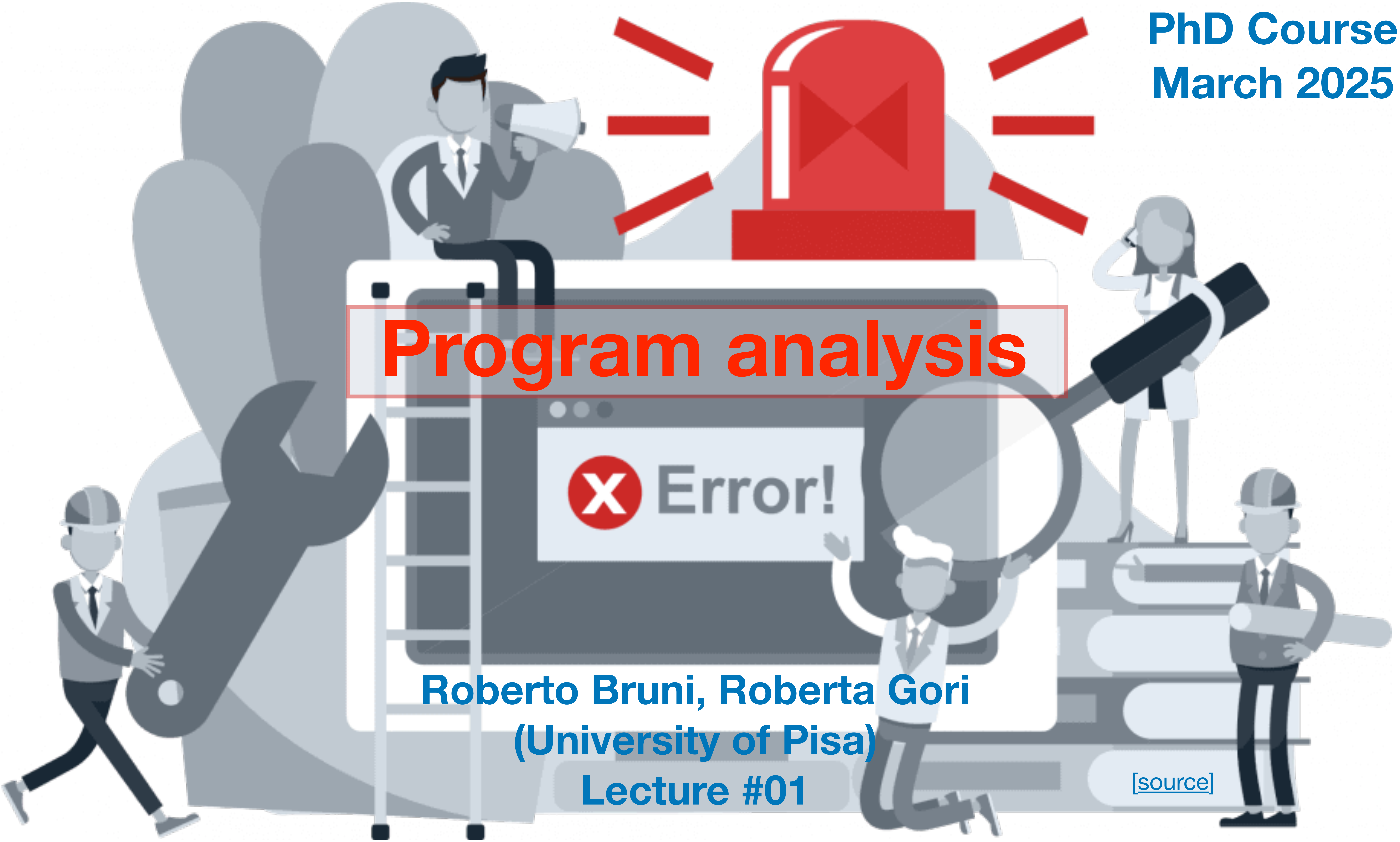
PhD Course
March 2025

Program analysis



Roberto Bruni, Roberta Gori
(University of Pisa)

Lecture #01

[source]

Lectures plan

Wednesday, March 12 11-13
Thursday, March 13 11-13

Wednesday, March 19 11-13
Thursday, March 20 11-13

Wednesday, March 26 11-13
Thursday, March 27 11-13

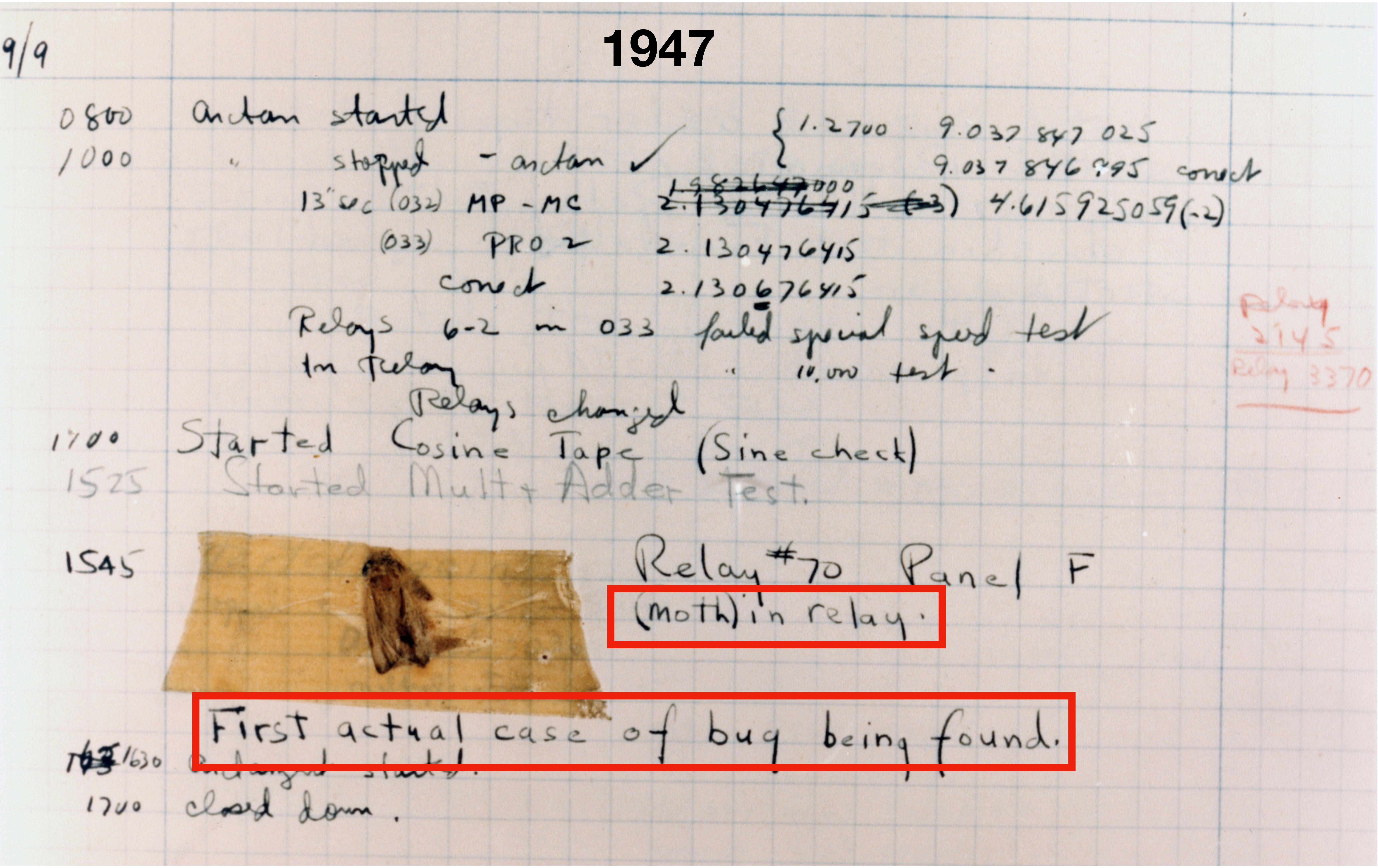
Wednesday, April 2 11-13
Thursday, April 3 11-13


Roberto Bruni


Azalea Raad


Roberta Gori

Bugs



1947

9/9

0800 Anctan started

1000 " stopped - anctan ✓ {1.2700 9.037 847 025

9.037 846 995 conct

13" sec (032) MP - MC ~~1.982147000~~ ~~2.130476415 (23)~~ 4.615925059(-2)

(033) PRO 2 2.130476415

conct 2.130676415

Relays 6-2 in 033 failed special speed test

In Relay " 10,000 test.

Relays changed

1100 Started Cosine Tape (Sine check)

1525 Started Mult+ Adder Test.

1545 Relay #70 Panel F

(moth) in relay.

First actual case of bug being found.

~~143~~ 1630 Antangent started.

1700 closed down.

Relay 2145

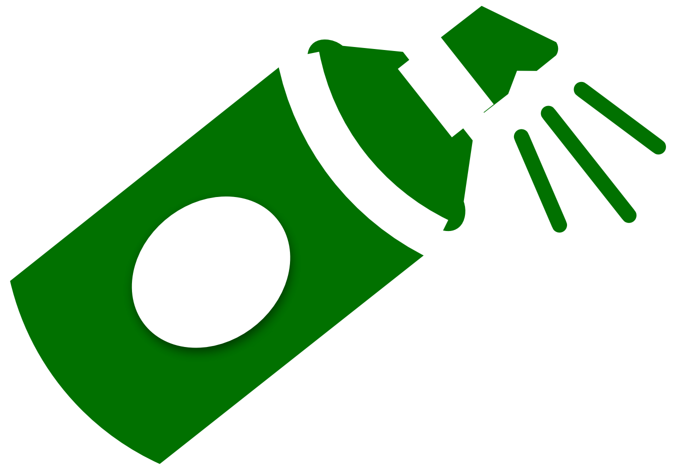
Relay 3370

WIKIPEDIA
The Free Encyclopedia

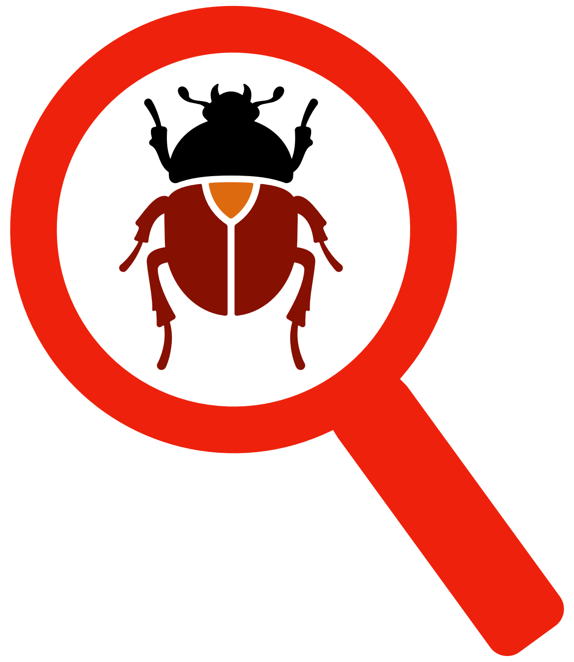
A **software bug** is an error, flaw or fault in the design, development, or operation of computer software that causes it to produce an incorrect or unexpected result

Software Verification

Correctness   the aim is to prove the absence of bugs



Incorrectness   the aim is to prove the presence of bugs



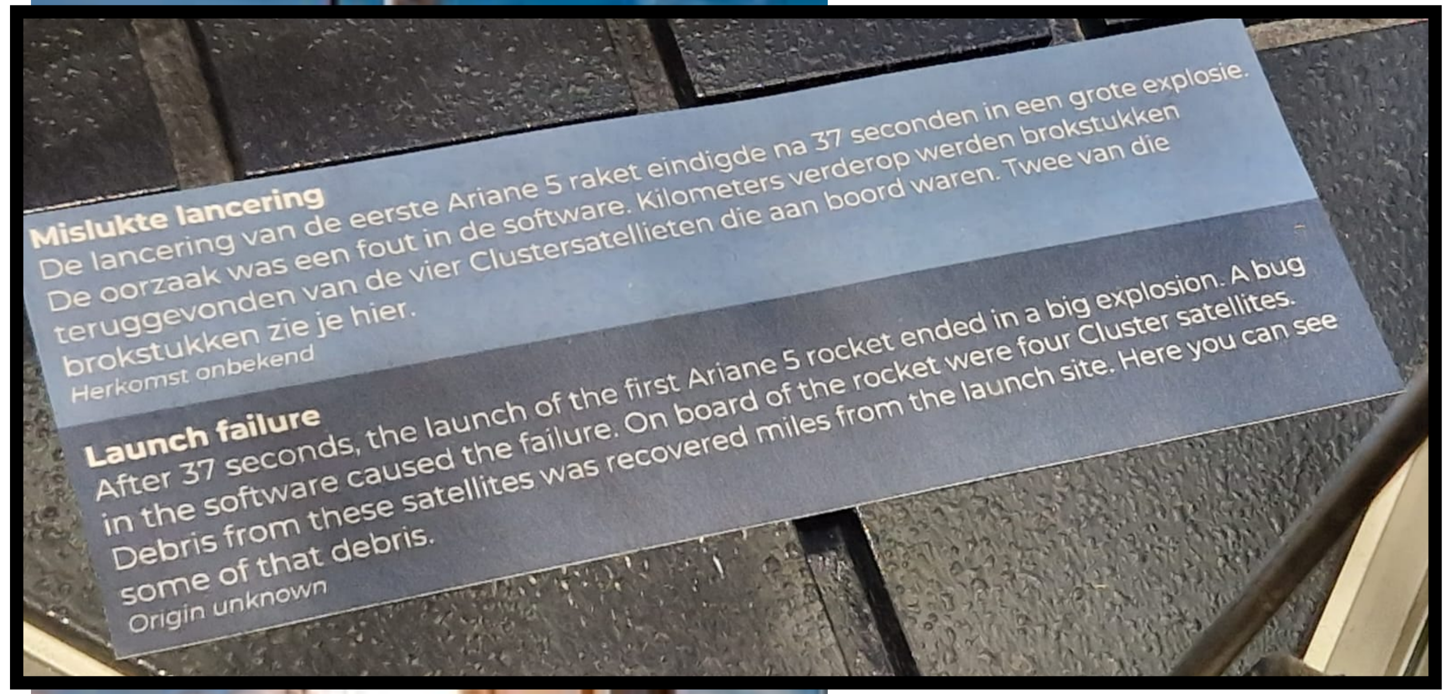
The need for verification

Friday, 24th June [1949]

Checking a large routine by Dr A. Turing.

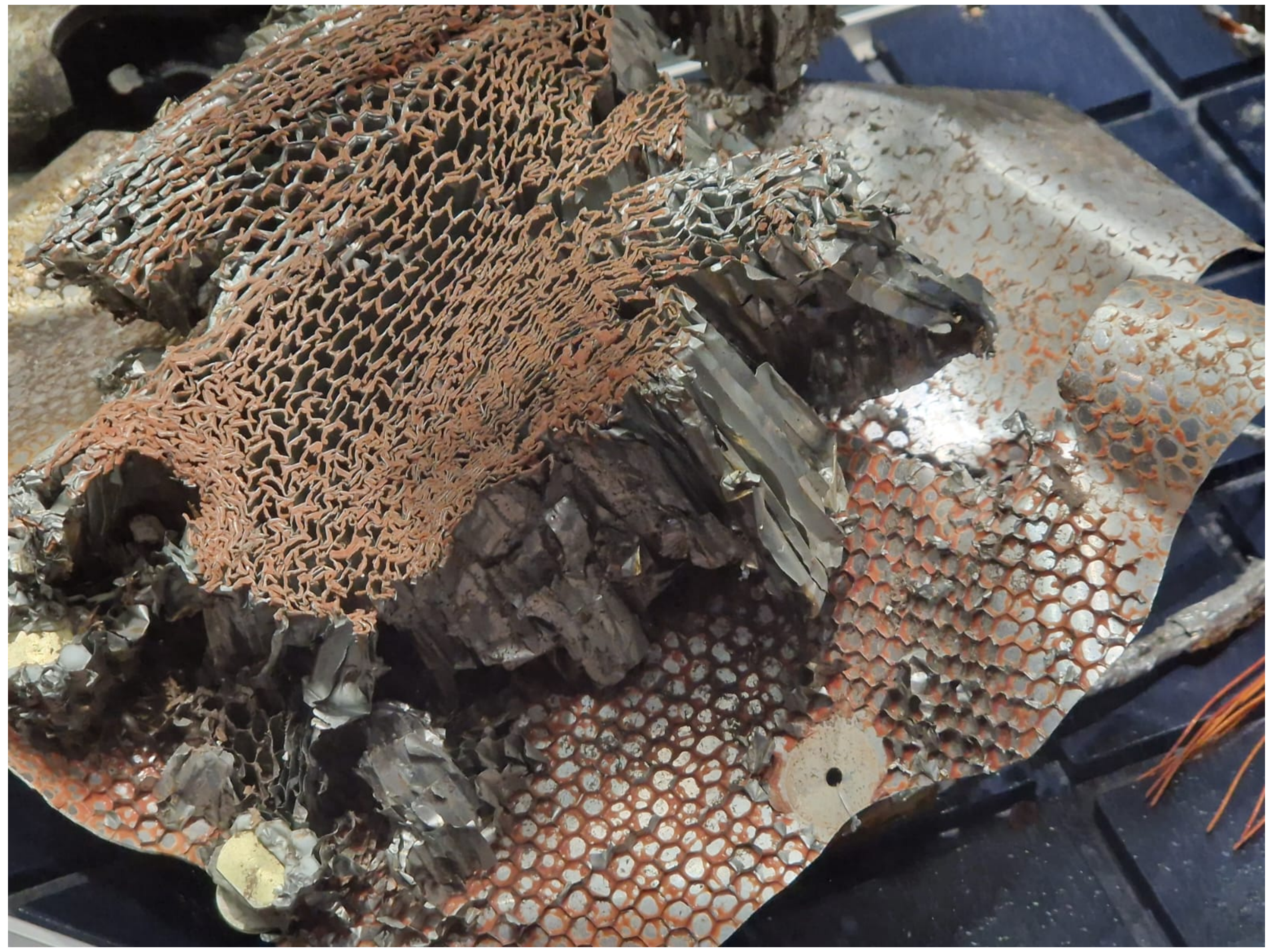
How can one check a routine in the sense of making sure that it is right?

Why do we need to verify our code?



Mislukte lancering
De lancering van de eerste Ariane 5 raket eindigde na 37 seconden in een grote explosie. De oorzaak was een fout in de software. Kilometers verderop werden brokstukken teruggevonden van de vier Clustersatellieten die aan boord waren. Twee van die brokstukken zie je hier.
Herkomst onbekend

Launch failure
After 37 seconds, the launch of the first Ariane 5 rocket ended in a big explosion. A bug in the software caused the failure. On board of the rocket were four Cluster satellites. Debris from these satellites was recovered miles from the launch site. Here you can see some of that debris.
Origin unknown



The code that exploded Ariane 5 rocket!

Ariane 5 Rocket Explosion (1996)

Caused due to numeric overflow error

Attempt to fit 64-bit format data into 16-bit space

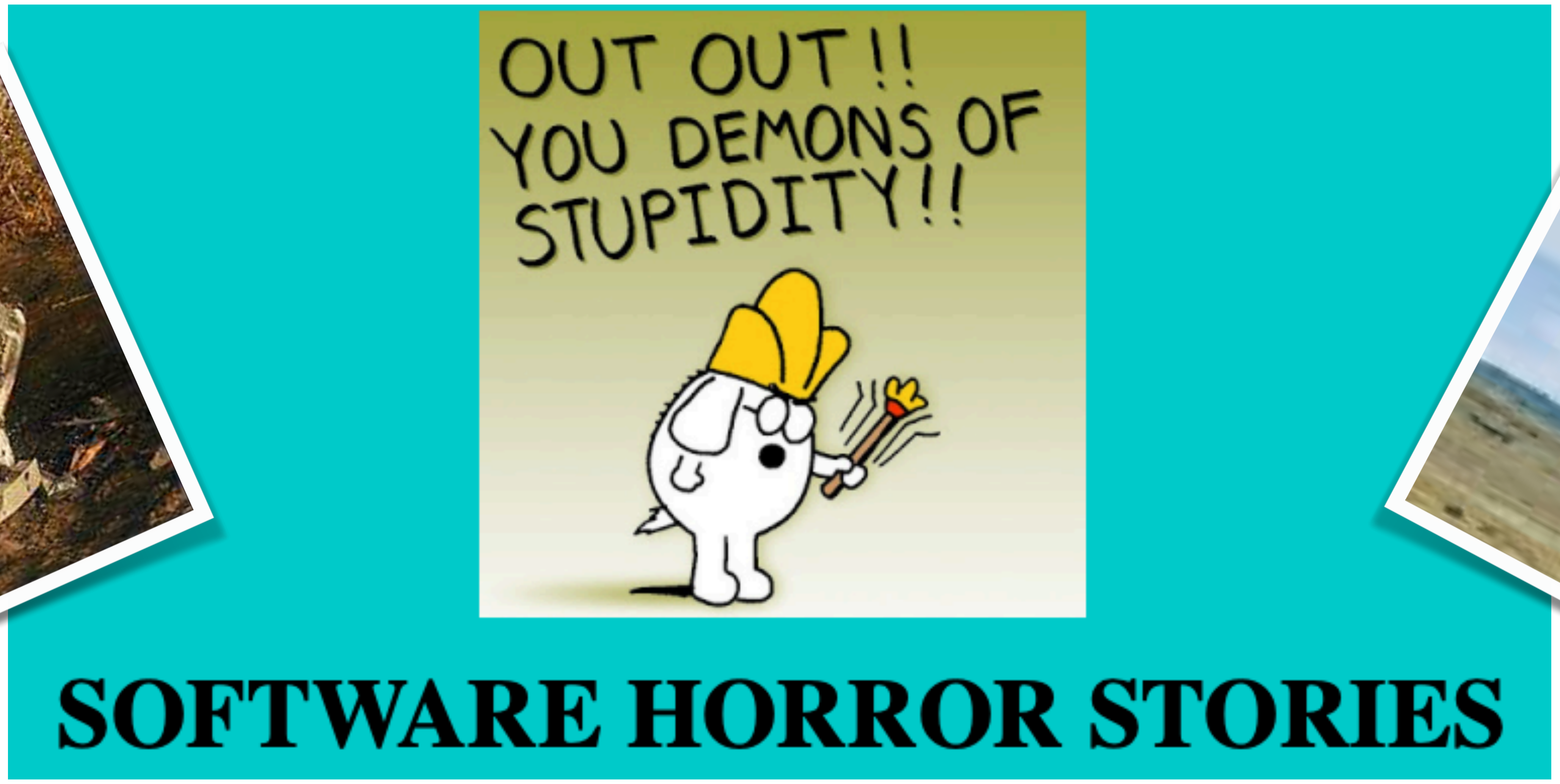
Cost: $100M for loss of mission

Multi-year set back to the Ariane program

Read more at:
https://www.bugsnag.com/blog/bug-day-ariane-5-disaster/

Unfortunately

It was one of the most serious but not the only one….



https://www.cs.tau.ac.il/~nachumd/horror.html

Toyota unintended acceleration
4 people died

Boeing 747 Max Crashes
350 people died

Costs of SW bugs

Knight Capital Trading Glitch (2012)
$ 440 M

Nissan Airbag Malfunction (2014)
1 Million Vehicles Recalled

Software Fails Watch (Tricentis, 2017): SW bugs lead to $ 1.7 Trillion revenue lost.

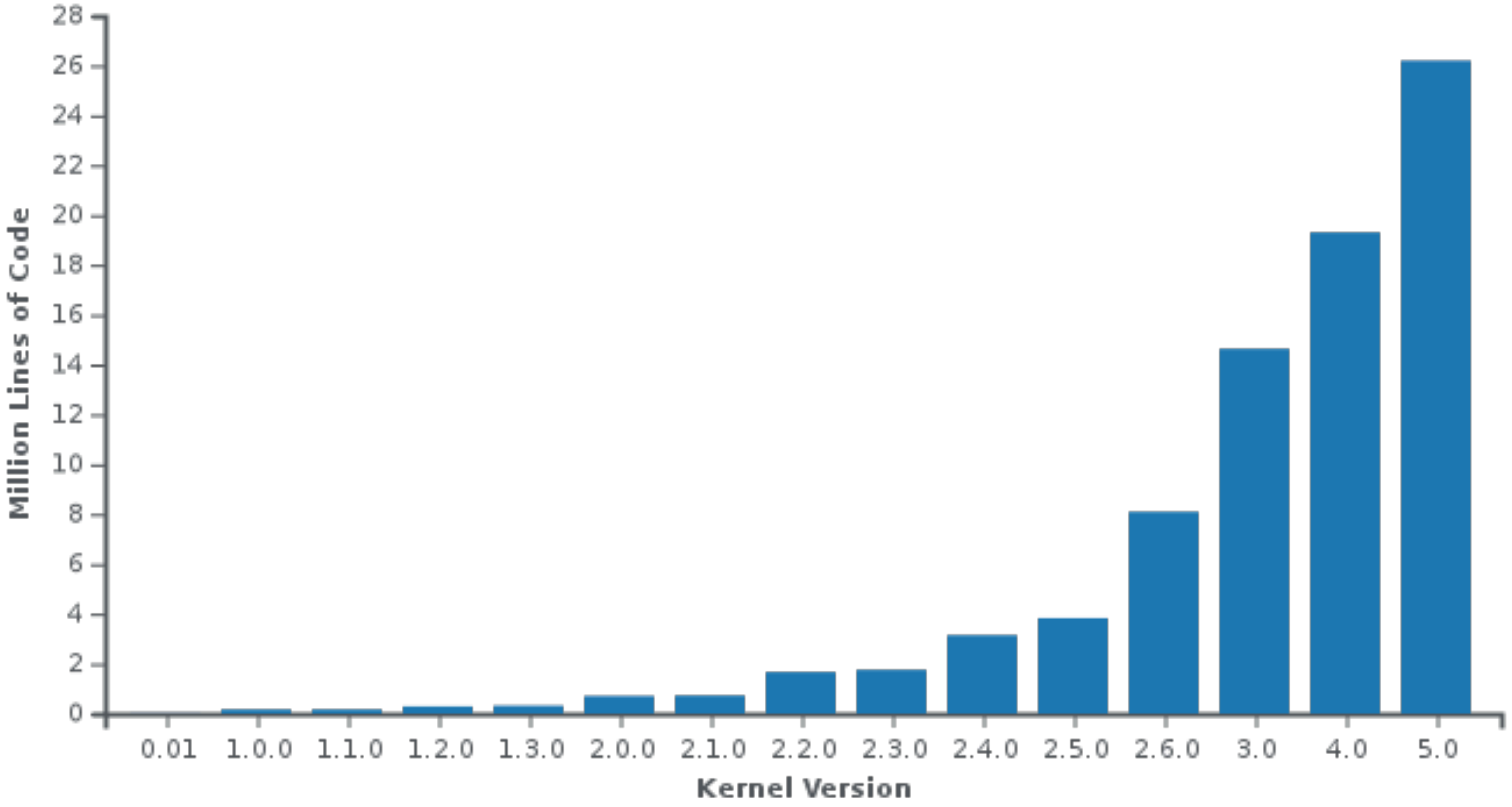
CISION PR Newswire (2020): SW bugs cost $ 61 Billion loss in productivity annually.

https://www.tricentis.com/news/tricentis-software-fail-watch-finds-3-6-billion-people-affected-and-1-7-trillion-revenue-lost-by-software-failures-last-year/

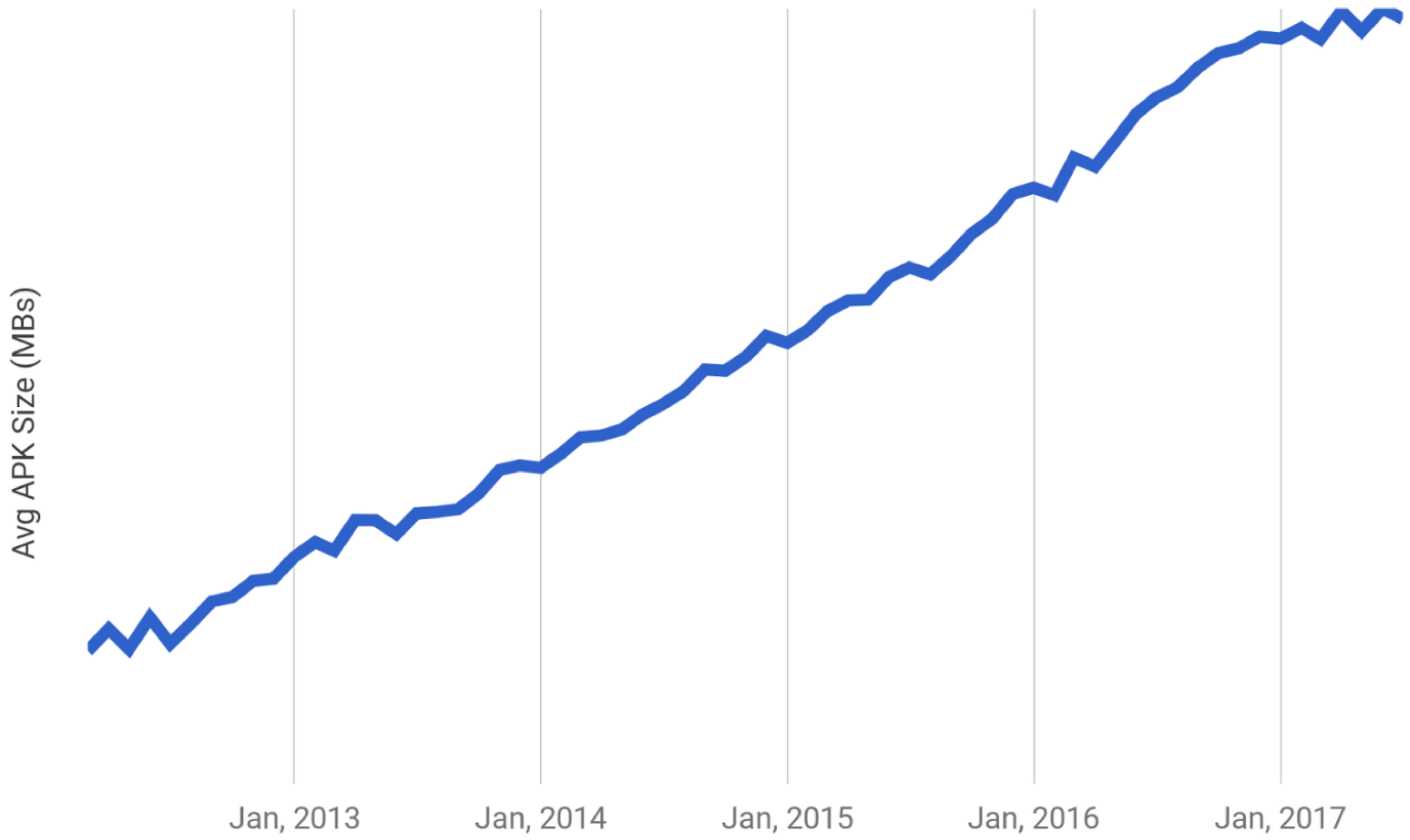
https://www.prnewswire.com/news-releases/study-software-failures-cost-the-enterprise-software-market-61b-annually-301066579.html

Complexity of programs

Size of Linux Kernel



Avg. Size of Android Apps



always increasing!

Success stories

A long time before success

Computer-assisted verification is an old idea

- Turing, 1948
- Floyd-Hoare logic, 1969

Success in practice: only from the mid-1990s

- Importance of the *increase of performance of computers*

A first success story:

- Paris metro line 14, using *Atelier B* (1998, refinement approach)

Other Famous Success Stories

- Flight control software of A380: *Astree* verifies absence of run-time errors (2005, abstract interpretation)
  `http://www.astree.ens.fr/`
- Microsoft's hypervisor: using Microsoft's *VCC* and the *Z3* automated prover (2008, deductive verification)
  `http://research.microsoft.com/en-us/projects/vcc/`
  More recently: verification of PikeOS
- Certified C compiler, developed using the *Coq* proof assistant (2009, correct-by-construction code generated by a proof assistant)
  `http://compcert.inria.fr/`
- L4.verified micro-kernel, using tools on top of *Isabelle/HOL* proof assistant (2010, Haskell prototype, C code, proof assistant)
  `http://www.ertos.nicta.com.au/research/l4.verified/`

The main question

Will our program behave as we intended?

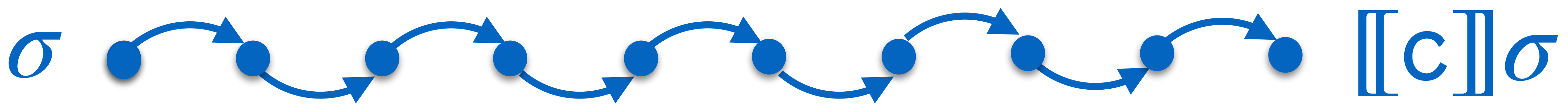
We need to analyse all executions of the program

The semantics of a program is a description of its run-time behaviors

Checking if a software will run as intended is equivalent to checking if the code satisfies a (semantic) property of interest

Forward semantics for deterministic programs

We start from input state σ and we want to characterise the reachable output states



A program

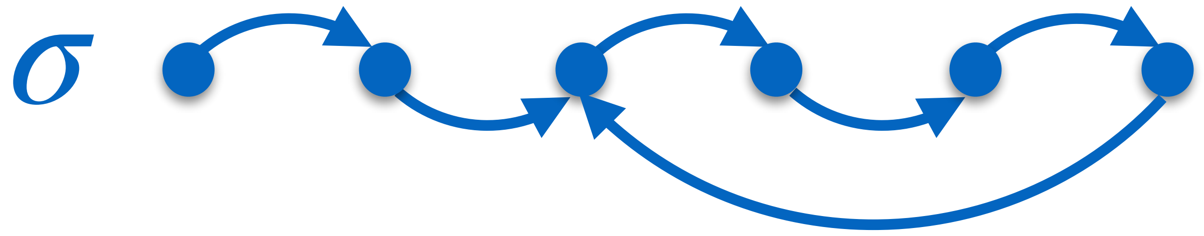
Input store

c

Output store

σ

$[\![\mathsf{c}]\!]\sigma$



σ

$[\![\mathsf{c}]\!]\sigma = \perp$

Non terminating execution

Denotational semantics

$[\![\mathsf{c}]\!] : \Sigma \to \Sigma_\perp$

Example

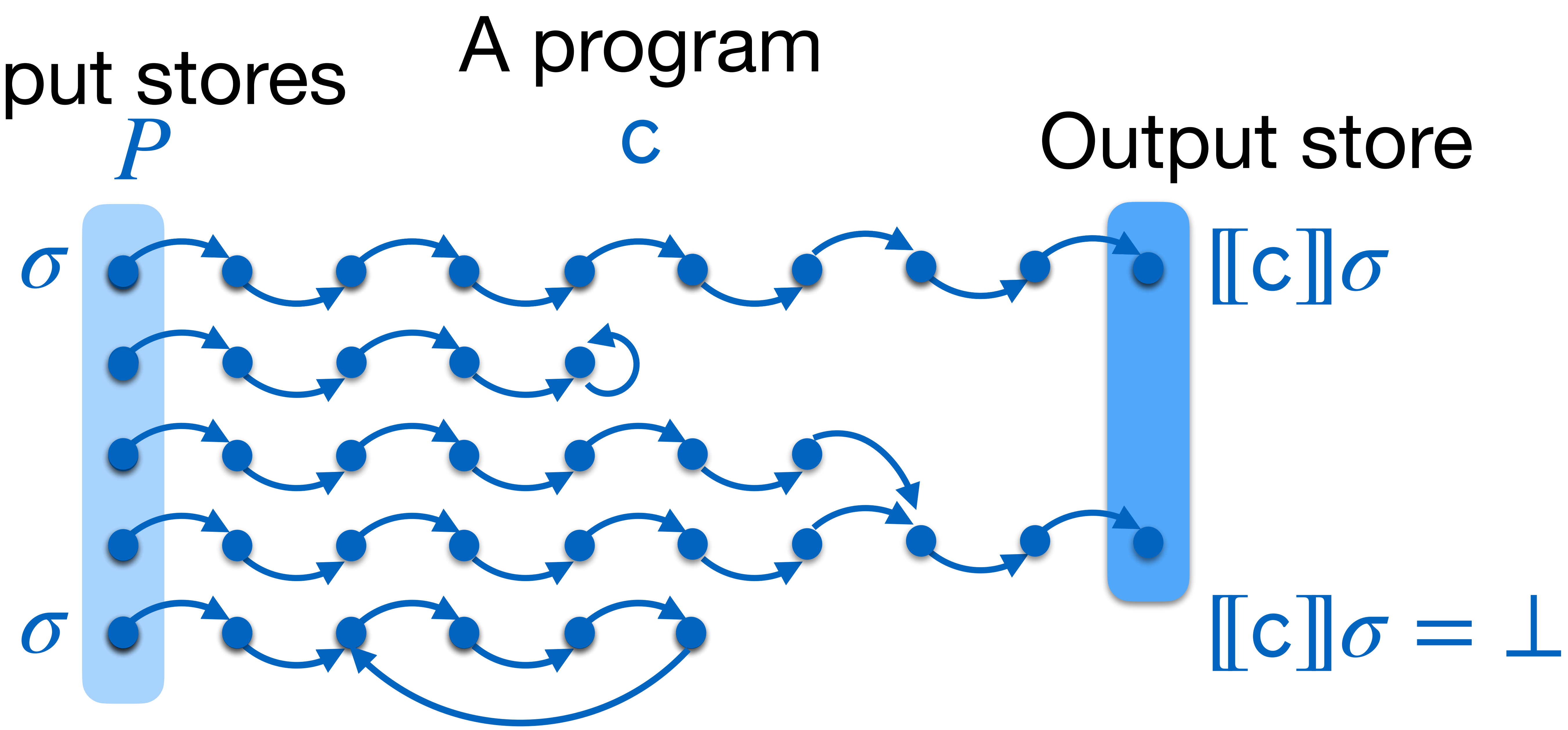
$c \triangleq$

```
while (n>1) {
    n := n+1;
    x := 0;
}
x := n-1;
```

$$[\![c]\!][\mathtt{n} \mapsto 1] = [\mathtt{n} \mapsto 1, \mathtt{x} \mapsto 0]$$

$$[\![c]\!][\mathtt{n} \mapsto 2] = \bot$$

Collecting semantics for deterministic programs



$$[\![c]\!]P = \bigcup_{\sigma \in P} [\![c]\!]\sigma$$

Denotational semantics $[\![c]\!] : \Sigma \to \Sigma_\bot$

Collecting semantics $[\![c]\!] : \wp(\Sigma) \to \wp(\Sigma)$

Example

$c \triangleq$

```
while (n>1) {
    n := n+1;
    x := 0;
}
x := n-1;
```

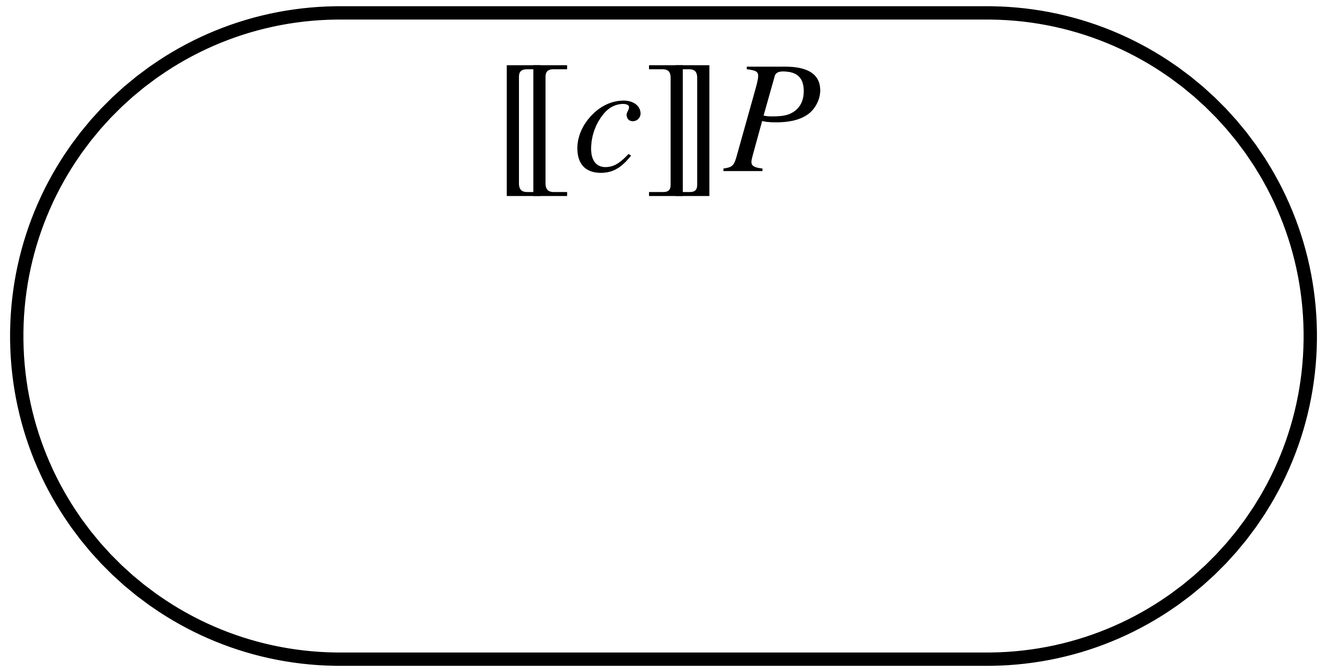
$$[\![c]\!](\mathrm{n} > 1) = \emptyset$$

$$[\![c]\!](\mathrm{n} > 0) = \{[\mathrm{n} \mapsto 1, \mathrm{x} \mapsto 0]\}$$

$$[\![c]\!](\mathrm{n} \geq 0) = \{[\mathrm{n} \mapsto 1, \mathrm{x} \mapsto 0], \\ [\mathrm{n} \mapsto 0, \mathrm{x} \mapsto -1]\}$$

Ideal exact analysis

$$[\![c]\!] : \wp(\Sigma) \to \wp(\Sigma)$$



🐞 $\in^{?} [\![c]\!]P$

semantic property of a program: a property about $[\![c]\!]$

$$\mathscr{P}(c) \equiv \forall P \,.\, \forall \sigma \in [\![c]\!]P \,.\, \sigma(x) \neq 0$$

Undecidability in the way

non trivial property:

- there exists a program c such that $\mathscr{P}(c)$ holds true
- and there exists also some program c such that $\mathscr{P}(c)$ is false

Rice theorem.

Let $\mathscr{P}(c)$ be a **non trivial** semantic property of programs c.

There exists no **algorithm** such that, **for every program** c,

it returns true **if and only if** $\mathscr{P}(c)$ holds true

no analysis method that is automatic, universal, exact !

algorithmic

for any program

no false positive/negative

For some program…

$$\mathscr{P}(c) \equiv \forall P \neq \emptyset \,.\, \exists \sigma \in [\![c]\!]P \,.\, \sigma(x) \neq 0$$

$c \triangleq$

```
x := 1;
```

✓

and for some other program...

$$\mathscr{P}(c) \equiv \forall P \neq \emptyset \,.\, \exists \sigma \in [\![c]\!]P \,.\, \sigma(x) \neq 0$$

$c \triangleq$

```
while (n>1) {
    n := n+1;
    x := 0;
}
x := n-1;
```

❌

WIKIPEDIA
The Free Encyclopedia

Collatz's conjecture

$$f(n) \triangleq \begin{cases} 1 & \text{if } n \leq 1 \\ f(n/2) & \text{if } n \neq 0 \wedge n\%2 = 0 \\ f(3n+1) & \text{otherwise} \end{cases}$$

$$\forall n, f(n) = 1$$

$$f(12) = f(6) = f(3) = f(10) = f(5) = f(16) = f(8) = f(4) = f(2) = f(1) = 1$$

The **Collatz conjecture**[a] is one of the most famous unsolved problems in mathematics. The conjecture asks whether repeating two simple arithmetic operations will eventually transform every positive integer into 1. It concerns sequences of integers in which each term is obtained from the previous term as follows: if a term is even, the next term is one half of it. If a term is odd, the next term is 3 times the previous term plus 1. The conjecture is that these sequences always reach 1, no matter which positive integer is chosen to start the sequence. The conjecture has been shown to hold for all positive integers up to 2.95×10^{20}, but no general proof has been found.

It is named after the mathematician Lothar Collatz, who introduced the idea in 1937, two years after receiving his doctorate.[4] The sequence of numbers involved is sometimes referred to as the **hailstone sequence**, **hailstone numbers** or **hailstone numerals** (because the values are usually subject to multiple descents and ascents like hailstones in a cloud),[5] or as **wondrous numbers**.[6]

Unsolved problem in mathematics:

?

- *For even numbers, divide by 2;*
- *For odd numbers, multiply by 3 and add 1.*

With enough repetition, do all positive integers converge to 1?

(more unsolved problems in mathematics)

but for Collatz's conjecture?

$$\mathscr{P}(c) \equiv \forall P \neq \emptyset \,.\, \exists \sigma \in [\![c]\!] P \,.\, \sigma(x) \neq 0$$

$c \triangleq$

```
while (x>1) {
    if (even(x)) { x := x/2; }
               else { x:= 3x+1; }
} % does it terminate for any value of x?
```

Limitations of the analysis

no analysis method that is automatic, universal, exact !

We need to give something up:

automation: machine-assisted techniques

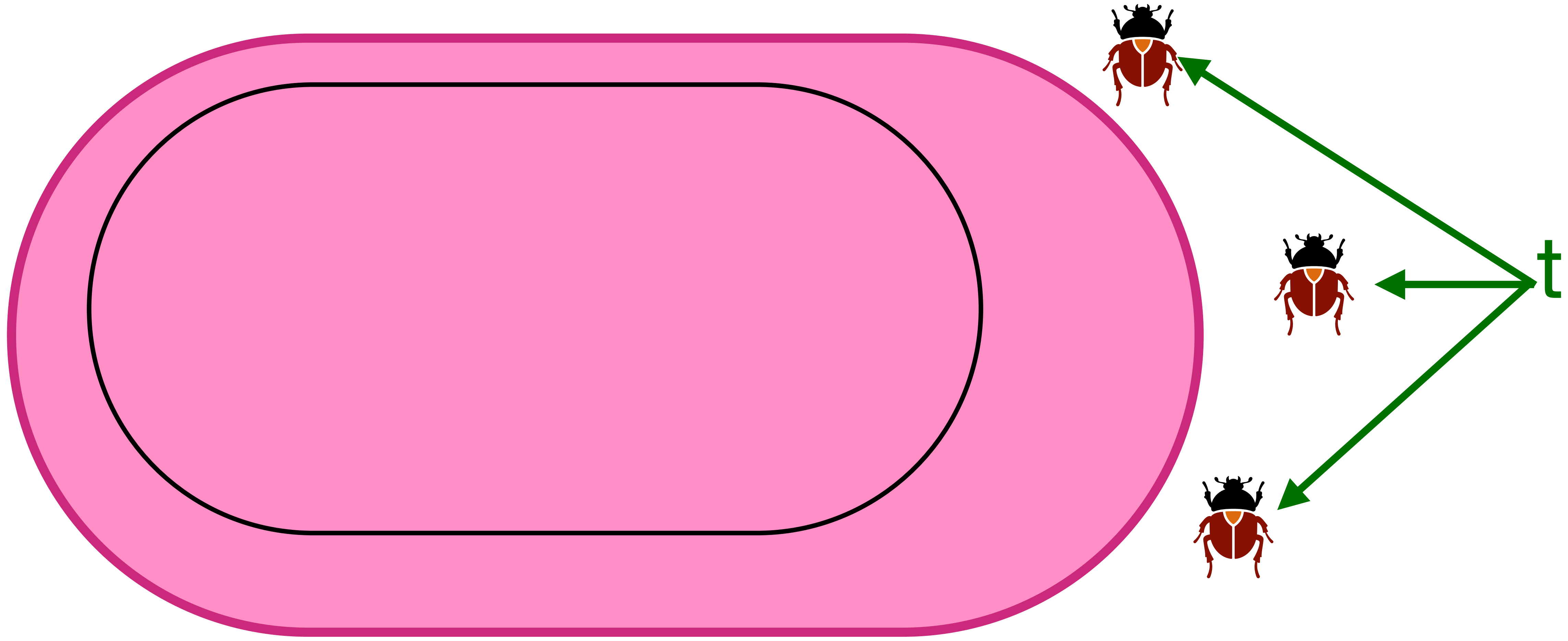
the **universality** “for all programs”:
targeting only a restricted class of programs

claim to find **exact** answers: introduce approximations


young
adult
old
Me
time
money
energy
time
money
energy
time
money
energy
time
money
energy

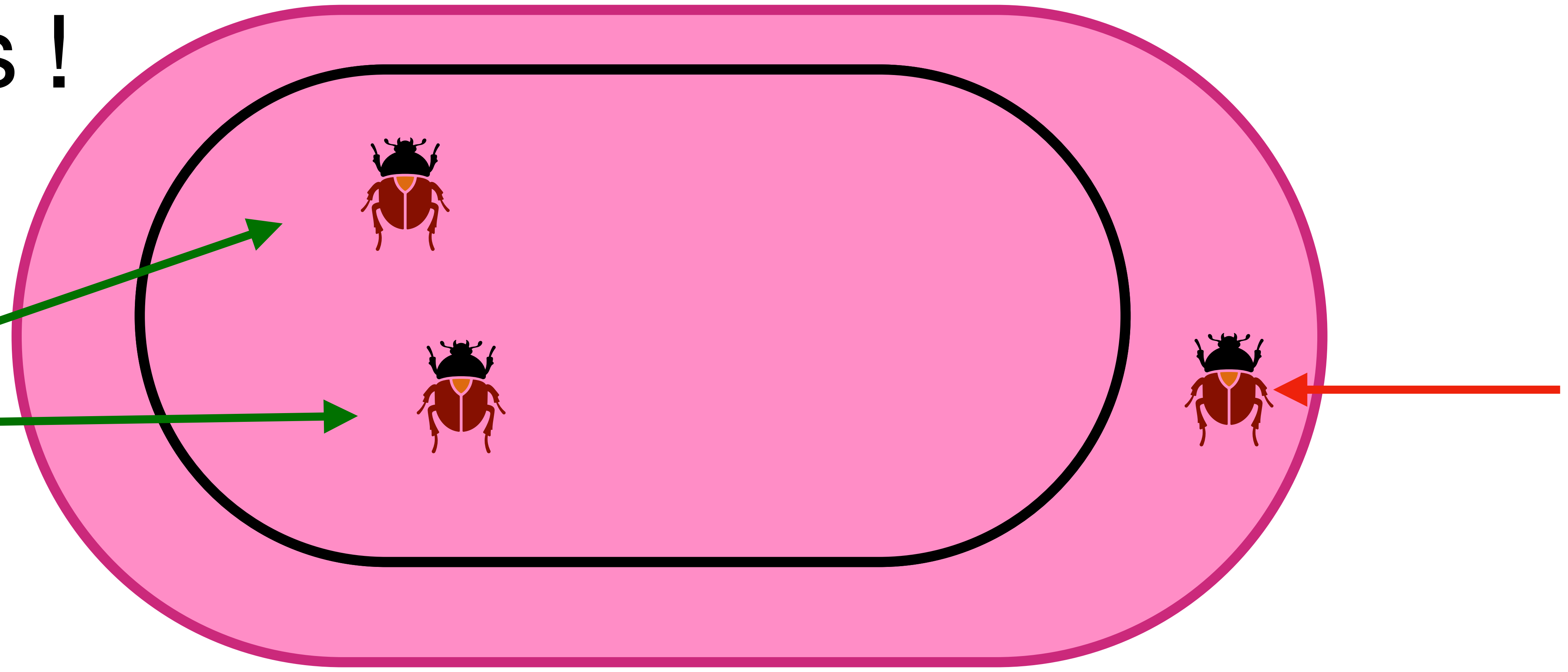
Over approximations

Good for proving correctness



true negative

Bad for bug-findings !



true positive

false positive

Example

$c \triangleq$

```
while (n>1) {
    n := n+1;
    x := 0;
}
x := n-1;
```

```
y := 1/(x-2);
```

Undefined behaviour for 🐞 x= 2

$$[\![c]\!](\mathtt{n} \geq 0) = \{[\mathtt{n} \mapsto 1, \mathtt{x} \mapsto 0], [\mathtt{n} \mapsto 0, \mathtt{x} \mapsto -1]\}$$

Correct

$$[\![c]\!]^{ov}(\mathtt{n} \geq 0) = \{\mathtt{n} \in \{0,1\}, \mathtt{x} \leq 0\}$$

🐞 $\notin [\![c]\!]^{ov}(\mathtt{n} \geq 0) \Longrightarrow$ 🐞 $\notin [\![c]\!](\mathtt{n} \geq 0)$

We can prove correctness!!

Example

$c \triangleq$

```
while (n>1) {
    n := n+1;
    x := 0;
}
x := n-1;

y := 1/(x+2);
```

Undefined behaviour for 🐞 x=-2

$$[\![c]\!](\mathtt{n} \geq 0) = \{[\mathtt{n} \mapsto 1, \mathtt{x} \mapsto 0], [\mathtt{n} \mapsto 0, \mathtt{x} \mapsto -1]\}$$

Correct

$$[\![c]\!]^{ov}(\mathtt{n} \geq 0) = \{\mathtt{n} \in \{0,1\}, \mathtt{x} \leq 0\}$$

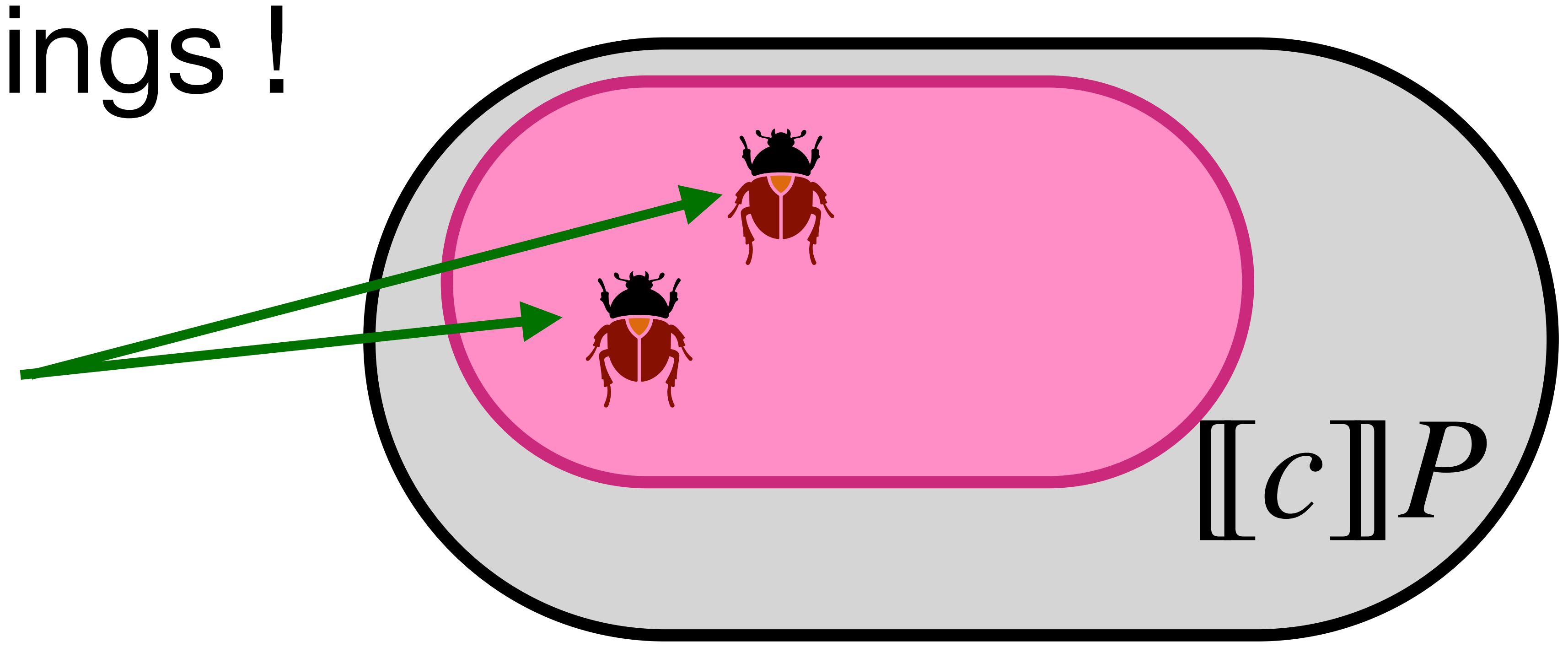
🐞 $\notin [\![c]\!](\mathtt{n} \geq 0)$

🐞 $\in [\![c]\!]^{ov}(\mathtt{n} \geq 0)$

False Positive

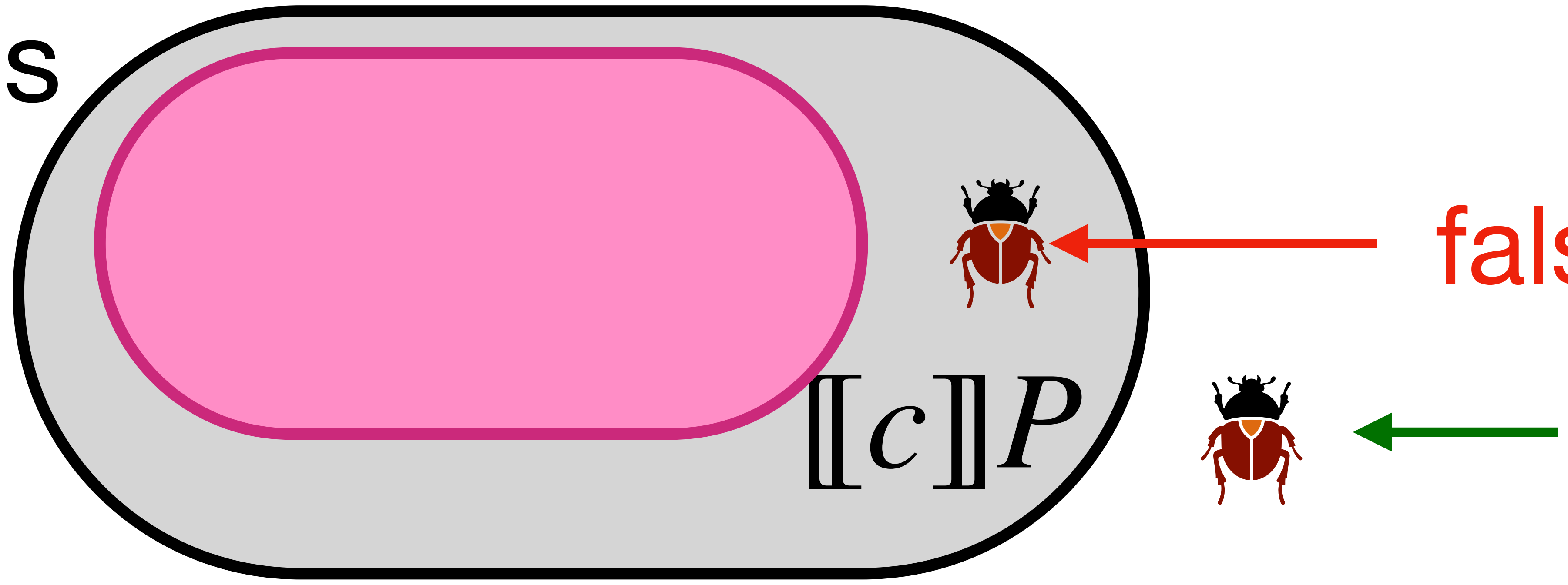
Under approximations

Good for bug-findings !



true positive

Bad for proving correctness



false negative

true negative

Example

$c \triangleq$

```
while (n>1) {
    n := n+1;
    x := 0;
}
x := n-1;
```

```
y := 1/(x);
```

Undefined behaviour for 🐞 x=0

$$[\![c]\!](\mathtt{n} \geq 0) = \{[\mathtt{n} \mapsto 1, \mathtt{x} \mapsto 0], [\mathtt{n} \mapsto 0, \mathtt{x} \mapsto -1]\}$$

Correct

$$[\![c]\!]^{un}(\mathtt{n} \geq 0) = \{[\mathtt{n} \mapsto 1, \mathtt{x} \mapsto 0]\}$$

🐞 $\in [\![c]\!]^{un}(\mathtt{n} \geq 0) \implies$ 🐞 $\in [\![c]\!](\mathtt{n} \geq 0)$

We can prove there is an error !!

Example

$c \triangleq$

```
while (n>1) {
    n := n+1;
    x := 0;
}
x := n-1;
```

```
y := 1/(x+1);
```

Undefined behaviour for 🐞 x=-1

$$[\![c]\!](\mathtt{n} \geq 0) = \{[\mathtt{n} \mapsto 1, \mathtt{x} \mapsto 0], [\mathtt{n} \mapsto 0, \mathtt{x} \mapsto -1]\}$$

Correct

$$[\![c]\!]^{un}(\mathtt{n} \geq 0) = \{[\mathtt{n} \mapsto 1, \mathtt{x} \mapsto 0]\}$$

🐞 $\notin [\![c]\!]^{un}(\mathtt{n} \geq 0)$

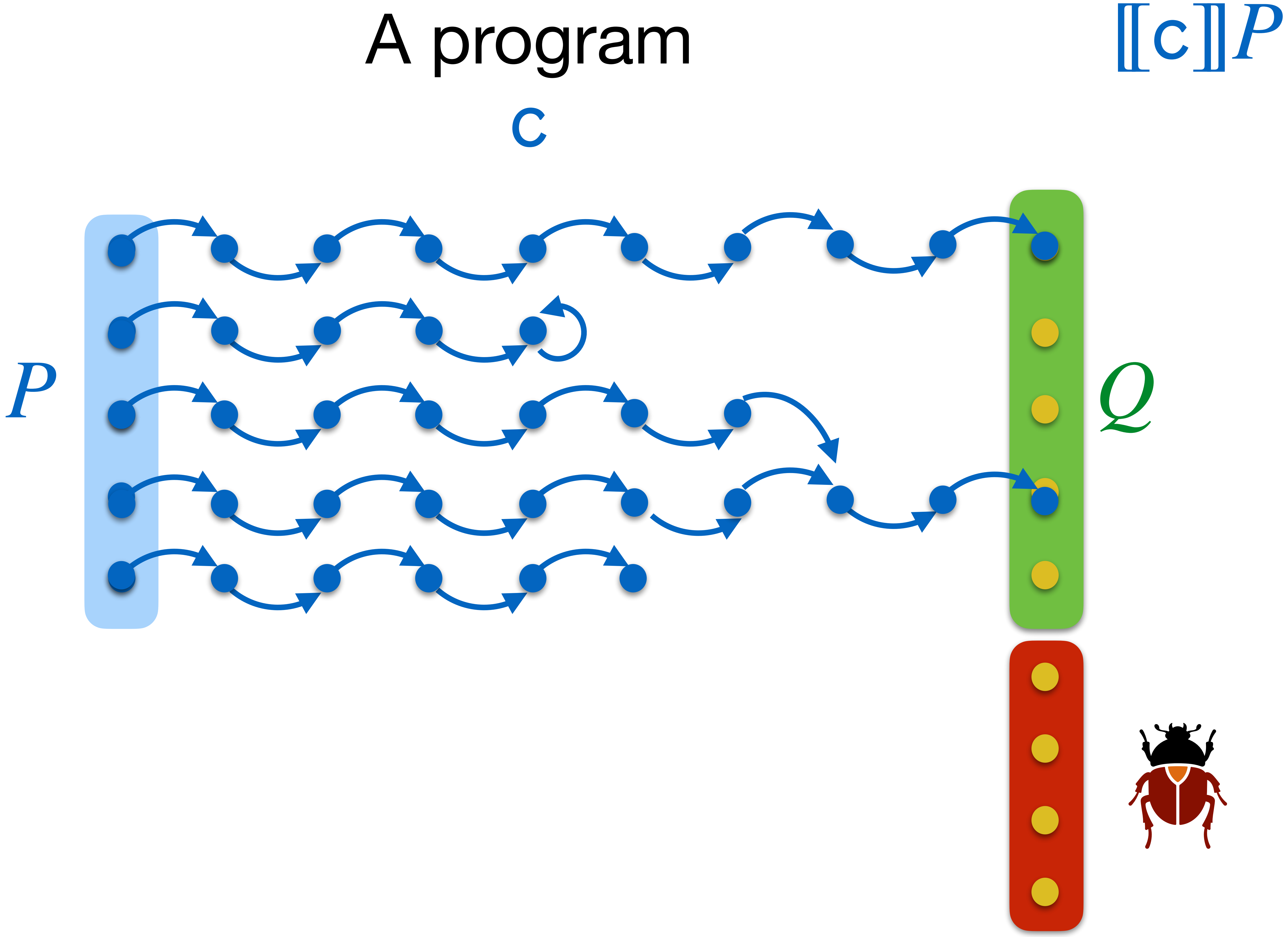
🐞 $\in [\![c]\!](\mathtt{n} \geq 0)$

False Negative

Proving Correctness: forward

✓

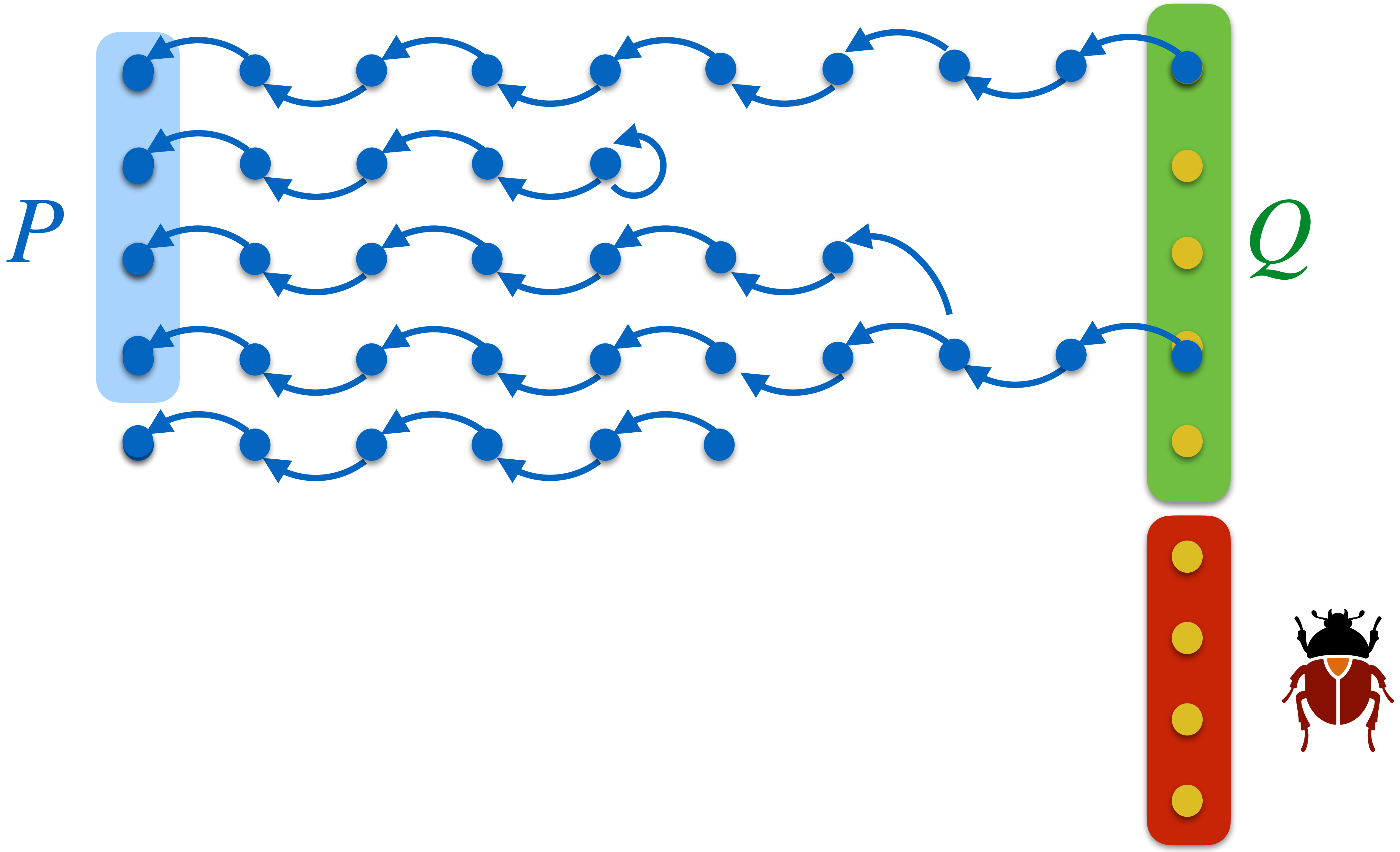
$[\![c]\!]P \subseteq Q$



$\forall \sigma \in P \, . \, [\![c]\!]\sigma$ either does not terminate or terminates in Q

Proving Correctness: backward

A program



✓

$P \subseteq wlp(c, Q)$

$[\![c]\!]P \subseteq Q$

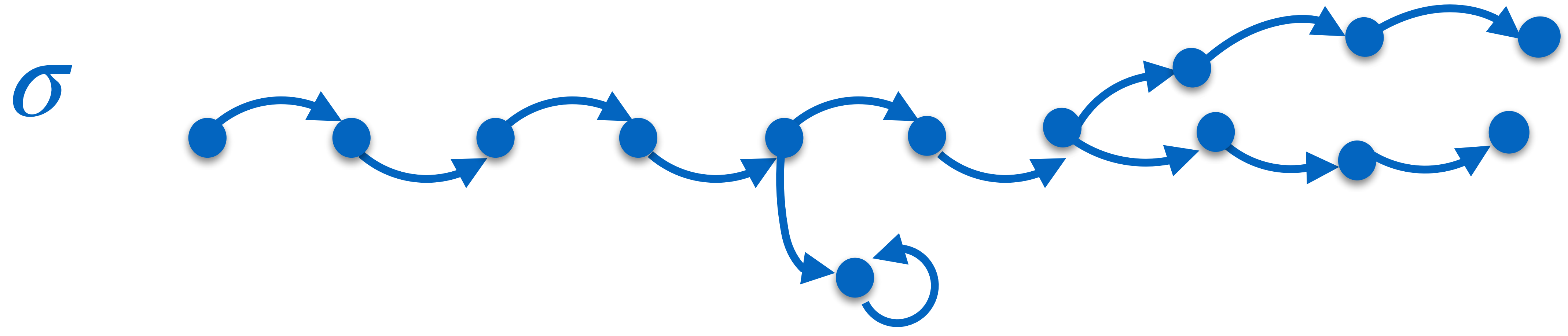
Dijkstra's weakest liberal precondition

$$wlp(c, Q) = \{\sigma \mid [\![c]\!]\{\sigma\} \subseteq Q\}$$

Nondeterministic programs

Some programs may exhibit nondeterministic behaviour
(lack of information, approximation, programming constructs $c_1 + c_2$)

A program c



$[\![c]\!] : \Sigma \to \wp(\Sigma)$

$[\![c]\!] P \subseteq Q$

$P \subseteq wlp(c, Q)$

all the outputs starting from $\sigma \in P$ either do non terminate or terminate in Q

Example

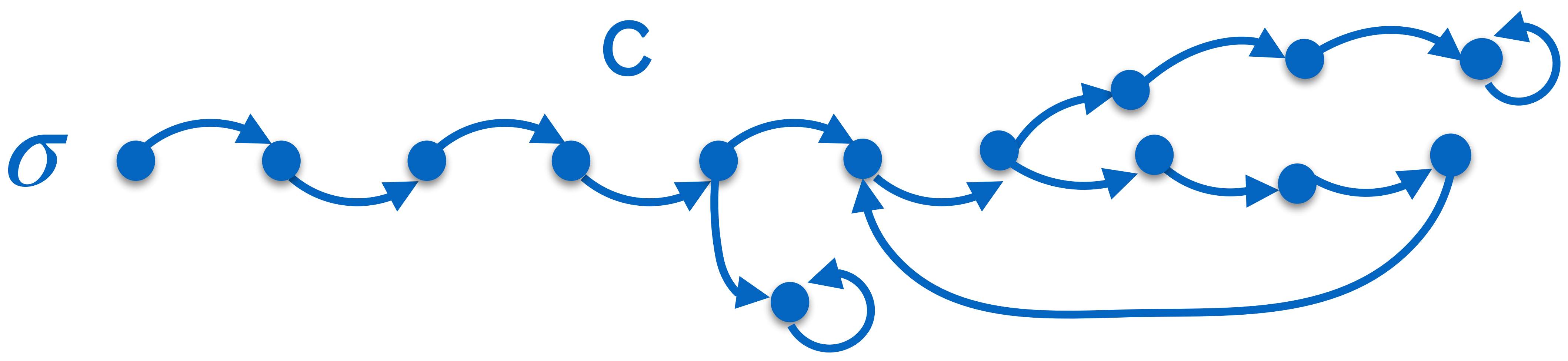
$c \triangleq$

```
Divisor_of(x)
{
…
s := nondet[2..x/2];
if (x%s=0)
    skip
else
    while true do skip
}
```

$$[\![c]\!][\mathrm{x} \mapsto 35] = \{\mathrm{x} = 35, \mathrm{s} \in \{5,7\}\}$$

An example: non-termination analysis

Given a program c and an input store σ does $[\![c]\!]\sigma = \varnothing$?



Using over-approximation: we try to prove $[\![c]\!]^{ov}\sigma \subseteq \varnothing$

Non termination

Using under-approximation: we try to prove $[\![c]\!]^{un}\sigma \supseteq Q$ for some $Q \neq \varnothing$

Termination

What we will see

	Forward	Backward	Over-approximation	Under-approximation
Hoare Logic (HL)	X		X	
Necessary Condition (NC)		X	X	
Separation Logic (SL)	X		X	
Incorrectness Logic (IL)	X			X
Incorrectness SL	X			X
UNTer	X	X		X
Sufficient Incorrectness Logic (SIL)		X		X
Separation SIL		X		X

Recap of fixpoint theory

A simple imperative language

command

integer variable

$$
\begin{array}{rll}
c ::= & x := a \\
| & \text{skip} \\
| & c_1 ; c_2 \\
| & \text{if } b \text{ then } c_1 \text{ else } c_2 \\
| & \text{while } b \text{ do } c
\end{array}
$$

arithmetic expression

Boolean expression

$$a ::= n \mid x \mid a_1 + a_2 \mid \ldots$$

$$b ::= a_1 \leq a_2 \mid b_1 \wedge b_2 \mid \ldots$$

Concrete domain

state

set of variables

$$\sigma : X \to \mathbb{Z}$$

set of integers

set of all states

$$\Sigma \triangleq \{\sigma : X \to \mathbb{Z}\}$$

concrete domain

$$\wp(\Sigma) \triangleq \{P \mid P \subseteq \Sigma\}$$

state property

Arithmetic expressions

$$[\![\cdot]\!] : \mathsf{Aexp} \to \Sigma \to \mathbb{Z}$$

$[\![a]\!]\sigma$

evaluates the arithmetic expression a in the current state σ

$$[\![\mathtt{n}]\!]\sigma \triangleq n$$
$$[\![\mathtt{x}]\!]\sigma \triangleq \sigma(\mathtt{x})$$
$$[\![a_0 \,\mathtt{op}\, a_1]\!]\sigma \triangleq [\![a_0]\!]\sigma \; op \; [\![a_1]\!]\sigma$$

Boolean expressions

$$[\![\cdot]\!] : \text{Bexp} \to \wp(\Sigma) \to \wp(\Sigma)$$

$[\![b]\!]P$ (intuitively $b \wedge P$)

is the set of all and only states in P that satisfy the condition b

$$[\![\texttt{true}]\!]P \triangleq P$$
$$[\![\texttt{false}]\!]P \triangleq \emptyset$$
$$[\![a_0 \,\texttt{cmp}\, a_1]\!]P \triangleq \{\, \sigma \in P \mid [\![a_0]\!]\sigma \; cmp \; [\![a_1]\!]\sigma \,\}$$
$$[\![b_0 \,\texttt{bop}\, b_1]\!]P \triangleq [\![b_0]\!]P \; bop \; [\![b_1]\!]P$$
$$[\![\texttt{not}\, b]\!]P \triangleq P \backslash ([\![b]\!]P)$$

Commands

$$[\![\cdot]\!] : \mathrm{Com} \to \wp(\Sigma) \to \wp(\Sigma)$$

$$[\![\texttt{skip}]\!]P \triangleq P$$

$$[\![x := a]\!]P \triangleq \{\ \sigma[n/x]\ \mid\ \sigma \in P \text{ and } n = [\![a]\!]\sigma\ \}$$

$$[\![c_0; c_1]\!]P \triangleq [\![c_1]\!]([\![c_0]\!]P)$$

$$[\![\text{if } b \text{ then } c_0 \text{ else } c_1]\!]P \triangleq [\![c_0]\!]([\![b]\!]P) \cup [\![c_1]\!]([\![\texttt{not}\ b]\!]P)$$

$$[\![\text{while } b \text{ do } c]\!]P \triangleq [\![\texttt{not}\ b]\!]P \cup [\![\text{while } b \text{ do } c]\!]([\![c]\!]([\![b]\!]P))$$

how do we know one solution exists? how do we know it is unique?

Fixpoint problem

The general problem

$$f : D \to D$$

a fixed point of f is $d \in D$ such that $d = f(d)$

let $F_f \triangleq \{d \in D \mid d = f(d)\} \subseteq D$

three questions:

- under which hypotheses $F_f \neq \varnothing$?
- if $F_f \neq \varnothing$, can we select a preferred element $\mathit{fix}(f) \in F_f$?
- and can we compute $\mathit{fix}(f)$?

Example

$D = \mathbb{N}$	F_f	$\mathit{fix}(f)$
$f(n) \triangleq n + 1$	$\varnothing$	
$f(n) \triangleq n/2$	$\{0\}$	0
$f(n) \triangleq n^2 - 5n + 8$	$\{2, 4\}$	2
$f(n) \triangleq n\%5$	$\{0, 1, 2, 3, 4\}$	0

Example

$D = \wp(\mathbb{N})$	F_f	$\mathit{fix}(f)$
$f(S) \triangleq S \cap \{1\}$	$\{\, \varnothing\,,\, \{1\}\, \}$	$\varnothing$
$f(S) \triangleq \mathbb{N} \setminus S$	$\varnothing$	
$f(S) \triangleq S \cup \{1\}$	$\{T \mid 1 \in T\}$	$\{1\}$
$f(S) \triangleq \{n \mid \exists m \in S, n \leq m\}$	$\{[0,k] \mid k \in \mathbb{N}\} \cup \{\varnothing, \mathbb{N}\}$	$\varnothing$

Ingredients

a partial order (to compare elements)

order preserving functions

iterative approximations

a base case

a limit solution

Complete partial orders

Partially ordered set (Poset or just PO)

a set

$(P, \sqsubseteq)$ a binary relation $\sqsubseteq \subseteq P \times P$

reflexive	$\forall p \in P.$	$p \sqsubseteq p$
antisymmetric	$\forall p, q \in P.$	$p \sqsubseteq q \ \wedge \ q \sqsubseteq p \ \Rightarrow \ p = q$
transitive	$\forall p, q, r \in P.$	$p \sqsubseteq q \ \wedge \ q \sqsubseteq r \ \Rightarrow \ p \sqsubseteq r$

$p \sqsubseteq q$ means that p and q are **comparable** and that p is less than (or equal to) q

Least element

$(P, \sqsubseteq)$ PO $\quad Q \subseteq P \quad \ell \in Q$

ℓ is a least element of Q if $\forall q \in Q.\ \ell \sqsubseteq q$

TH. (uniqueness of least element)

$(P, \sqsubseteq)$ PO $\quad Q \subseteq P \quad \ell_1, \ell_2$ least elements of Q implies $\ell_1 = \ell_2$

$$\left.\begin{array}{lll} \ell_1 \text{ least element of } Q & \Rightarrow & \ell_1 \sqsubseteq \ell_2 \\ \ell_2 \text{ least element of } Q & \Rightarrow & \ell_2 \sqsubseteq \ell_1 \end{array}\right\} \Rightarrow \ell_1 = \ell_2$$

by antisymmetry

Bottom

$(P, \sqsubseteq)$ PO

the least element of P
(if it exists) is called **bottom** and denoted $\bot$

sometimes written $\bot_P$

Examples

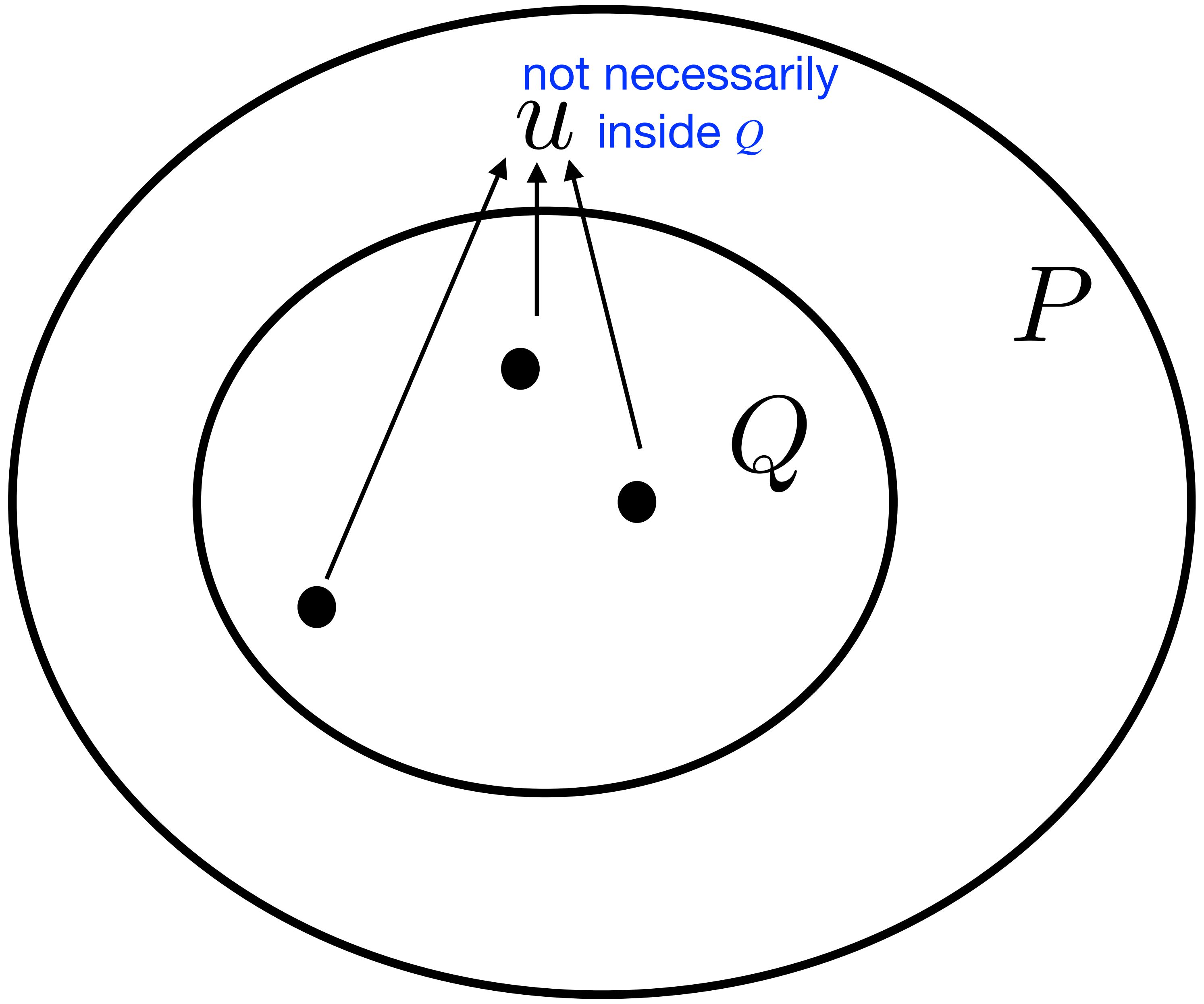
PO	bottom?
$(\mathbb{N} \cup \{\infty\}, \leq)$	0
$(\wp(S), \subseteq)$	$\varnothing$
$(\mathbb{Z}, \leq)$	❌
$(\mathbb{Z} \cup \{-\infty, \infty\}, \leq)$	$-\infty$

Upper bound

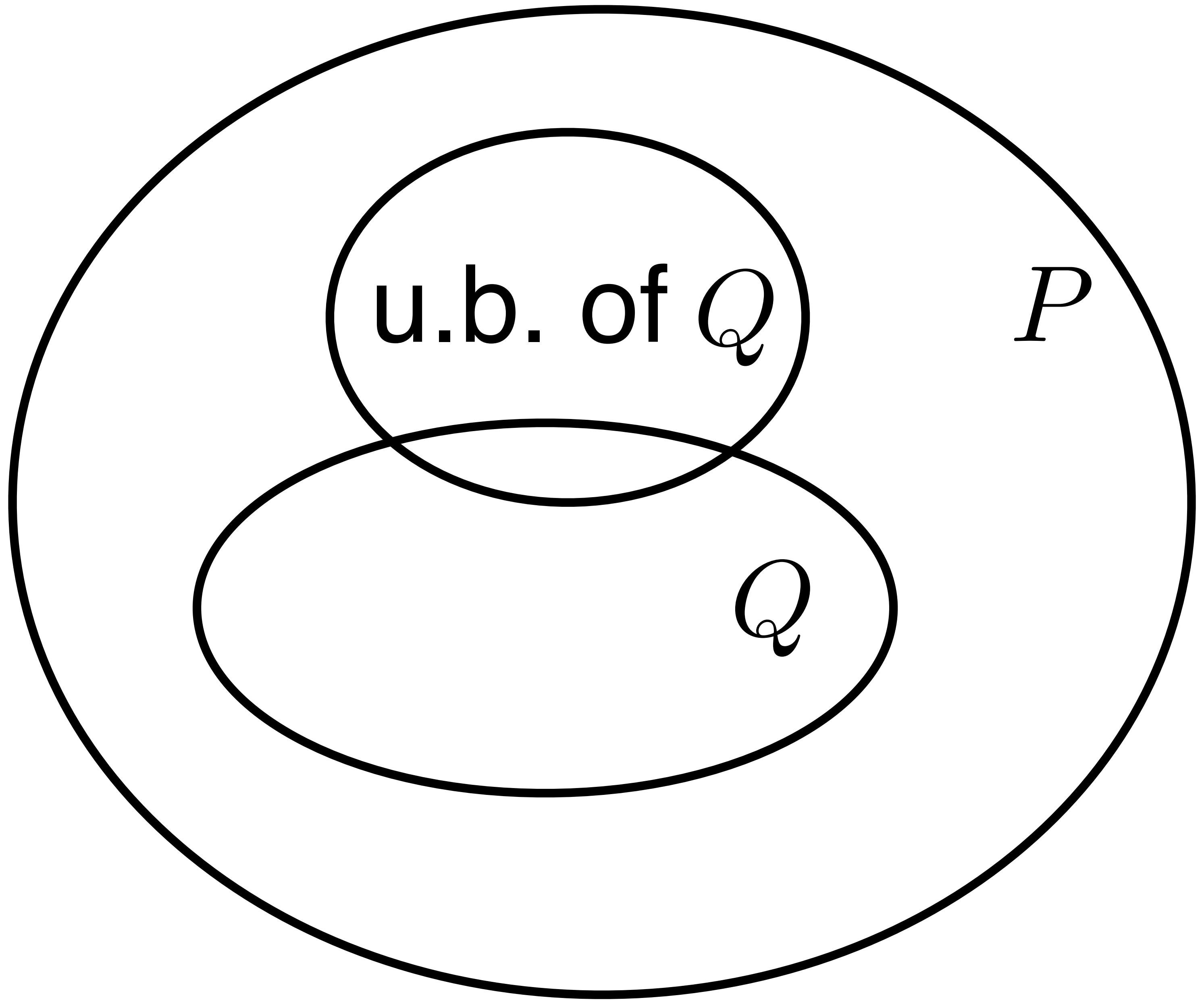
$(P, \sqsubseteq)$ PO $\quad Q \subseteq P \quad u \in P$

u is an upper bound of Q if $\forall q \in Q.\ q \sqsubseteq u$

(all the elements of Q are smaller than u)



Q may have many upper bounds



Least upper bound

$(P, \sqsubseteq)$ PO $\quad Q \subseteq P \quad p \in P$

p is the least upper bound (lub) of Q if

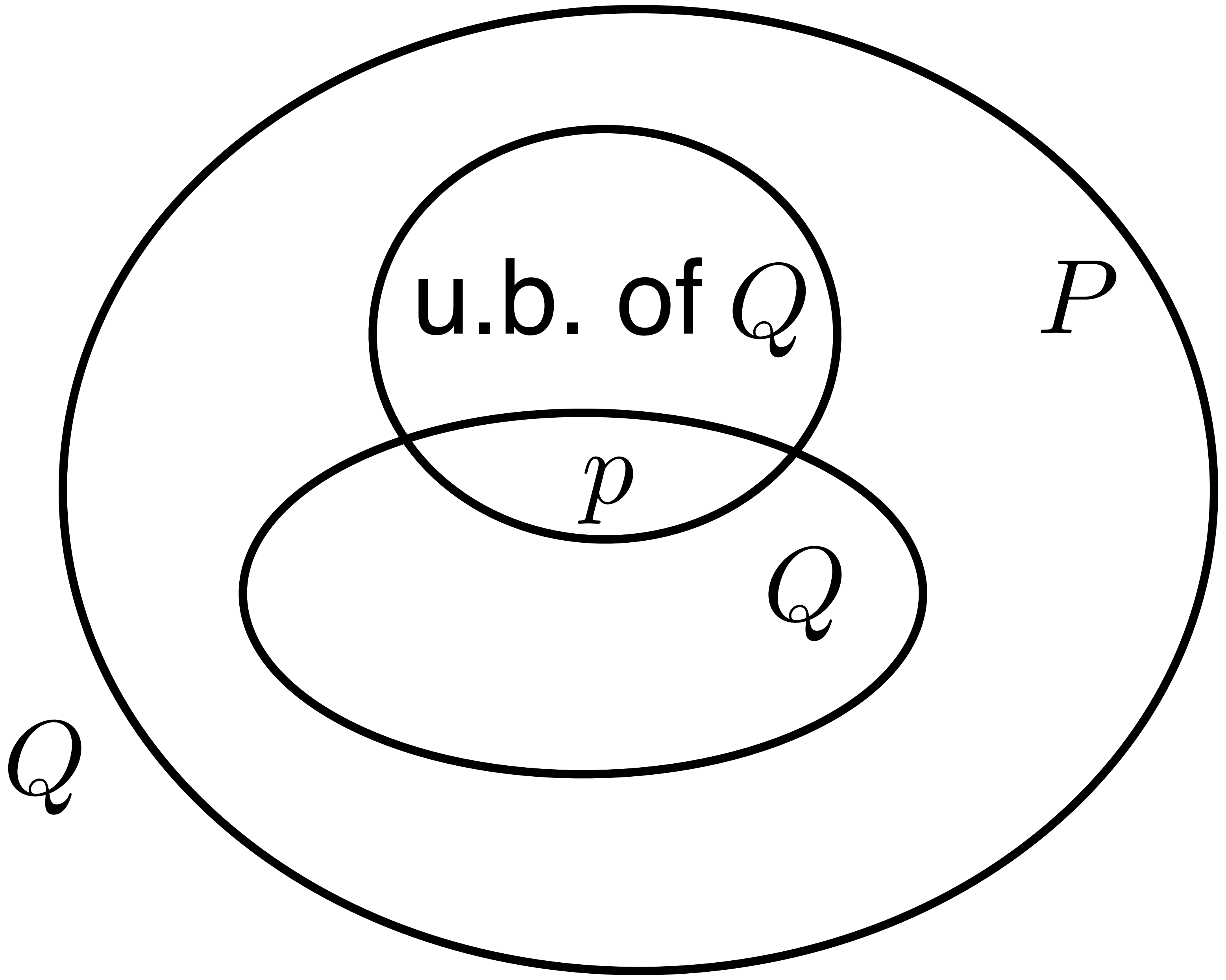
1. it is an upper bound of Q $\quad \forall q \in Q. \quad q \sqsubseteq p$
2. it is smaller than any other upper bound of Q

$$\forall u \in P. \quad (\forall q \in Q. q \sqsubseteq u) \Rightarrow p \sqsubseteq u$$

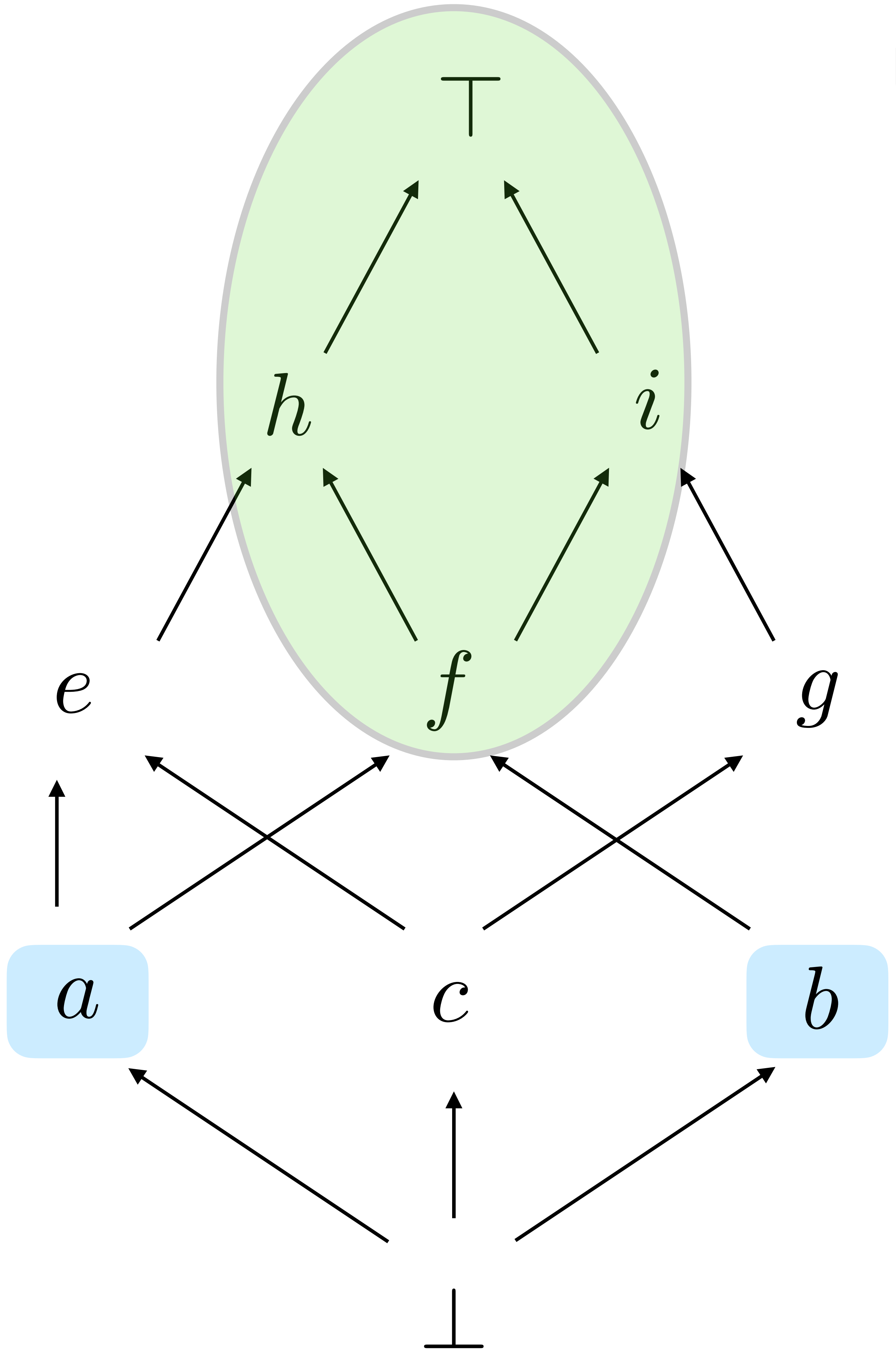
we write $p = lub\ Q$

intuitively, it is the least element that represents all of Q

p not necessarily an element of Q



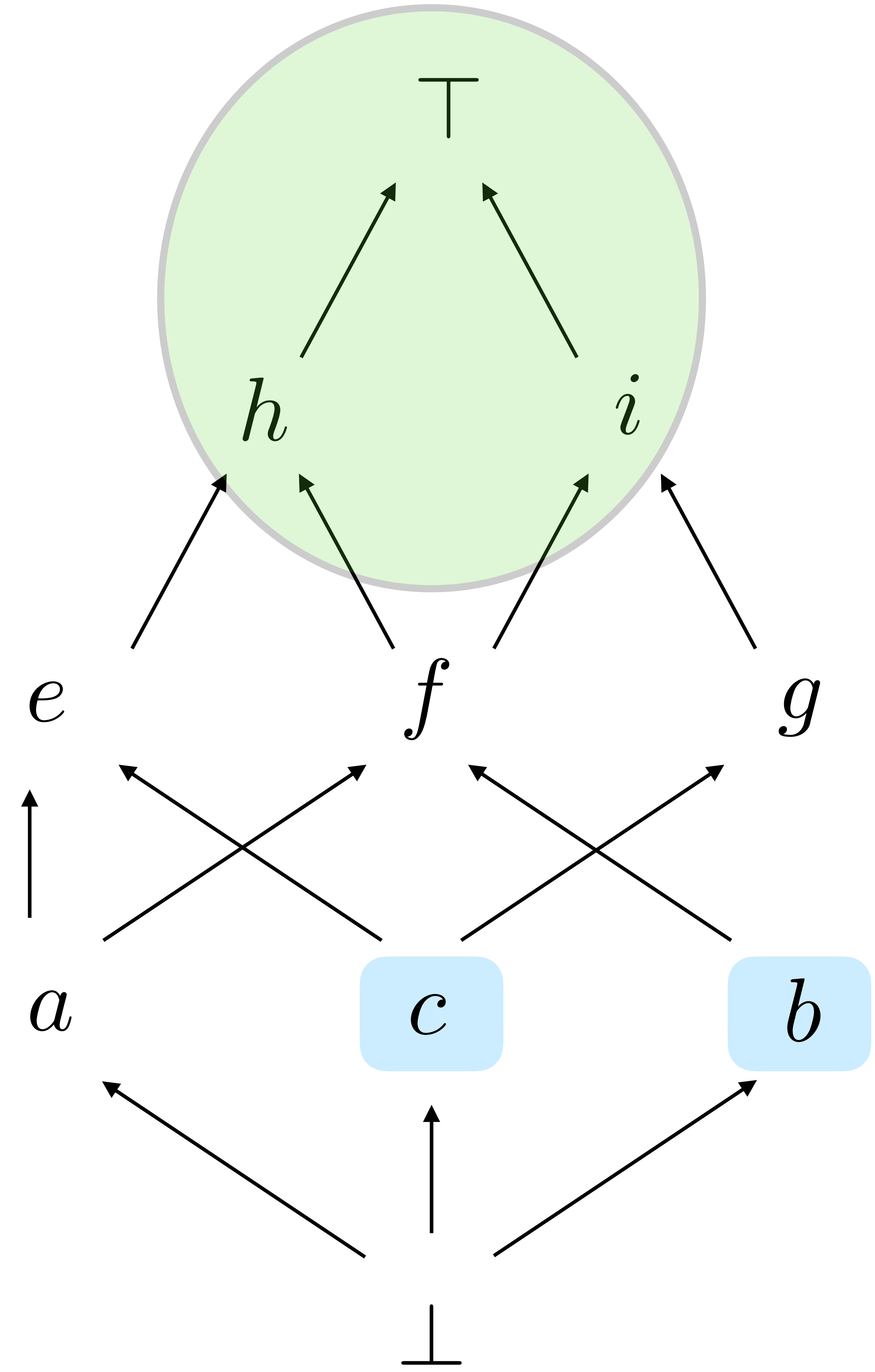
Example



Upper bounds of $\{a, b\}$? $\{f, h, i, \top\}$

lub? f

Example



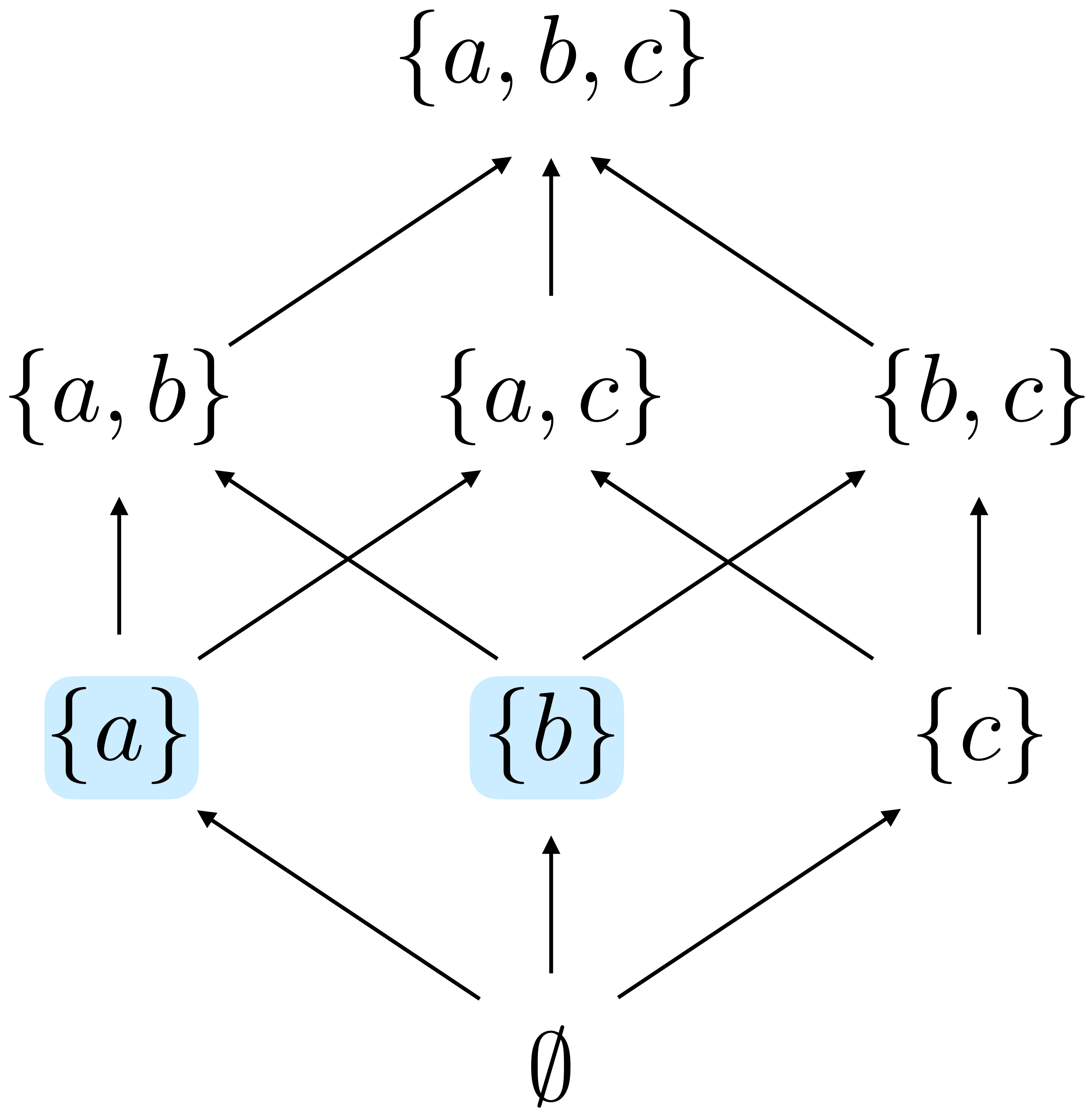
Upper bounds of $\{b, c\}$? $\{h, i, \top\}$

lub? no lub!

Example

$(\wp(S), \subseteq)$ $Q \subseteq \wp(S)$ lub?

$$lub\ Q = \bigcup_{T \in Q} T$$



$$lub\ \{\{a\}, \{b\}\} = \{a, b\}$$

Chain

$(P, \sqsubseteq)$ PO $\{d_i\}_{i\in\mathbb{N}}$ is a chain if $\forall i \in \mathbb{N}.\ d_i \sqsubseteq d_{i+1}$

$$d_0 \sqsubseteq d_1 \sqsubseteq d_2 \sqsubseteq \cdots \sqsubseteq d_n \sqsubseteq \cdots$$

any chain is an infinite list

finite chain: there are only finitely many distinct elements

$$\exists k \in \mathbb{N}.\ \forall i \geq k.\ d_i = d_{i+1}$$

or equivalently

$$\exists k \in \mathbb{N}.\ \forall i \geq k.\ d_i = d_k$$

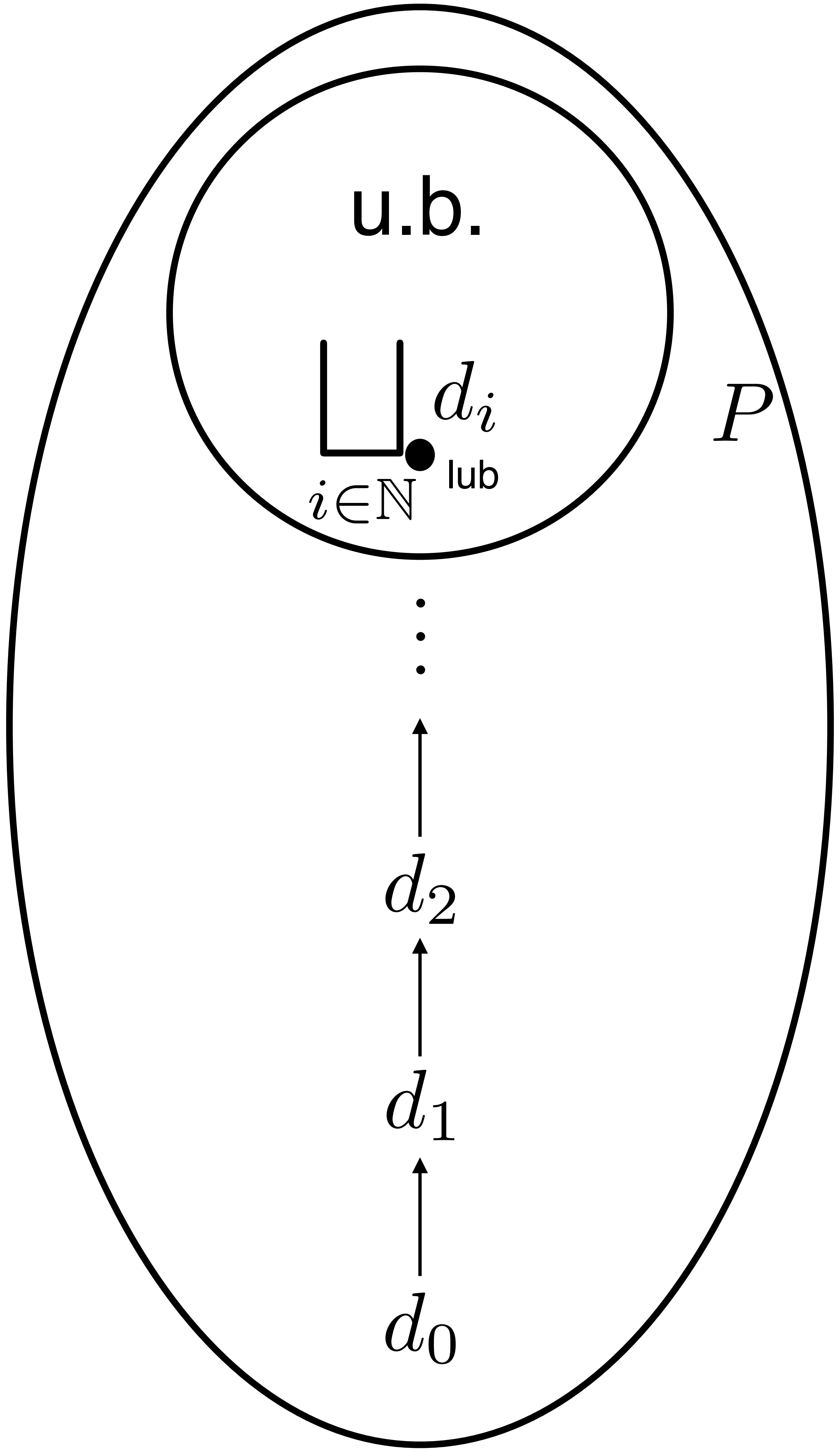
Limit of a chain

$(P, \sqsubseteq)$ PO $\{d_i\}_{i\in\mathbb{N}}$ a chain

we denote by $\bigsqcup_{i\in\mathbb{N}} d_i$ the lub of $\{d_i\}_{i\in\mathbb{N}}$ if it exists

and call it the limit of the chain

Limit illustrated



Complete partial order

$(P, \sqsubseteq)$ PO P is complete if each chain has a limit (lub)

Continuous functions

Monotone function

$(D, \sqsubseteq_D)$ PO $\quad (E, \sqsubseteq_E)$ PO $\quad f : D \to E$

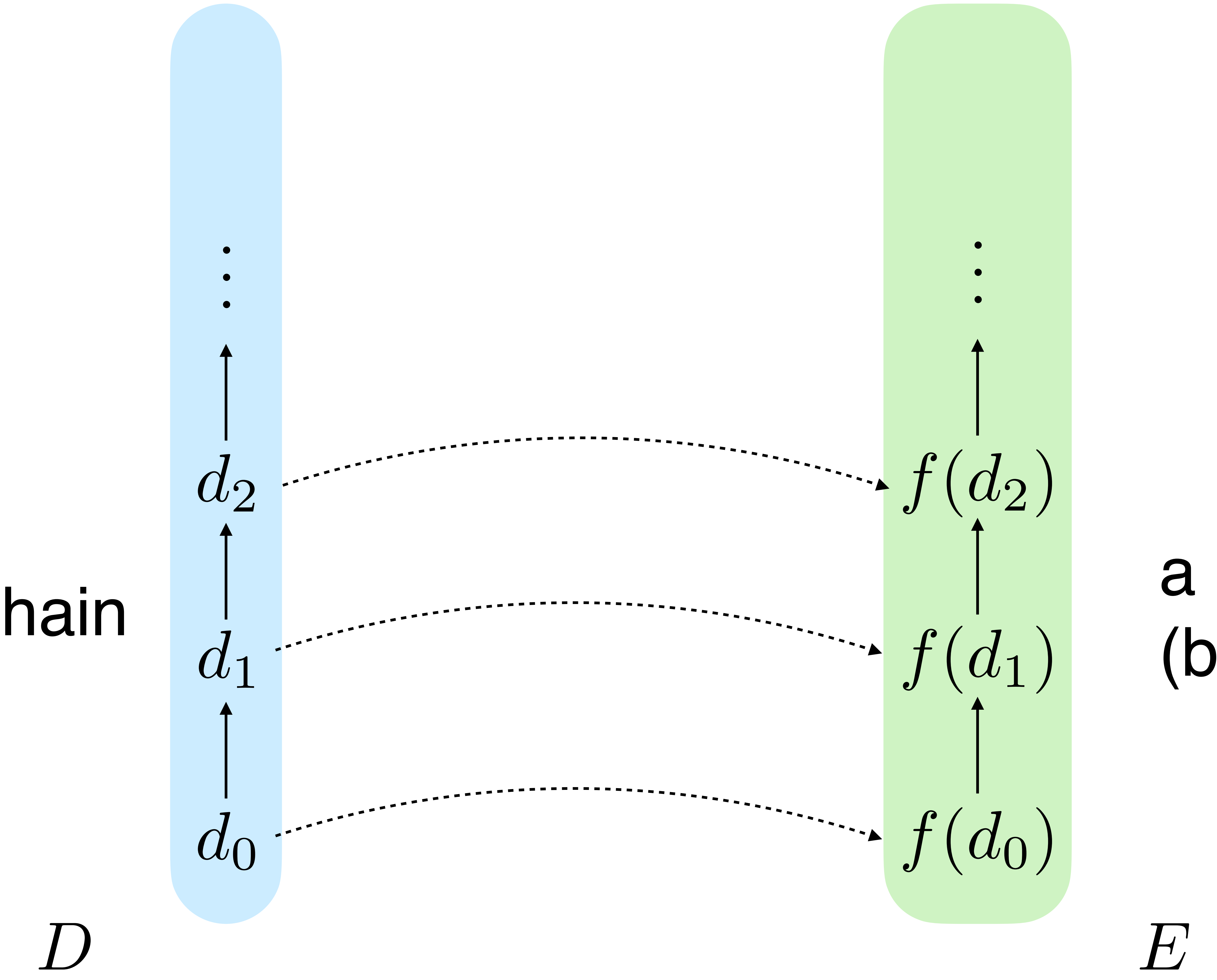
f is monotone if $\forall d_1, d_2 \in D.\ d_1 \sqsubseteq_D d_2 \Rightarrow f(d_1) \sqsubseteq_E f(d_2)$

Monotone = Order preserving

$$\left.\begin{array}{l} \{d_i\}_{i\in\mathbb{N}} \text{ a chain in } D \\ f \text{ monotone} \end{array}\right\} \Rightarrow \{f(d_i)\}_{i\in\mathbb{N}} \text{ a chain in } E$$

When $D = E$ we say $f : D \to D$ is a function on D

Monotonicity illustrated



a chain

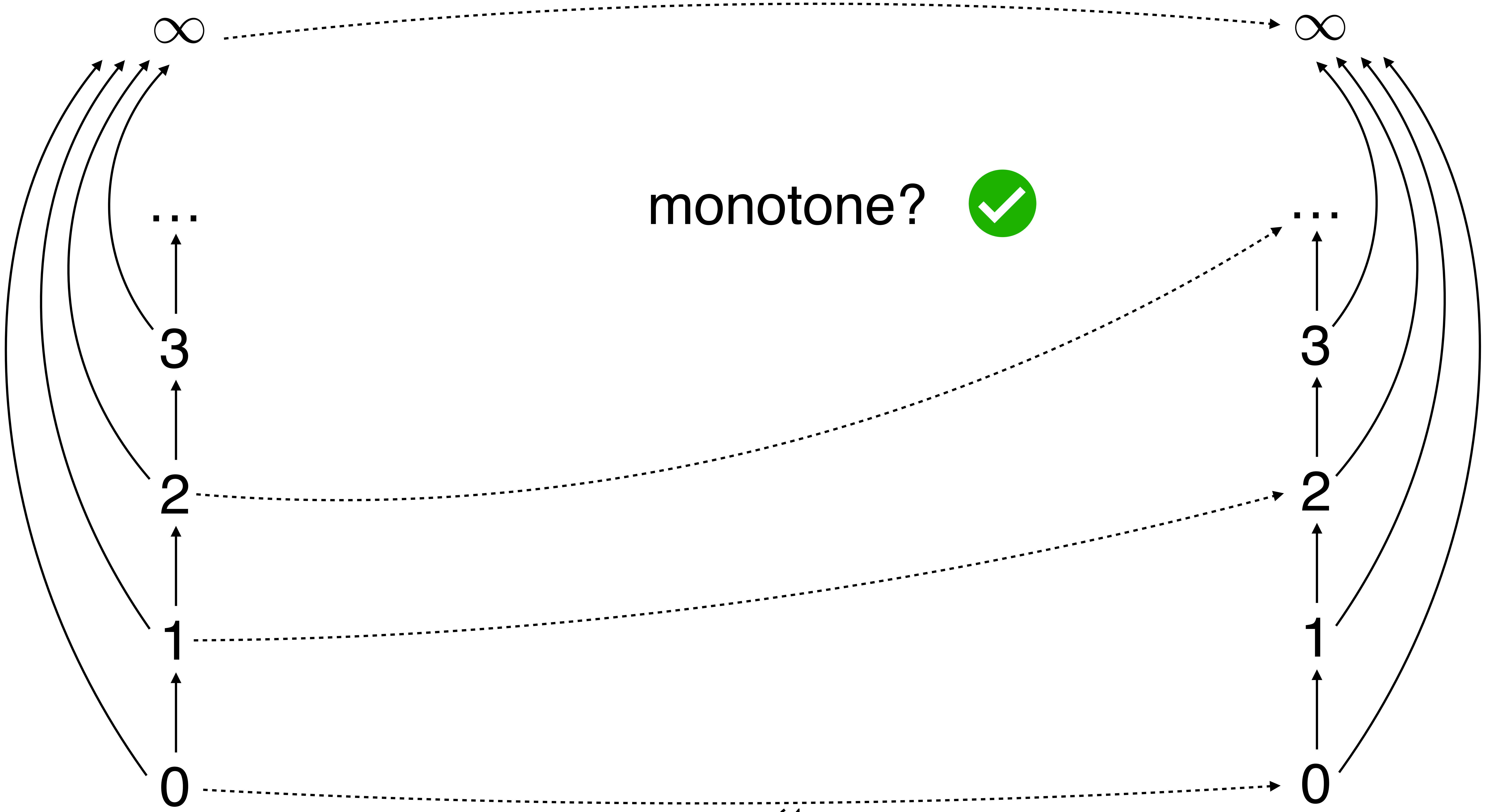
a chain
(by monotonicity)

Example

$(\mathbb{N} \cup \{\infty\}, \leq)$

$$f(n) = 2 \cdot n$$
$$f(\infty) = \infty$$

$(\mathbb{N} \cup \{\infty\}, \leq)$



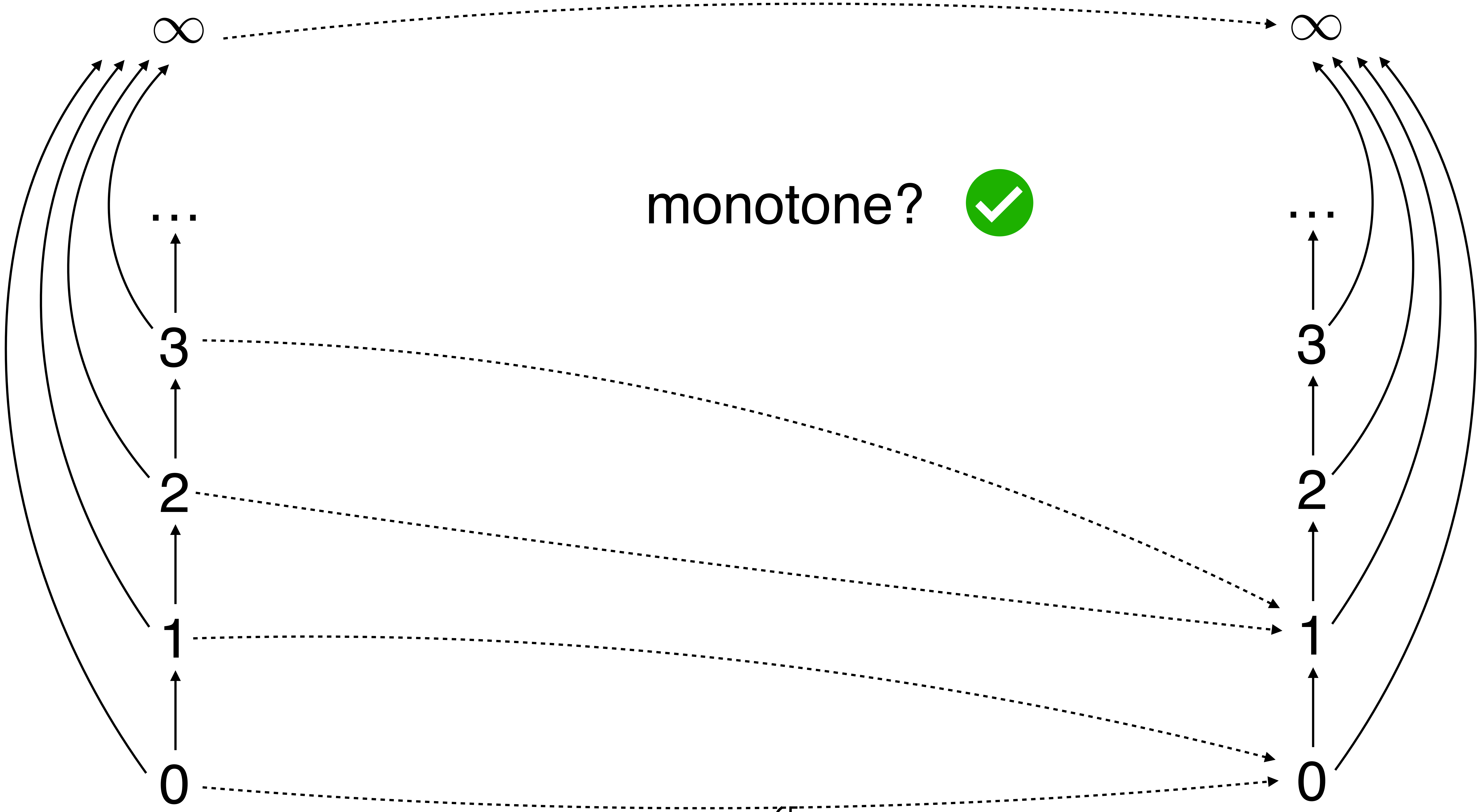
monotone? ✓

Example

$(\mathbb{N} \cup \{\infty\}, \leq)$

$$f(n) = n/2$$
$$f(\infty) = \infty$$

$(\mathbb{N} \cup \{\infty\}, \leq)$



monotone? ✓

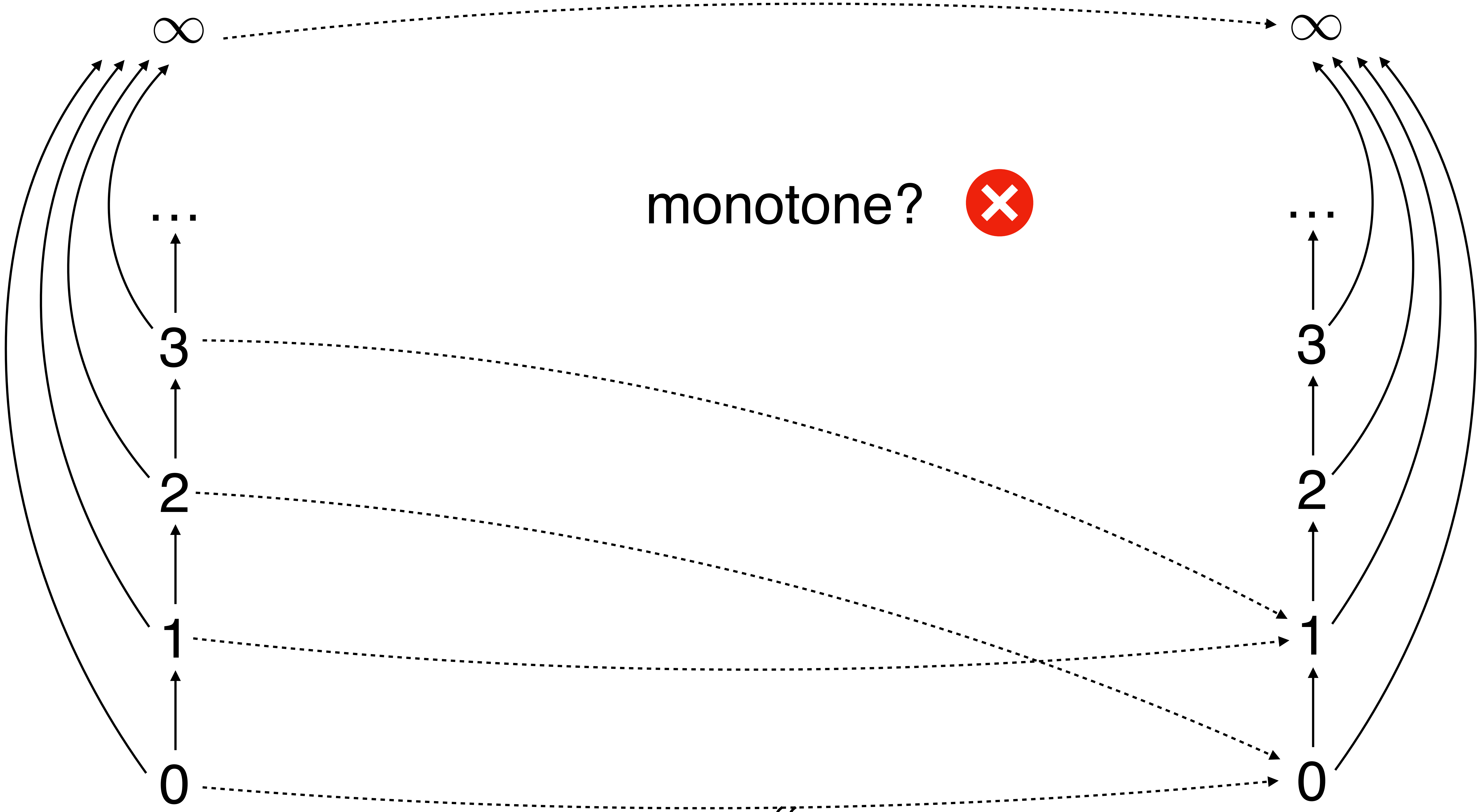
Example

$(\mathbb{N} \cup \{\infty\}, \leq)$

$f(n) = n \ \% \ 2$

$f(\infty) = \infty$

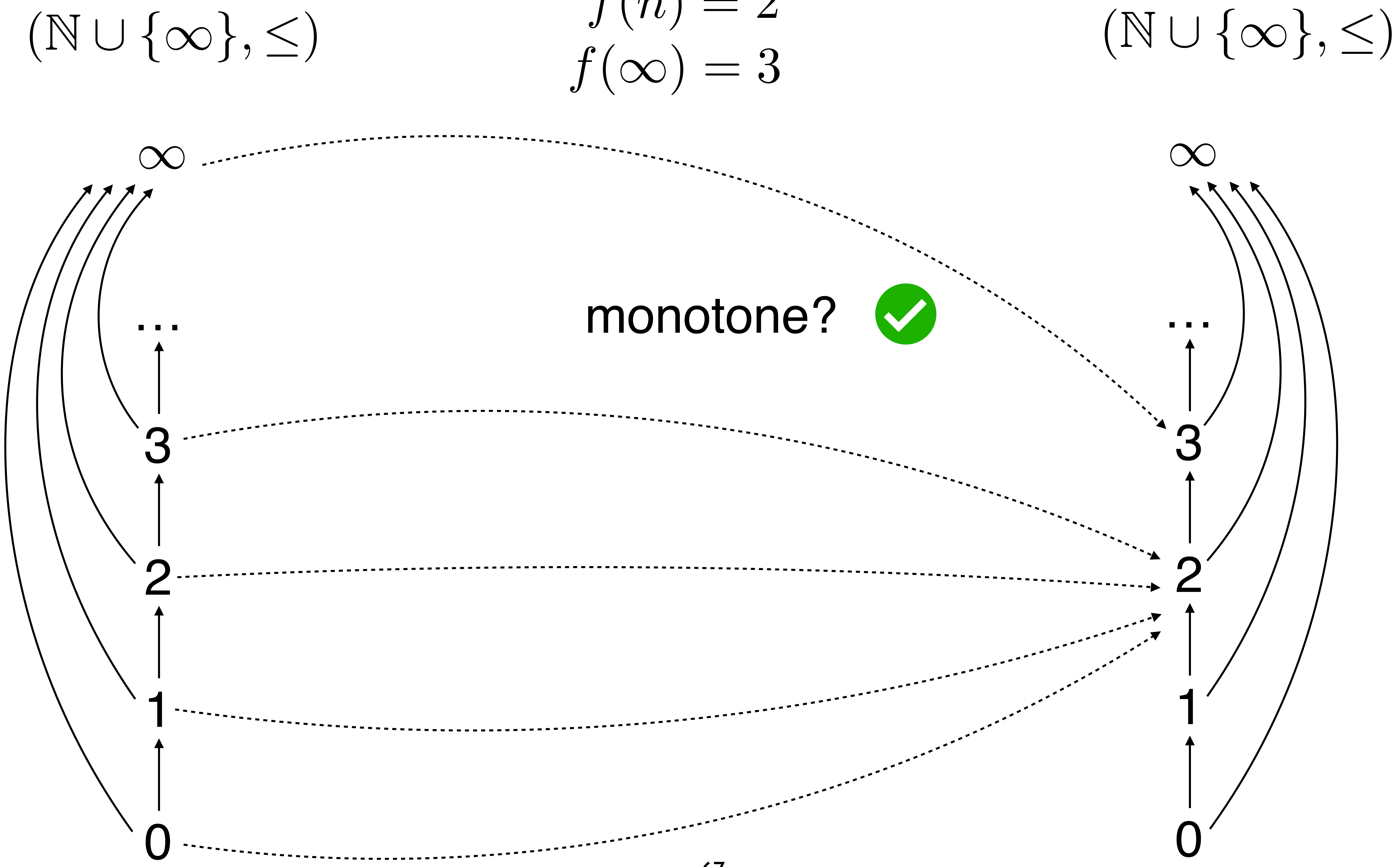
$(\mathbb{N} \cup \{\infty\}, \leq)$



monotone? ✗

Example

$$f(n) = 2$$
$$f(\infty) = 3$$



monotone? ✓

Continuous function

$(D, \sqsubseteq_D)$ CPO $(E, \sqsubseteq_E)$ CPO $f : D \to E$ monotone

f is continuous if $\forall \{d_i\}_{i \in \mathbb{N}}.\ f\left(\bigsqcup_{i \in \mathbb{N}} d_i\right) = \bigsqcup_{i \in \mathbb{N}} f(d_i)$

chain

limit in D

limit in E

Continuous = limit preserving

Continuity illustrated

$$\begin{tikzcd}
\text{lub} & \bigsqcup_{i\in\mathbb{N}} d_i \arrow[rr, dashed, bend left] & & f\left(\bigsqcup_{i\in\mathbb{N}} d_i\right) & \text{u.b.} \\ & & & & \text{(by monotonicity)} \\
& & & \bigsqcup_{i\in\mathbb{N}} f(d_i) \arrow[uu, "=?"'] & \text{lub} \\
& \vdots & & \vdots & \\
& d_2 \arrow[u] \arrow[rr, dashed, bend left] & & f(d_2) \arrow[u] & \\
\text{a chain} & d_1 \arrow[u] \arrow[rr, dashed, bend left] & & f(d_1) \arrow[u] & \text{a chain} \\ & & & & \text{(by monotonicity)} \\
& d_0 \arrow[uu] \arrow[rr, dashed, bend left] & & f(d_0) \arrow[uu] &
\end{tikzcd}$$

Continuity illustrated

lub

$$\begin{tikzcd}
\bigsqcup_{i\in\mathbb{N}} d_i \arrow[rr, dashed, bend left] & & f\left(\bigsqcup_{i\in\mathbb{N}} d_i\right) \\
 & & \bigsqcup_{i\in\mathbb{N}} f(d_i) \arrow[u, "=?"'] \\
\vdots & & \vdots \\
d_2 \arrow[u] \arrow[rr, dashed, bend left] & & f(d_2) \arrow[u] \\
d_1 \arrow[u] \arrow[rr, dashed, bend left] & & f(d_1) \arrow[u] \\
d_0 \arrow[u] \arrow[rr, dashed, bend left] & & f(d_0) \arrow[u]
\end{tikzcd}$$

a chain

$$\bigsqcup_{i\in\mathbb{N}} f(d_i) \sqsubseteq_E f\left(\bigsqcup_{i\in\mathbb{N}} d_i\right)$$

follows from monotonicity (and CPO)

Continuity illustrated

$$\begin{tikzcd}
\text{lub} & \bigsqcup_{i\in\mathbb{N}} d_i \arrow[rr, dashed, bend left=10] & & f\left(\bigsqcup_{i\in\mathbb{N}} d_i\right) \\
 & & & \bigsqcup_{i\in\mathbb{N}} f(d_i) \arrow[u, "=?"'] \\
 & \vdots & & \vdots \\
 & d_2 \arrow[u] \arrow[rr, dashed, bend left=10] & & f(d_2) \arrow[u] \\
\text{a chain} & d_1 \arrow[u] \arrow[rr, dashed, bend left=10] & & f(d_1) \arrow[u] \\
 & d_0 \arrow[u] \arrow[rr, dashed, bend left=10] & & f(d_0) \arrow[u]
\end{tikzcd}$$

$$\bigsqcup_{i\in\mathbb{N}} f(d_i) \overset{?}{\sqsupseteq}_E f\left(\bigsqcup_{i\in\mathbb{N}} d_i\right)$$

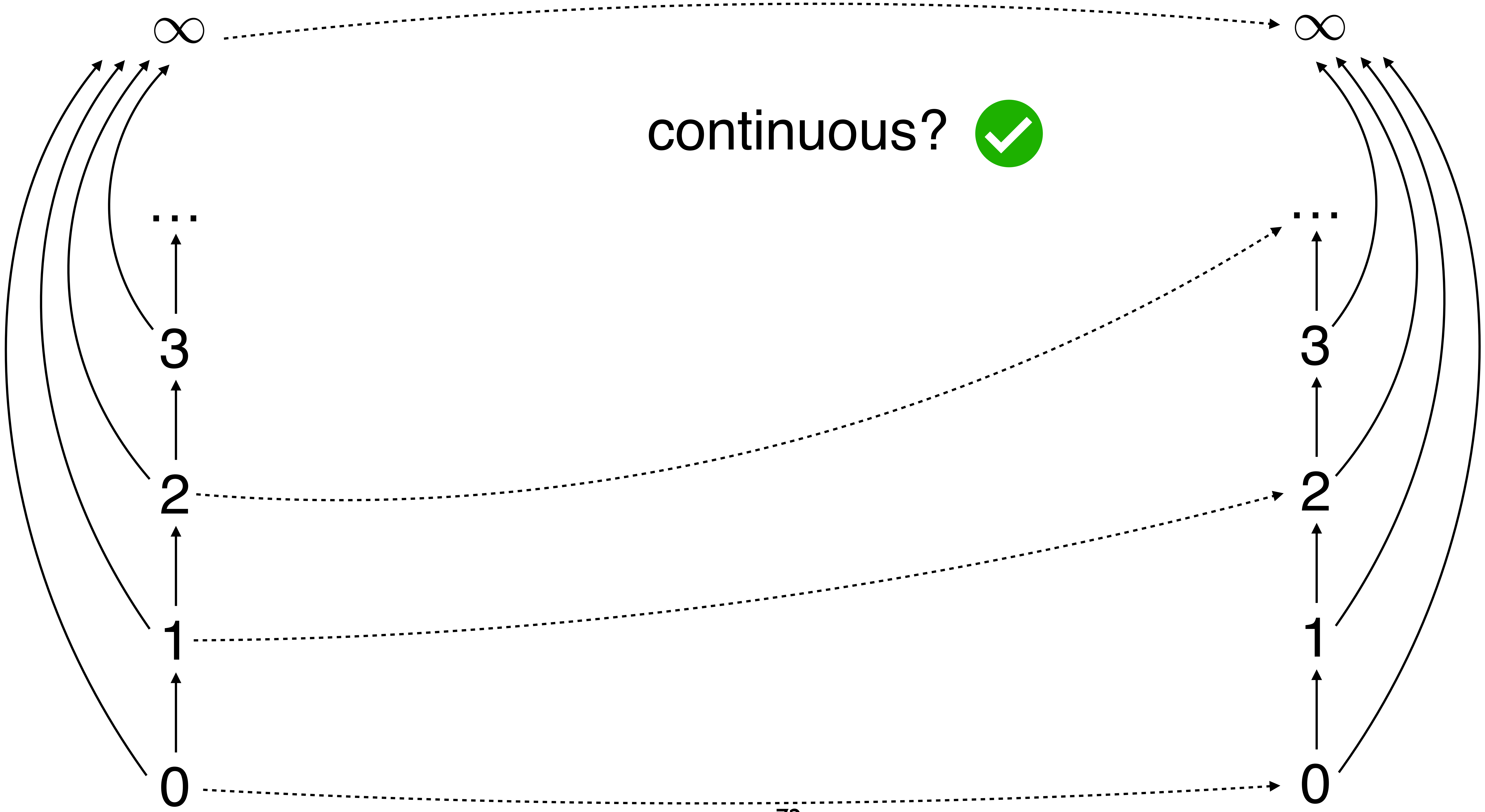
continuity

Example

$(\mathbb{N} \cup \{\infty\}, \leq)$

$$f(n) = 2 \cdot n$$
$$f(\infty) = \infty$$

$(\mathbb{N} \cup \{\infty\}, \leq)$



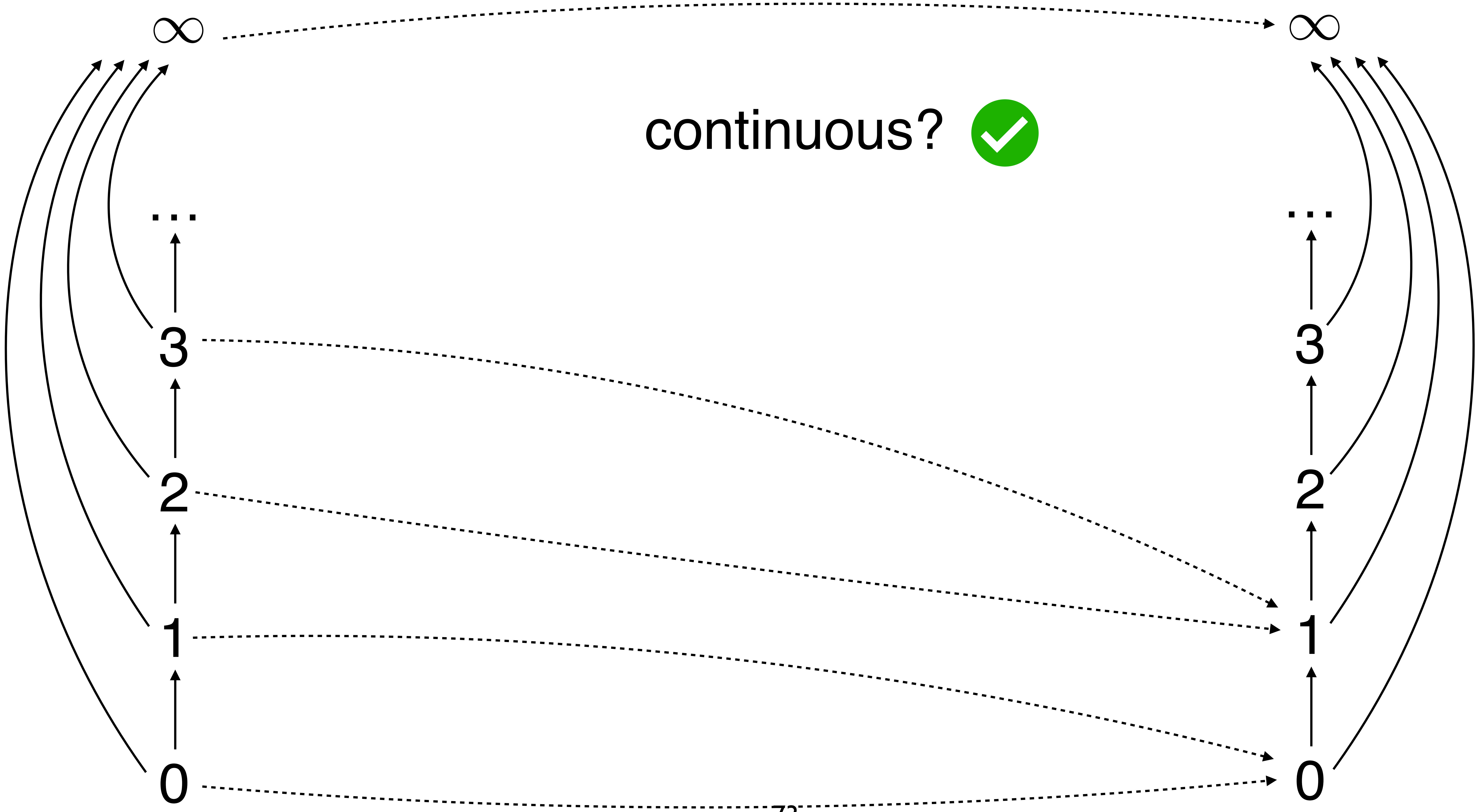
continuous? ✓

Example

$(\mathbb{N} \cup \{\infty\}, \leq)$

$$f(n) = n/2$$
$$f(\infty) = \infty$$

$(\mathbb{N} \cup \{\infty\}, \leq)$

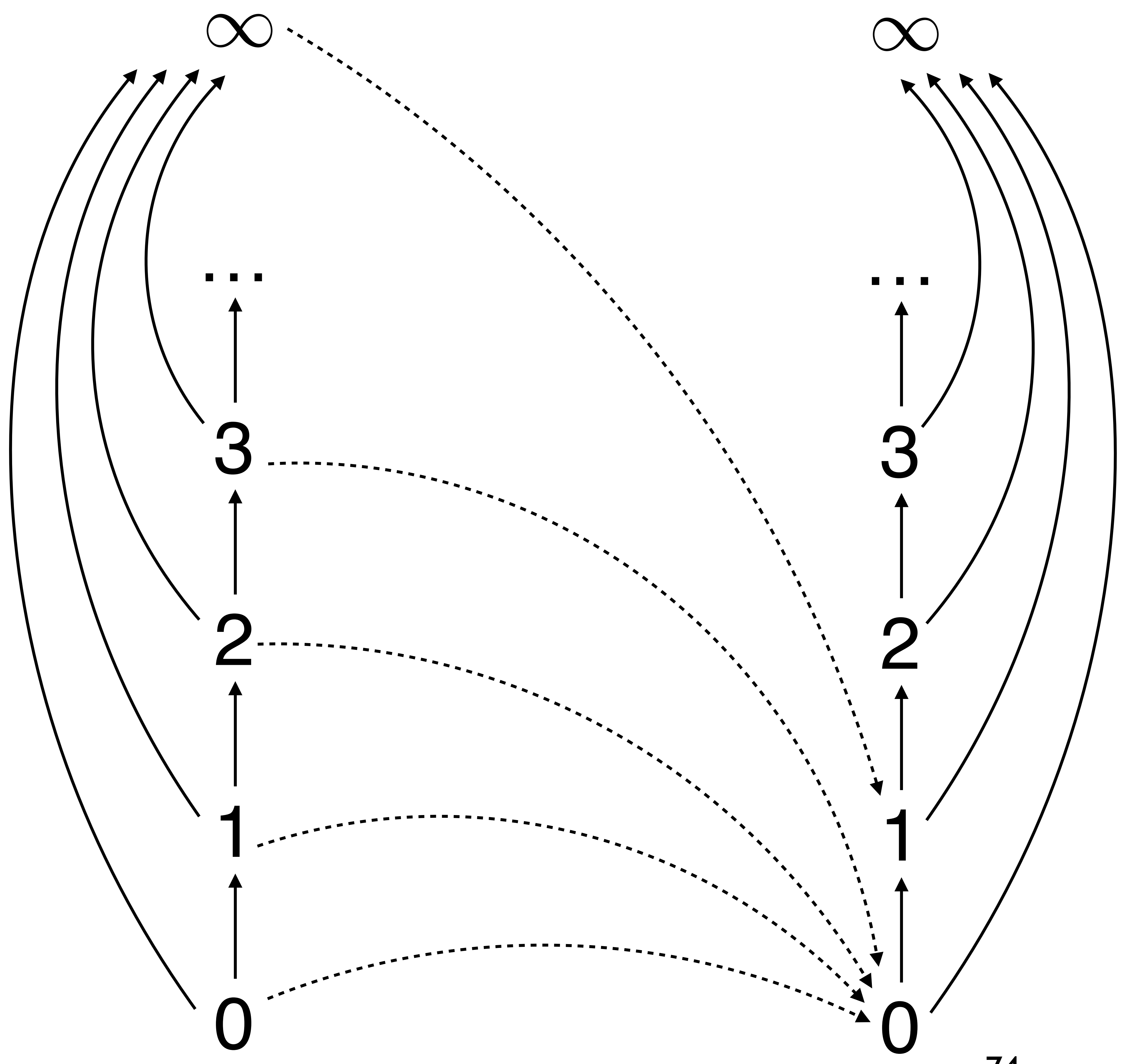


continuous? ✓

Example

$(\mathbb{N} \cup \{\infty\}, \leq)$

monotone function, not continuous



$$f(x) = \begin{cases} 0 & \text{if } x \in \mathbb{N} \\ 1 & \text{if } x = \infty \end{cases}$$

$$d_i = 2 \cdot i \qquad \bigsqcup_{i \in \mathbb{N}} d_i = \infty$$

$$f\left(\bigsqcup_{i \in \mathbb{N}} d_i\right) = f(\infty) = 1$$

$$f(d_i) = 0$$

$$\bigsqcup_{i \in \mathbb{N}} f(d_i) = \bigsqcup_{i \in \mathbb{N}} 0 = 0$$

Kleene’s fixpoint theorem

Repeated application

$$f : D \to D$$

$$f^0(d) \triangleq d$$

$$f^{n+1}(d) \triangleq f(f^n(d))$$

$$f^n(d) = \overbrace{f(\cdots(f}^{n \text{ times}}(d))\cdots)$$

$$f^n : D \to D$$

Towards Kleene's theo.

when $(D, \sqsubseteq)$ is a $\mathrm{CPO}_\bot$

then $\{f^n(\bot)\}_{n\in\mathbb{N}}$ is a chain

it must have a limit

$\{f^n(d)\}_{n\in\mathbb{N}}$
not necessarily
a chain!

Kleene's fix point theorem states that
if f is continuous, then the limit of the above chain
is the least fixpoint of f

Kleene's theorem

$(D, \sqsubseteq)$ $\mathrm{CPO}_{\perp}$ $\quad f : D \to D$ continuous

let $\mathit{fix}(f) \triangleq \bigsqcup_{n \in \mathbb{N}} f^n(\perp)$

1. $\mathit{fix}(f)$ is a fix point of f

$$f(\mathit{fix}(f)) = \mathit{fix}(f)$$

2. $\mathit{fix}(f)$ is the least fixpoint of f

$$\forall d \in D.\ f(d) = d \Rightarrow \mathit{fix}(f) \sqsubseteq d$$

if d is a fixpoint then $\mathit{fix}(f)$ is smaller than d

Example

$n = 2 \cdot n$ $(\mathbb{N} \cup \{\infty\}, \leq)$ $\bot = 0$ $\mathrm{CPO}_\bot$

$$f(n) = 2 \cdot n$$
$$f(\infty) = \infty$$

monotone? ok

continuous? ok

$$f^0(0) = 0$$

$$f^1(0) = f(0) = 2 \cdot 0 = 0$$

fixpoint reached!

Example

$n = n + 1$ $(\mathbb{N} \cup \{\infty\}, \leq)$ $\bot = 0$ $\text{CPO}_\bot$

$$\begin{aligned} f(n) &= n + 1 \\ f(\infty) &= \infty \end{aligned}$$

monotone? ok

continuous? ok

$$\begin{aligned} f^0(0) &= 0 \\ f^1(0) &= f(0) = 0 + 1 = 1 \\ f^2(0) &= f(f^1(0)) = f(1) = 1 + 1 = 2 \\ f^3(0) &= f(f^2(0)) = f(2) = 2 + 1 = 3 \end{aligned}$$

$$\bigsqcup_{n \in \mathbb{N}} f^n(0) = \bigsqcup_{n \in \mathbb{N}} n = \infty$$ fixpoint

Example

$X = X \cap \{1\}$ $(\wp(\mathbb{N}), \subseteq)$ $\bot = \emptyset$ $\mathrm{CPO}_\bot$

$$f(X) = X \cap \{1\}$$

monotone? ok

continuous? ok

$$f^0(\emptyset) = \emptyset$$

$$f^1(\emptyset) = f(\emptyset) = \emptyset \cap \{1\} = \emptyset$$

fixpoint reached!

Example

$X = \mathbb{N} \setminus X$ $(\wp(\mathbb{N}), \subseteq)$ $\bot = \emptyset$ $\mathrm{CPO}_\bot$

$f(X) = \mathbb{N} \setminus X$

monotone? NO

the larger X the smaller $f(X)$

$$f^0(\emptyset) = \emptyset$$

$$f^1(\emptyset) = f(\emptyset) = \mathbb{N} \setminus \emptyset = \mathbb{N}$$

$$f^2(\emptyset) = f(f^1(\emptyset)) = f(\mathbb{N}) = \mathbb{N} \setminus \mathbb{N} = \emptyset$$

not a chain!

Example

$X = X \cup \{1\}$ $(\wp(\mathbb{N}), \subseteq)$ $\bot = \emptyset$ $\mathrm{CPO}_\bot$

$$f(X) = X \cup \{1\}$$

monotone? ok
continuous? ok

$$f^0(\emptyset) = \emptyset$$

$$f^1(\emptyset) = f(\emptyset) = \emptyset \cup \{1\} = \{1\}$$

$$f^2(\emptyset) = f(f^1(\emptyset)) = f(\{1\}) = \{1\} \cup \{1\} = \{1\}$$

fixpoint reached!

Back to commands semantics

Loops as fix points

$$[\![\,\cdot\,]\!] : \mathsf{Com} \to \wp(\Sigma) \to \wp(\Sigma)$$

$$[\![\mathsf{while}\ b\ \mathsf{do}\ c]\!]P \triangleq [\![\mathtt{not}\ b]\!]P \cup [\![\mathsf{while}\ b\ \mathsf{do}\ c]\!]([\![c]\!]([\![b]\!]P))$$

how do we know one solution exists? how do we know it is unique?

$$[\![\mathsf{while}\ b\ \mathsf{do}\ c]\!] \triangleq \lambda P\,.\,[\![\mathtt{not}\ b]\!]P \cup [\![\mathsf{while}\ b\ \mathsf{do}\ c]\!]([\![c]\!]([\![b]\!]P))$$

$$[\![\mathsf{while}\ b\ \mathsf{do}\ c]\!] \triangleq (\lambda\varphi\,.\,\lambda P\,.\,[\![\mathtt{not}\ b]\!]P \cup \varphi([\![c]\!]([\![b]\!]P)))\ [\![\mathsf{while}\ b\ \mathsf{do}\ c]\!]$$

$$\Gamma \triangleq \lambda\varphi\,.\,\lambda P\,.\,[\![\mathtt{not}\ b]\!]P \cup \varphi([\![c]\!]([\![b]\!]P))$$

looking for fixpoint

$$[\![\mathsf{while}\ b\ \mathsf{do}\ c]\!] \triangleq \Gamma([\![\mathsf{while}\ b\ \mathsf{do}\ c]\!]) = \mathsf{fix}\ \Gamma$$

Loops as fix points

$[\![\textsf{while}\, b \,\textsf{do}\, c]\!] \triangleq \textsf{fix}\ \Gamma$ $\qquad$ $\Gamma \triangleq \lambda\varphi\,.\,\lambda P\,.\,[\![\texttt{not}\ b]\!]P \cup \varphi(([\![c]\!] \circ [\![b]\!])P)$

domain is $\wp(\Sigma) \to \wp(\Sigma)$ (a CPO with bottom!)

bottom element: $\bot = \lambda P\,.\,\emptyset$ $\quad$ Γ is continuous (proof omitted)

fixpoint: $\bigsqcup_{n} \Gamma^n(\ \bot\ )$

$$\varphi_0 = \Gamma^0(\ \bot\ ) = \ \bot = \lambda P\,.\,\emptyset$$
$$\varphi_1 = \Gamma^1(\ \bot\ ) = \Gamma(\varphi_0) = \lambda P\,.\,[\![\texttt{not}\ b]\!]P$$
$$\varphi_2 = \Gamma^2(\ \bot\ ) = \Gamma(\varphi_1) = \lambda P\,.\,[\![\texttt{not}\ b]\!]P \cup [\![\texttt{not}\ b]\!](([\![c]\!] \circ [\![b]\!])P)$$
$$\varphi_3 = \Gamma^3(\ \bot\ ) = \Gamma(\varphi_2) = \lambda P\,.\,[\![\texttt{not}\ b]\!]P \cup [\![\texttt{not}\ b]\!](([\![c]\!] \circ [\![b]\!])P) \cup [\![\texttt{not}\ b]\!](([\![c]\!] \circ [\![b]\!])^2P)$$
$$\ldots$$
$$\varphi_{n+1} = \Gamma^{n+1}(\ \bot\ ) = \Gamma(\varphi_n) = \lambda P\,.\,[\![\texttt{not}\ b]\!]P \cup \cdots \cup [\![\texttt{not}\ b]\!](([\![c]\!] \circ [\![b]\!])^nP)$$

Loops as fix points

$[\![\text{while } b \text{ do } c]\!] \triangleq \text{fix } \Gamma$ $\qquad$ $\Gamma \triangleq \lambda\varphi \,.\, \lambda P \,.\, [\![\texttt{not } b]\!]P \cup \varphi(([\![c]\!] \circ [\![b]\!])P)$

fixpoint: $\bigsqcup_{n} \Gamma^n(\,\bot\,)$ $\qquad$ where: $\Gamma^{n+1}(\,\bot\,) = \lambda P \,.\, [\![\texttt{not } b]\!]P$

$$\cup\; [\![\texttt{not } b]\!](([\![c]\!] \circ [\![b]\!])P)$$

$$\ldots$$

$$\cup\; [\![\texttt{not } b]\!](([\![c]\!] \circ [\![b]\!])^n P)$$

$$(\Gamma^{n+1}(\,\bot\,))\,P = [\![\texttt{not } b]\!]P$$

$$\cup\; [\![\texttt{not } b]\!](([\![c]\!] \circ [\![b]\!])P)$$

$$\ldots$$

$$\cup\; [\![\texttt{not } b]\!](([\![c]\!] \circ [\![b]\!])^n P)$$

Loops as fix points

$$[\![\text{while}\ b\ \text{do}\ c]\!] \triangleq \text{fix}\ \Gamma \qquad \Gamma \triangleq \lambda\varphi\,.\,\lambda P\,.\,[\![\mathtt{not}\ b]\!]P \cup \varphi(([\![c]\!] \circ [\![b]\!])P)$$

$$[\![\text{while}\ b\ \text{do}\ c]\!]P = \left(\bigsqcup_n \Gamma^n(\perp)\right)P = \left(\bigsqcup_n (\Gamma^n(\perp))\,P\right)$$

$$= \bigcup_n [\![\mathtt{not}\ b]\!](([\![c]\!] \circ [\![b]\!])^n P)$$

$$= [\![\mathtt{not}\ b]\!] \bigcup_n (([\![c]\!] \circ [\![b]\!])^n P)$$

we can stop when $\bigcup_{n=0}^{k+1} (([\![c]\!] \circ [\![b]\!])^n P) \subseteq \bigcup_{n=0}^{k} (([\![c]\!] \circ [\![b]\!])^n P)$

Example

$$[\![\text{while}\, x > 0\, \text{do}\, x := x - 1]\!]\ (x > 1)$$

$$= [\![x \leq 0]\!] \bigcup_{n} ([\![x := x - 1]\!] \circ [\![x > 0]\!])^n\ (x > 1)$$

$$\bigcup_{n=0}^{0} ([\![x := x - 1]\!] \circ [\![x > 0]\!])^n\ (x > 1) = (x > 1)$$

$$\bigcup_{n=0}^{1} ([\![x := x - 1]\!] \circ [\![x > 0]\!])^n\ (x > 1) = (x > 0)$$

$$\bigcup_{n=0}^{2} ([\![x := x - 1]\!] \circ [\![x > 0]\!])^n\ (x > 1) = (x \geq 0)$$

$$\bigcup_{n=0}^{3} ([\![x := x - 1]\!] \circ [\![x > 0]\!])^n\ (x > 1) = (x \geq 0)$$

$$[\![\text{while}\, x > 0\, \text{do}\, x := x - 1]\!]\ (x > 1) = [\![x \leq 0]\!](x \geq 0) = (x = 0)$$

Questions

Question 1

Let $c \triangleq (z := x) + (z := y)$

and let $P \triangleq (x = y = 0)$

What is $[\![c]\!]P$?

$$(x = y = z = 0)$$

Question 2

Let $c \triangleq$ if $x < y$ then $x := y$ else (while true do skip)

and let $Q \triangleq (x = y = 0)$

What is $wlp(c, Q)$?

$$(x \geq y \vee y = 0)$$