z\}), \subseteq)$



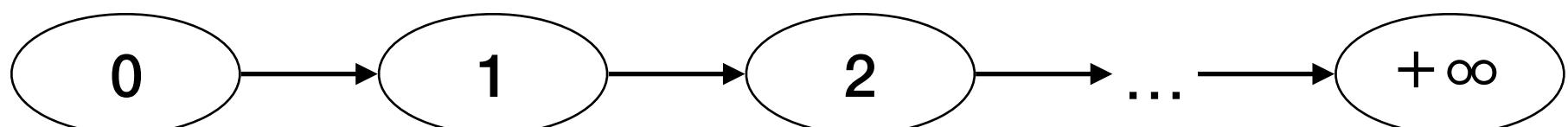
- Example 2:  $(\mathbb{Z}_\perp, \sqsubseteq)$



- Example 3:  $(\mathbb{N}, \leq)$



- Example 4:  $(\mathbb{N} + \{+\infty\}, \leq)$



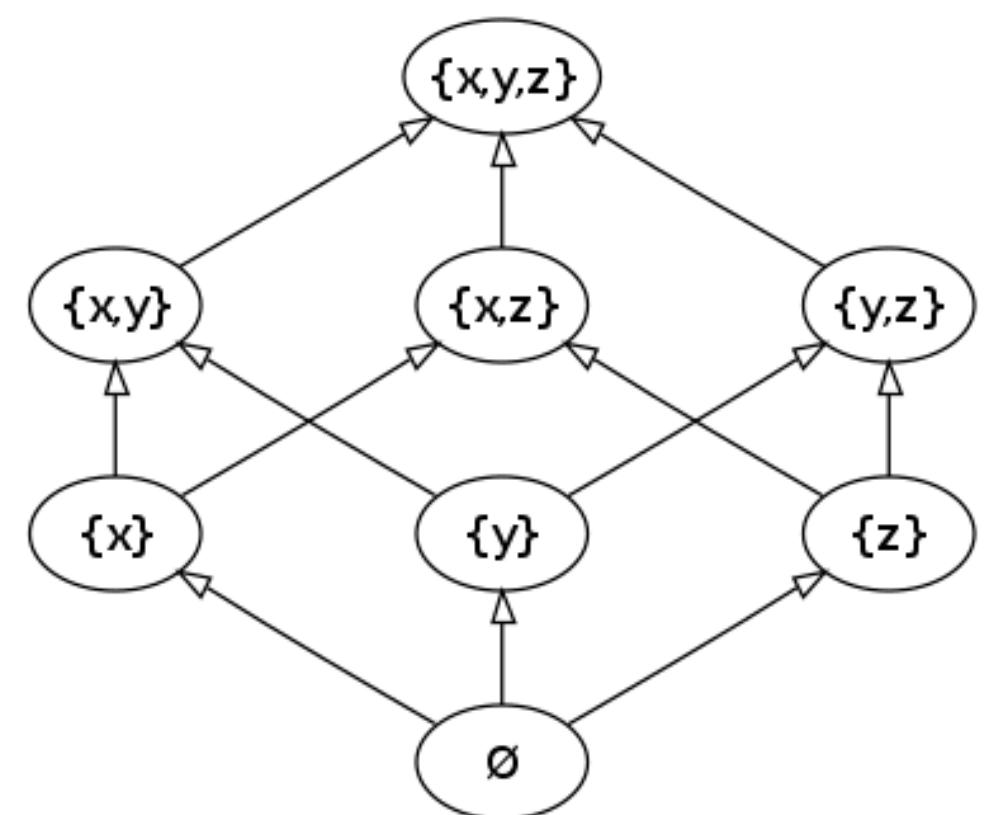
# Chain

**Definition (Chain).** Let  $(D, \sqsubseteq)$  be a partial ordered set. A subset  $X \subseteq D$  is called **chain** if  $X$  is totally ordered:

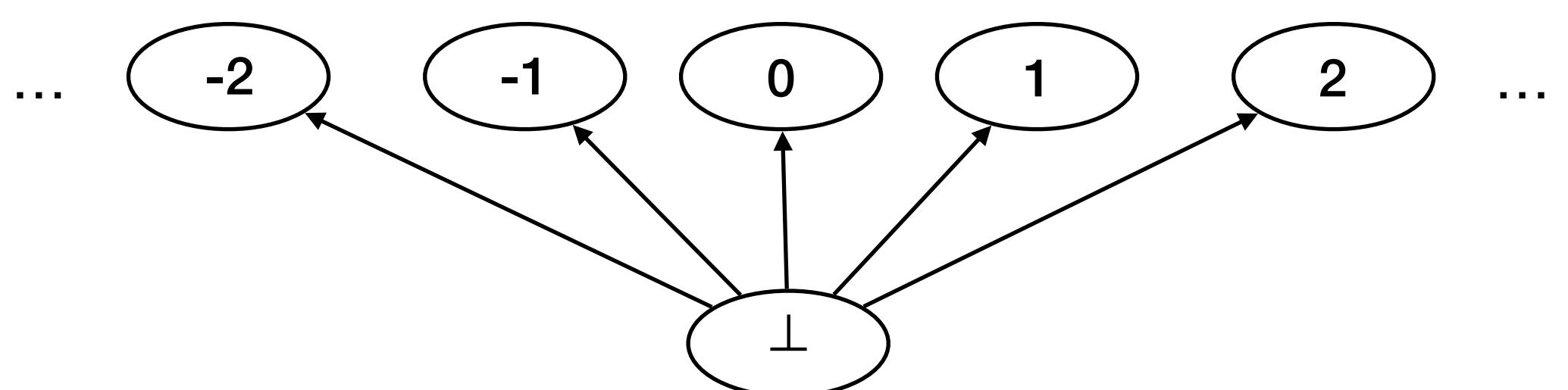
$$\forall x_1, x_2 \in X. x_1 \sqsubseteq x_2 \text{ or } x_2 \sqsubseteq x_1.$$

# Chain

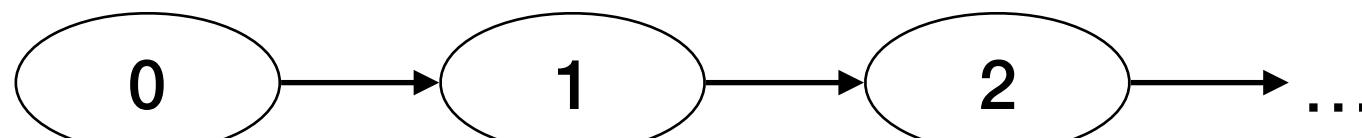
- Example 1:  $(\wp(\{x, y, z\}), \subseteq)$



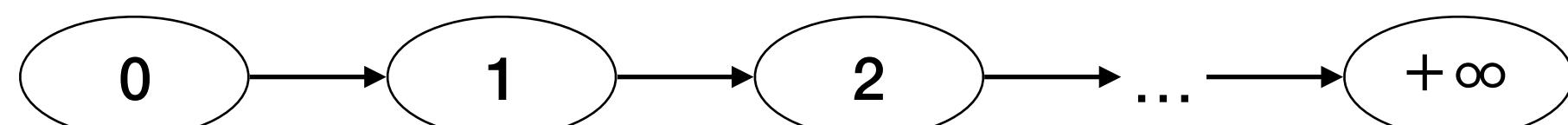
- Example 2:  $(\mathbb{Z}_\perp, \sqsubseteq)$



- Example 3:  $(\mathbb{N}, \leq)$



- Example 4:  $(\mathbb{N} + \{+\infty\}, \leq)$



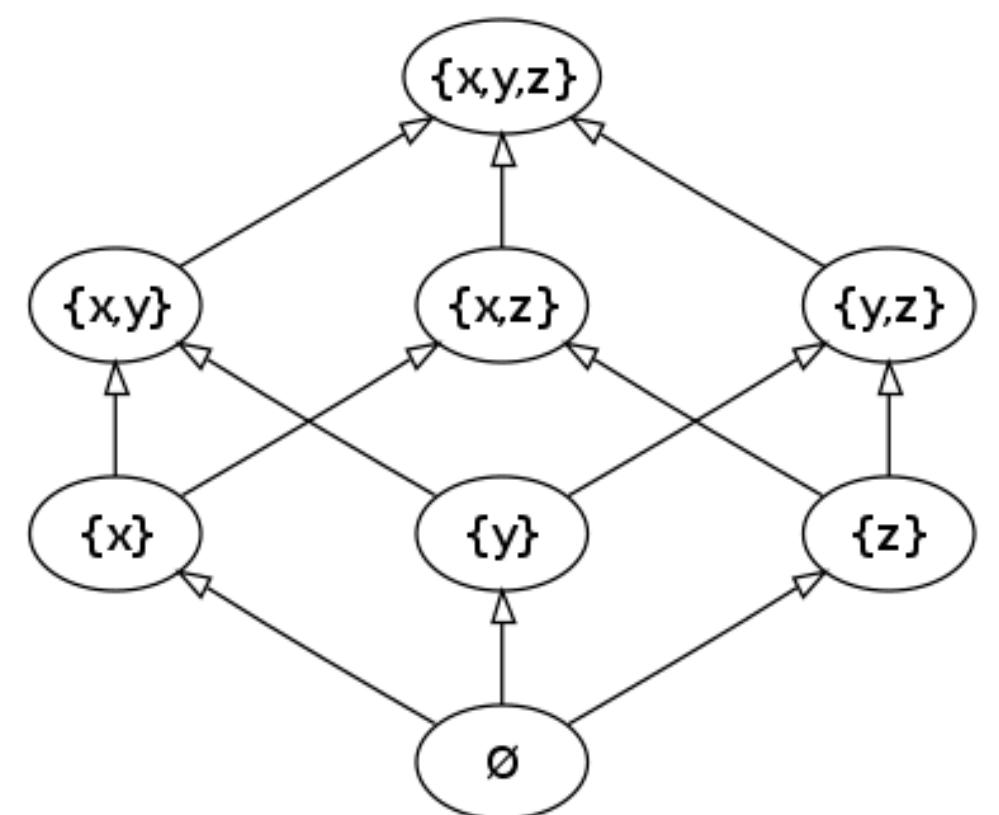
# CPO

**Definition (CPO).** A poset  $(D, \sqsubseteq)$  is a **CPO** (complete partial order) if every chain  $X$  of  $D$  has  $\bigsqcup X \in D$ .

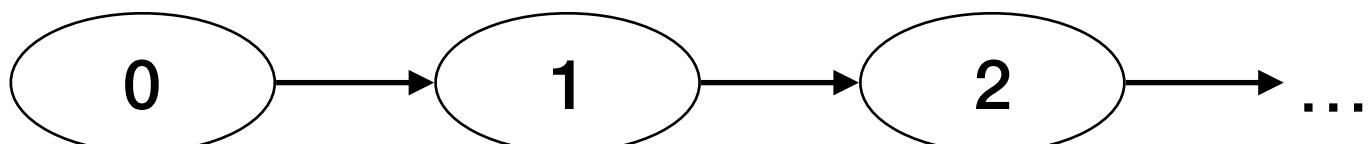
**Lemma.** If poset  $(D, \sqsubseteq)$  is a CPO, it has the **least element**  $\perp = \bigsqcup \emptyset$

# CPO

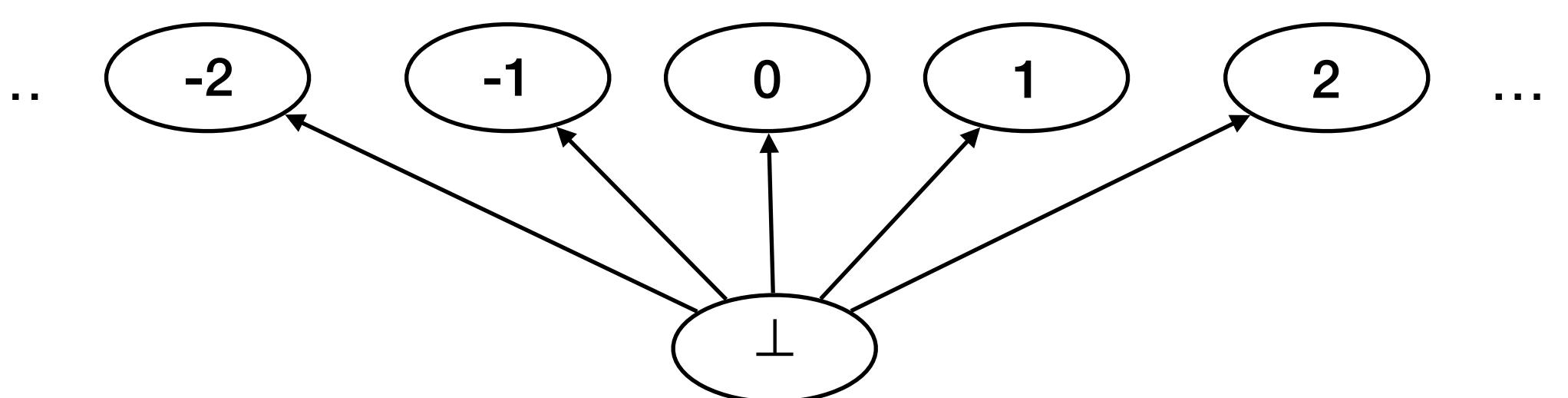
- Example 1:  $(\wp(\{x, y, z\}), \subseteq)$



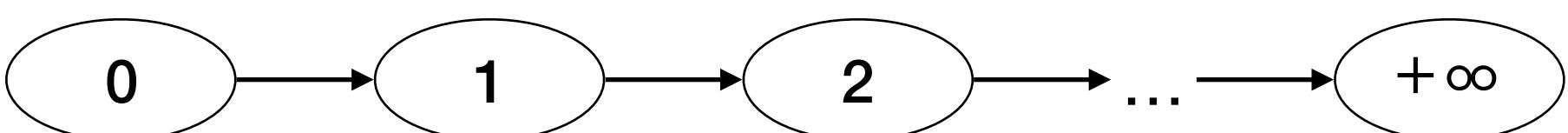
- Example 3:  $(\mathbb{N}, \leq)$



- Example 2:  $(\mathbb{Z}_\perp, \sqsubseteq)$



- Example 4:  $(\mathbb{N} + \{+\infty\}, \leq)$

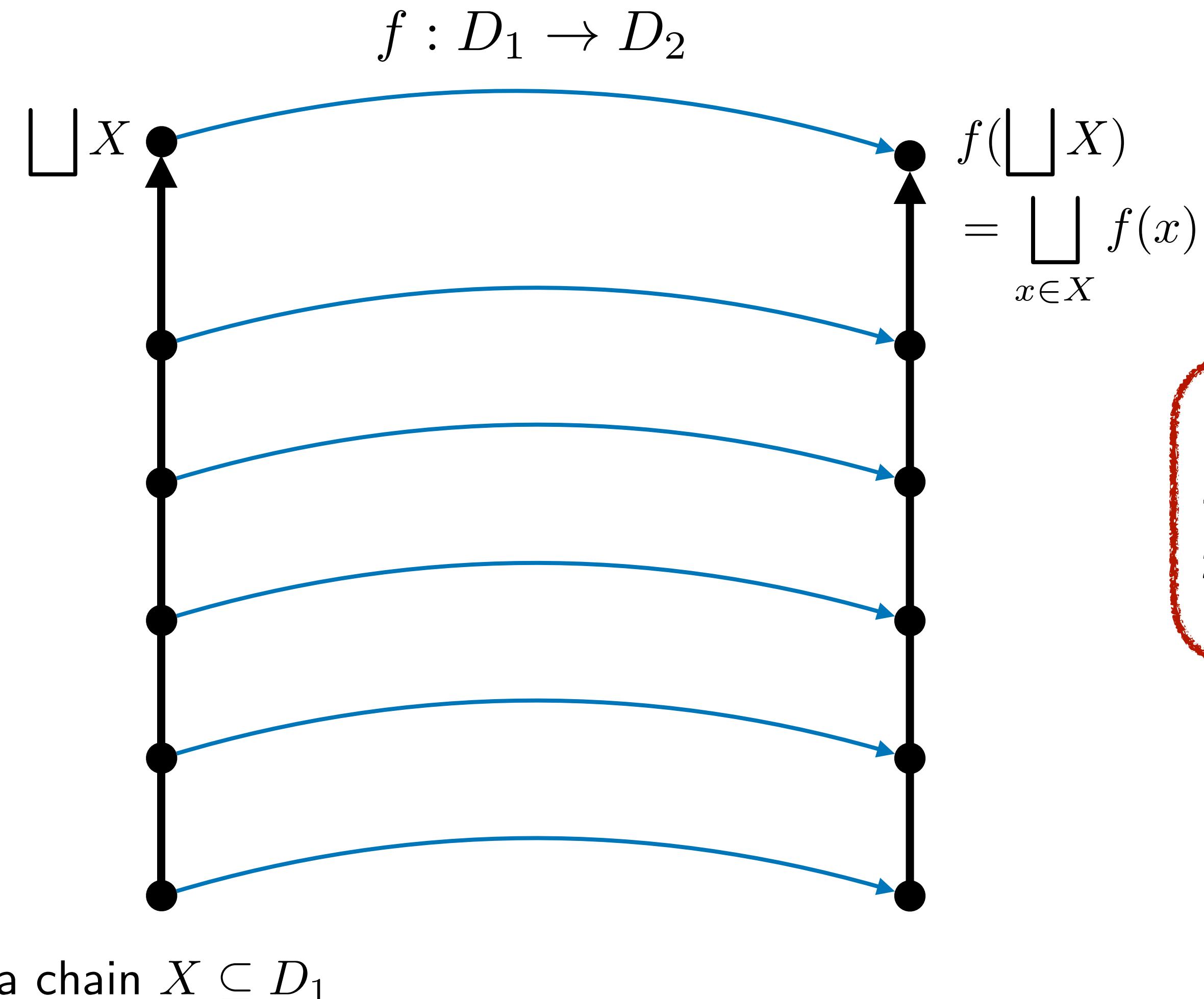


# Continuous Function

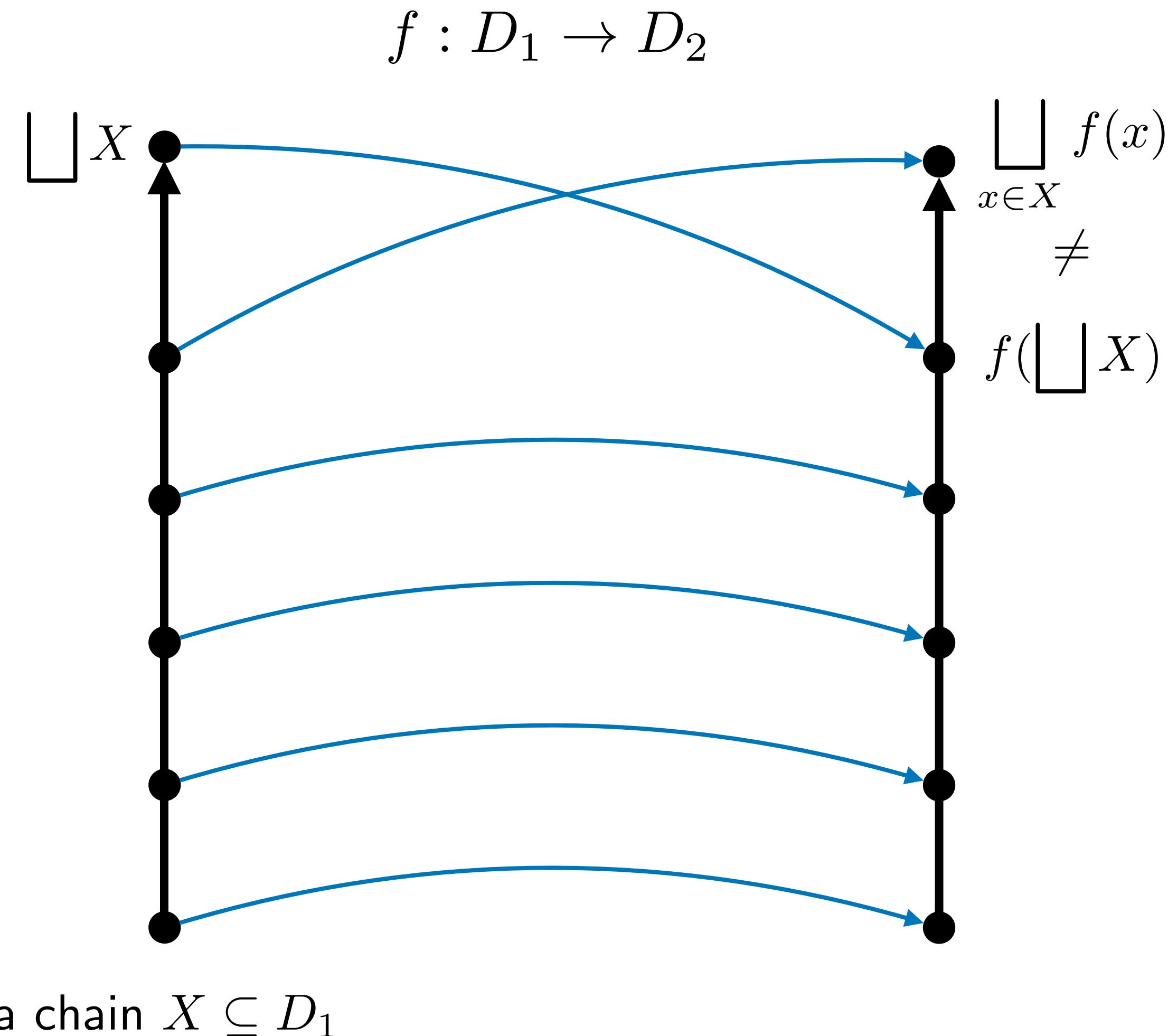
**Definition (Continuous Function).** Given two partially ordered sets  $D_1$  and  $D_2$ , a function  $f : D_1 \rightarrow D_2$  is **continuous** if it preserves least upper bounds of chains:

$$\forall \text{chain } X \subseteq D_1. \bigcup_{x \in X} f(x) = f(\bigcup X).$$

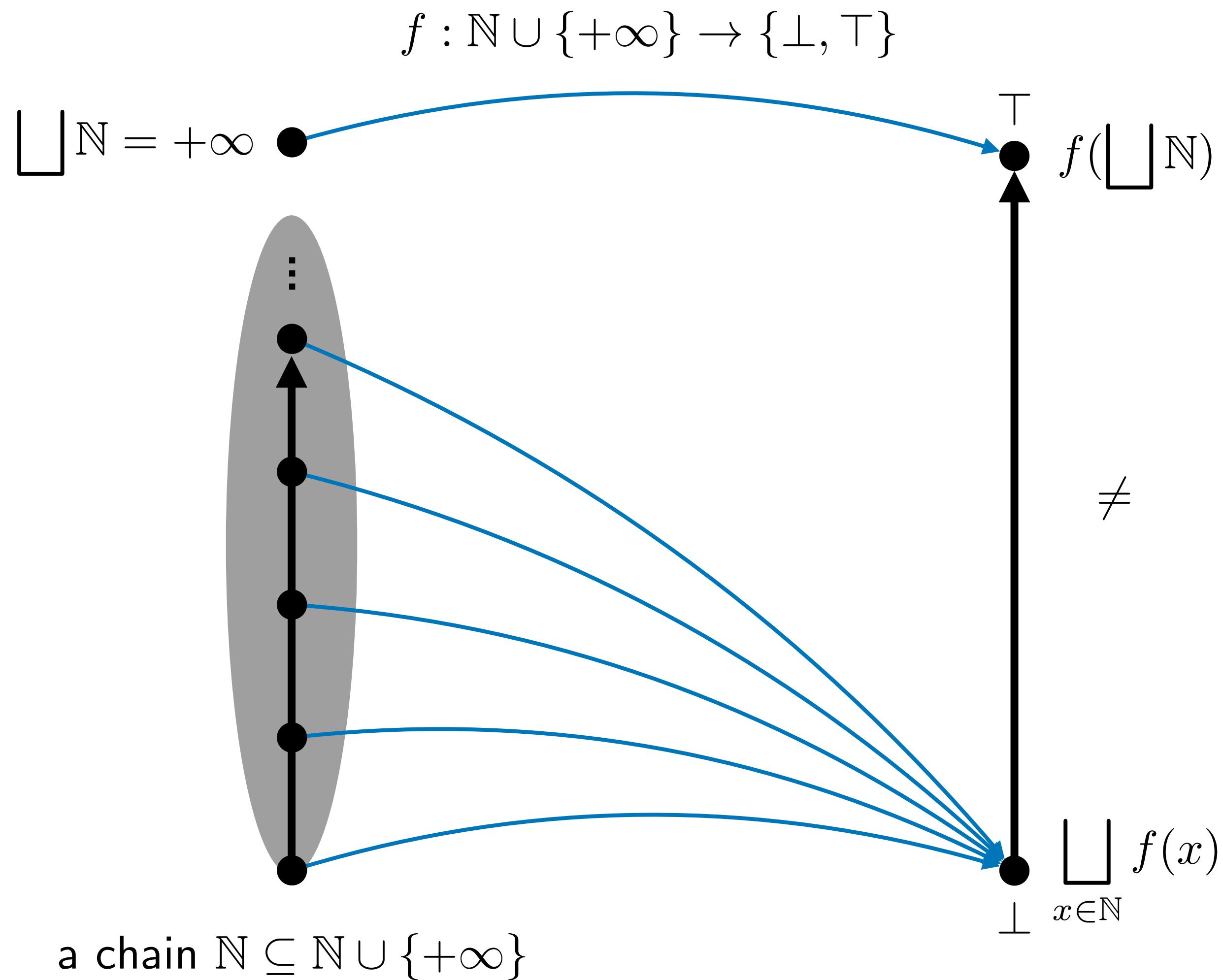
# Continuous Function



# Non-Continuous Function (1)



# Non-Continuous Function (2)



# Continuous Function

**Lemma.** If a function  $f$  is continuous,  $f$  is monotone.

*Proof.* Suppose that  $f$  is not monotone, i.e., for some  $a \sqsubseteq b$ ,  $f(a) \sqsupset f(b)$ . Then,

$$\begin{aligned} f(a) &\sqsubseteq f(a) \sqcup f(b) && (\text{by definition of } \sqcup) \\ &= f(a \sqcup b) && (\text{by continuity of } f) \\ &= f(b) && (\text{because } a \sqsubseteq b) \end{aligned}$$

This is a contradiction. Therefore,  $f(a) \sqsubseteq f(b)$ .

# Fixed Point

**Definition (Fixed Point).** Let  $(D, \sqsubseteq)$  be a partial ordered set. A **fixed point** of a function  $f : D \rightarrow D$  is an element  $x$  such that  $f(x) = x$ . We write  $\text{lfp } f$  for the **least fixed point** of  $f$  such that

$$f(\text{lfp } f) = \text{lfp } f \quad \text{and} \quad \forall d \in D. \ f(d) = d \implies \text{lfp } f \sqsubseteq d$$

**Theorem (Kleene Fixed Point).** Let  $f : D \rightarrow D$  be a continuous function on a CPO  $D$ . Then  $f$  has the **least fixed point**  $\text{lfp } f$  and

$$\text{lfp } f = \bigsqcup_{i \geq 0} f^i(\perp)$$

# Proof

$$\text{lfp } f = \bigsqcup_{i \geq 0} f^i(\perp)$$

- Plans: It is enough to show the following two things:
  - (1) There exists the chain  $\perp \sqsubseteq f(\perp) \sqsubseteq f^2(\perp) \sqsubseteq \dots$  and its least upper bound  $\bigsqcup_{i \geq 0} f^i(\perp)$  in  $D$
  - (2) The least upper bound  $\bigsqcup_{i \geq 0} f^i(\perp)$  is the least fixed point of  $f$

# Proof of (1)

(1) There exists the chain  $\perp \sqsubseteq f(\perp) \sqsubseteq f^2(\perp) \sqsubseteq \dots$  and its least upper bound  $\bigsqcup_{i \geq 0} f^i(\perp)$  in  $D$

**Proof.** We show by induction that  $\forall n \in \mathbb{N}. f^n(\perp) \sqsubseteq f^{n+1}(\perp)$ :

- $\perp \sqsubseteq f(\perp)$  ( $\perp$  is the least element of the CPO)
- $f^n(\perp) \sqsubseteq f^{n+1}(\perp) \implies f^{n+1}(\perp) \sqsubseteq f^{n+2}(\perp)$  (by monotonicity of  $f$ )

By definition of CPO, least upper bounds of all chains are also in the CPO. Therefore, the least upper bound  $\bigsqcup_{i \geq 0} f^i(\perp)$  of the above chain is in  $D$ .

# Proof of (2)

(2) The least upper bound  $\bigsqcup_{i \geq 0} f^i(\perp)$  is the least fixed point of  $f$

The proof consists of two parts:

(2-1)  $\bigsqcup_{i \geq 0} f^i(\perp)$  is a fixed point of  $f$

(2-2)  $\bigsqcup_{i \geq 0} f^i(\perp)$  is smaller than all the other fixed points

# Proof of (2-1)

(2-1)  $\bigsqcup_{i \geq 0} f^i(\perp)$  is a fixed point of  $f$

## Proof.

$$\begin{aligned} f\left(\bigsqcup_{n \geq 0} f^n(\perp)\right) &= \bigsqcup_{n \geq 0} f(f^n(\perp)) && (\text{by continuity of } f) \\ &= \bigsqcup_{n \geq 0} f^{n+1}(\perp) \\ &= \bigsqcup_{n \geq 0} f^n(\perp) \end{aligned}$$

## Proof of (2-2)

(2-2)  $\bigcup_{i \geq 0} f^i(\perp)$  is smaller than all the other fixed points

**Proof.** Suppose  $d$  is a fixed point, i.e.,  $d = f(d)$ . We show that any element  $f^i(\perp)$  is smaller than  $d$  by induction:

$$\forall n \in \mathbb{N}. f^n(\perp) \sqsubseteq d.$$

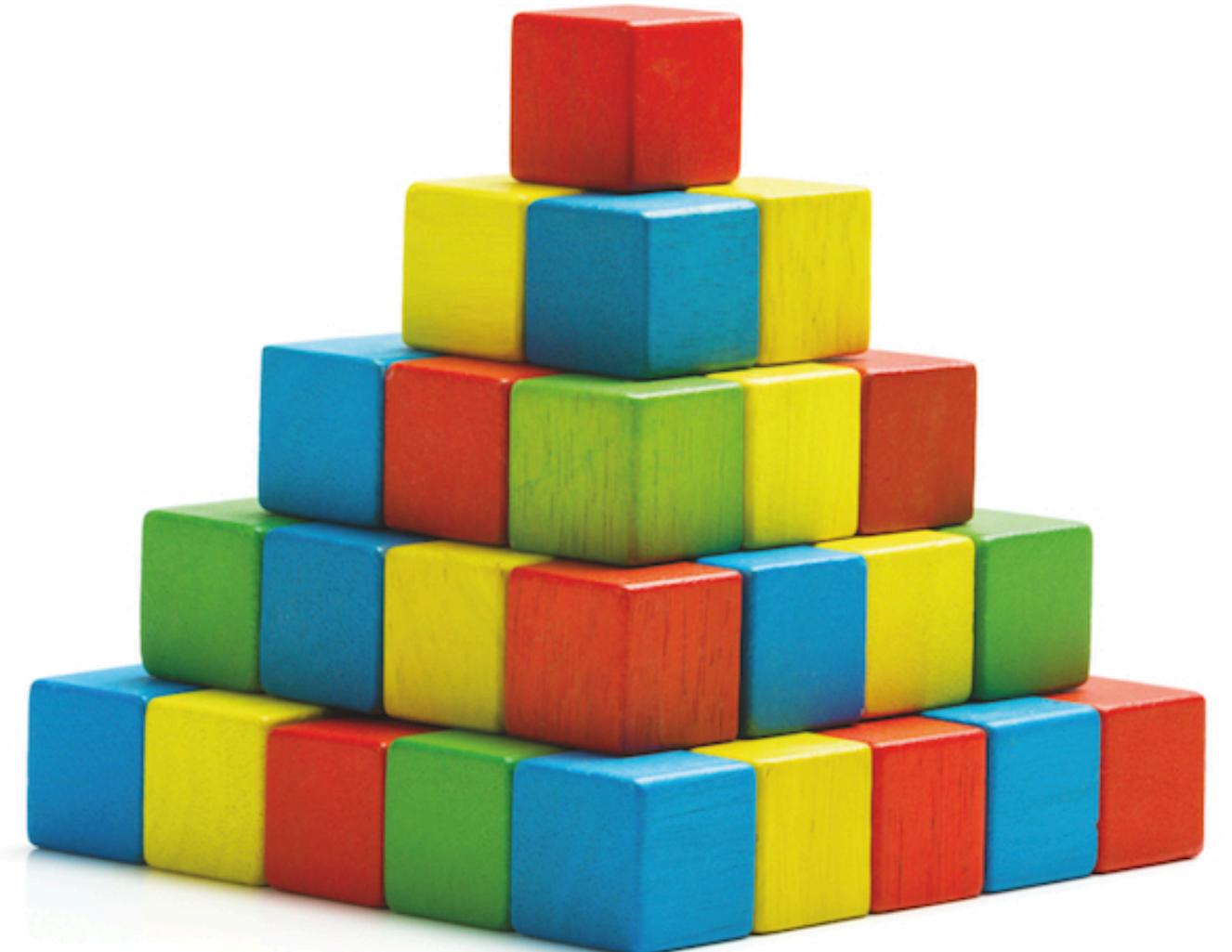
- $\perp \sqsubseteq d$  ( $\perp$  is the least element of the CPO)
- $f^n(\perp) \sqsubseteq d \implies f^{n+1}(\perp) \sqsubseteq f(d) = d$  (by monotonicity of  $f$ )

Because all the elements  $f^i(\perp)$  are smaller than  $d$ , their least upper bound  $\bigcup_{i \geq 0} f^i(\perp)$  is also smaller than  $d$ . Therefore

$$\bigcup_{i \geq 0} f^i(\perp) = \text{lfp } f$$

# 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 :  $D = D_1 \rightarrow D_2$



# Lifted Set

- $D = S_{\perp}$

For any set  $S$ , let  $D = S + \{\perp\}$  where  $\perp$  is an element not in  $S$ . Then  $(D, \sqsubseteq)$  is a CPO where

$$d \sqsubseteq d' \iff (d = d') \vee (d = \perp)$$

- Why CPO?

# Cartesian Product

- $D = D_1 \times D_2$

Given two CPOs  $(D_1, \sqsubseteq_1)$  and  $(D_2, \sqsubseteq_2)$ ,  $(D, \sqsubseteq)$  is a CPO where

$$D = D_1 \times D_2 = \{(d_1, d_2) \mid d_1 \in D_1 \wedge d_2 \in D_2\}$$

$$(d_1, d_2) \sqsubseteq (d'_1, d'_2) \iff (d_1 \sqsubseteq_1 d'_1) \wedge (d_2 \sqsubseteq_2 d'_2)$$

- Why CPO?

# Separated Sum

$$D = D_1 + D_2$$

Given two CPOs  $(D_1, \sqsubseteq_1)$  and  $(D_2, \sqsubseteq_2)$ ,  $(D, \sqsubseteq)$  is a CPO where

$$D = D_1 + D_2 = \{(d_1, 1) \mid d_1 \in D_1\} \cup \{(d_2, 2) \mid d_2 \in D_2\} \cup \{\perp\}$$

$$(d_1, 1) \sqsubseteq (d'_1, 1) \iff d_1 \sqsubseteq_1 d'_1$$

$$(d_2, 2) \sqsubseteq (d'_2, 2) \iff d_2 \sqsubseteq_2 d'_2$$

- Why CPO?

# Function

$$D = D_1 \rightarrow D_2$$

Given two CPOs  $(D_1, \sqsubseteq_1)$  and  $(D_2, \sqsubseteq_2)$ ,  $(D, \sqsubseteq)$  is a CPO where

$$D = D_1 \rightarrow D_2 = \{f \mid f : D_1 \rightarrow D_2 \text{ is a continuous function}\}$$

$$f \sqsubseteq f' \iff \forall d_1 \in D_1. f(d_1) \sqsubseteq_2 f'(d_1)$$

- Why CPO?

# Semantic Domains (Revisited)

- Domains of memories, variables, and values: CPOs

$$\mathbb{M} = \mathbb{X} \rightarrow \mathbb{V}$$

$$\mathbb{X} = Var_{\perp}$$

$$\mathbb{V} = \mathbb{Z}_{\perp} + \mathbb{B}_{\perp}$$

- Domain of commands: function CPO

$$[C] \in \mathbb{M} \rightarrow \mathbb{M}$$

- Domain of expressions: function CPO

$$[E] \in \mathbb{M} \rightarrow \mathbb{Z}_{\perp}$$

$$[B] \in \mathbb{M} \rightarrow \mathbb{B}_{\perp}$$

# Semantics of while (Revisited)

- Semantics of while is defined as the least fixed point as follows:  
(i.e., the lfp exists)
  - All the sets are CPOs
  - All the functions are continuous

$$\llbracket \text{while } B \text{ } C \rrbracket = \text{lfp} \mathcal{F}_{B,C}$$

$$\text{where } \mathcal{F}_{B,C}(X)(m) = \begin{cases} X(\llbracket C \rrbracket(m)) & \text{if } \llbracket B \rrbracket(m) = \text{true} \\ m & \text{if } \llbracket B \rrbracket(m) = \text{false} \end{cases}$$

# Example

- `while (x < 10) x := x + 1`

$$[\![\text{while } (\text{x} < 10) \text{ x} := \text{x} + 1]\!] = \lambda m. \begin{cases} [\![\text{while } (\text{x} < 10) \text{ x} := \text{x} + 1]\!]([\![\text{x} := \text{x} + 1]\!](m)) & \text{if } [\![\text{x} < 10]\!](m) = \text{true} \\ m & \text{if } [\![\text{x} < 10]\!](m) = \text{false} \end{cases}$$

$$[\![\text{while } (\text{x} < 10) \text{ x} := \text{x} + 1]\!] = \text{lfp} \mathcal{F} \text{ where } \mathcal{F}(X) = \lambda m. \begin{cases} X([\![\text{x} := \text{x} + 1]\!](m)) & \text{if } [\![\text{x} < 10]\!](m) = \text{true} \\ m & \text{if } [\![\text{x} < 10]\!](m) = \text{false} \end{cases}$$

$$\text{lfp} \mathcal{F} = \perp \sqcup \mathcal{F}(\perp) \sqcup \mathcal{F}^2(\perp) \sqcup \dots$$

# Example

$$\mathcal{F}(X) = \lambda m. \begin{cases} X(\llbracket x := x + 1 \rrbracket(m)) & \text{if } \llbracket x < 10 \rrbracket(m) = \text{true} \\ m & \text{if } \llbracket x < 10 \rrbracket(m) = \text{false} \end{cases}$$

$\perp$

0 iter

$$\mathcal{F}(\perp) = \lambda m. \begin{cases} \perp(\llbracket x := x + 1 \rrbracket(m)) & \text{if } \llbracket x < 10 \rrbracket(m) = \text{true} \\ m & \text{if } \llbracket x < 10 \rrbracket(m) = \text{false} \end{cases}$$

0, 1 iter

$$\mathcal{F}^2(\perp) = \lambda m. \begin{cases} \mathcal{F}(\perp)(\llbracket x := x + 1 \rrbracket(m)) & \text{if } \llbracket x < 10 \rrbracket(m) = \text{true} \\ m & \text{if } \llbracket x < 10 \rrbracket(m) = \text{false} \end{cases}$$

0,1,2 iter

$$\mathcal{F}^3(\perp) = \dots$$

$$= \lambda m. \begin{cases} \begin{cases} \perp(\llbracket x := x + 1 \rrbracket^2(m)) & \text{if } \llbracket x < 10 \rrbracket(\llbracket x := x + 1 \rrbracket(m)) = \text{true} \\ \llbracket x := x + 1 \rrbracket(m) & \text{if } \llbracket x < 10 \rrbracket(\llbracket x := x + 1 \rrbracket(m)) = \text{false} \end{cases} & \text{if } \llbracket x < 10 \rrbracket(m) = \text{true} \\ m & \text{if } \llbracket x < 10 \rrbracket(m) = \text{false} \end{cases}$$

# Semantics of Commands (Revisited)

$$\begin{aligned}\llbracket C \rrbracket &: \mathbb{M} \rightarrow \mathbb{M} \\ \llbracket \text{skip} \rrbracket &= \lambda m. m \\ \llbracket C_0 ; C_1 \rrbracket &= \lambda m. \llbracket C_1 \rrbracket(\llbracket C_0 \rrbracket(m)) \\ \llbracket x := E \rrbracket &= \lambda m. m\{x \mapsto \llbracket E \rrbracket(m)\} \\ \llbracket \text{input}(x) \rrbracket &= \lambda m. m\{x \mapsto n\} \\ \llbracket \text{if } B \text{ then } C_1 \text{ else } C_2 \rrbracket &= \lambda m. \begin{cases} \llbracket C_1 \rrbracket(m) & \text{if } \llbracket B \rrbracket(m) = \text{true} \\ \llbracket C_2 \rrbracket(m) & \text{if } \llbracket B \rrbracket(m) = \text{false} \end{cases} \\ \llbracket \text{while } B \text{ } C \rrbracket &= \text{lfp} \lambda X. \left( \lambda m. \begin{cases} X(\llbracket C \rrbracket(m)) & \text{if } \llbracket B \rrbracket(m) = \text{true} \\ m & \text{if } \llbracket B \rrbracket(m) = \text{false} \end{cases} \right)\end{aligned}$$

# Generality of Fixed Point

- Consider the following function  $H$  that has two components  $F$  and  $G$

$$H(x, y) = (F(x), G(y))$$

where  $F(x) = 1$

and  $G(y) = G(y - 1) \times y$

- The least fixed point of  $H$  :

$$\begin{aligned}\text{lfp}H &= (1, \text{lfp}G) \\ &= (\text{lfp}F, \text{lfp}G)\end{aligned}$$

# Semantics of Programs

- Generally, semantics of a program is defined as the least fixed point
- A program  $P$  consists of subcomponents  $C_1, \dots, C_n$

$$P = (C_1, \dots, C_i, \dots C_n)$$

- The semantics of  $P$  consists of that of  $C_1, \dots, C_n$

$$\begin{aligned}\llbracket P \rrbracket &= (\llbracket C_1 \rrbracket, \dots, \llbracket C_i \rrbracket, \dots \llbracket C_n \rrbracket) \\ &= (A_1, \dots, \dots \llbracket C_i \rrbracket \dots, \dots A_n) \\ &= \mathcal{F}(\llbracket C_1 \rrbracket, \dots, \llbracket C_i \rrbracket, \dots \llbracket C_n \rrbracket) \\ &= \text{lfp } \mathcal{F}\end{aligned}$$

# Summary

- Denotational semantics describes mathematical meaning of programs
  - semantics of a program is an element of a **CPO**:  $\llbracket P \rrbracket \in D$
  - semantics is the **least fixed point** of a **continuous** function:  $\mathcal{F} \in D \rightarrow D$
  - **compositionally** defined by the semantics of subcomponents
  - the least fixed point is the **least upper bound** of the following chain:

$$\begin{aligned}\llbracket P \rrbracket &= \mathcal{F}(\llbracket P \rrbracket) \\ &= \text{lfp } \mathcal{F} \\ &= \bigsqcup_{i \geq 0} \mathcal{F}^i(\perp)\end{aligned}$$