

CSE P 504

Advanced topics in Software Systems

Fall 2022

Abstract Interpretation

November 28, 2022

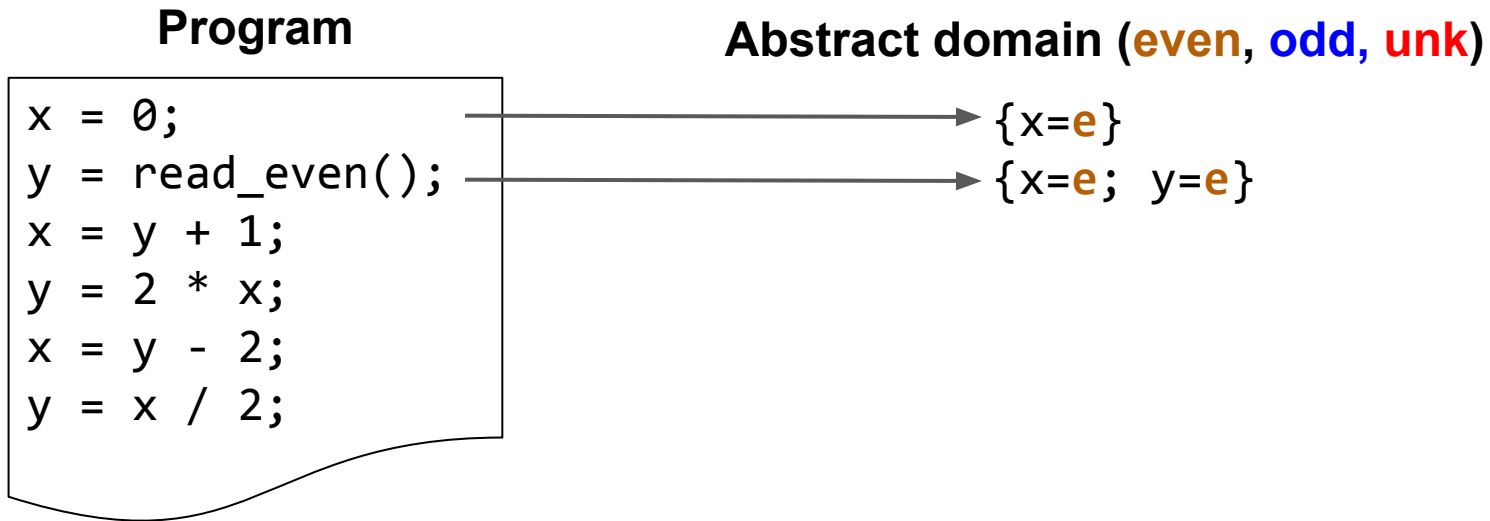
Today

Abstract interpretation

- Lattices
- Abstraction function
- Concretization function
- Transfer function (vs. lub vs. glb)
- Galois connection
- Exercise: concrete examples

Abstract interpretation (intuition)

Abstract domain and abstraction function (intuition)



Transfer function (intuition)

Program

```
x = 0;  
y = read_even();  
x = y + 1;  
y = 2 * x;  
x = y - 2;  
y = x / 2;
```

+	unk	odd	even
unk	unk	unk	unk
odd	unk	even	odd
even	unk	odd	even

Transfer function corresponds to the “abstract execution” of +

Abstract interpretation (a bit more formal)

Set, semilattice, lattice

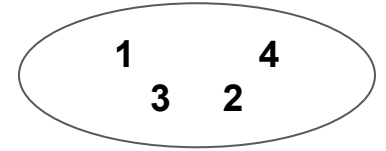
Set, semilattice, lattice

Set

Set, semilattice, lattice

Set

- **unordered** collection of **distinct** objects

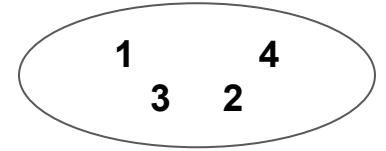


Set, semilattice, lattice

Set

- unordered collection of distinct objects

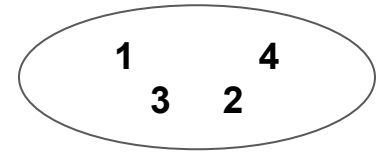
Partially ordered set



Set, semilattice, lattice

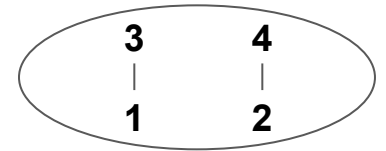
Set

- unordered collection of distinct objects



Partially ordered set

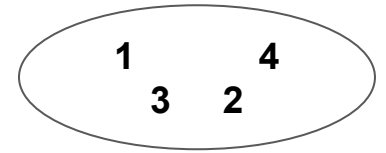
- Binary relationship \leq :
 - Reflexive: $x \leq x$
 - Anti-symmetric: $x \leq y \wedge y \leq x \Rightarrow x = y$
 - Transitive: $x \leq y \wedge y \leq z \Rightarrow x \leq z$



Set, semilattice, lattice

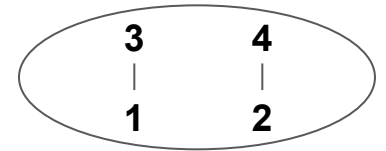
Set

- unordered collection of distinct objects



Partially ordered set

- Binary relationship \leq :
 - Reflexive: $x \leq x$
 - Anti-symmetric: $x \leq y \wedge y \leq x \Rightarrow x = y$
 - Transitive: $x \leq y \wedge y \leq z \Rightarrow x \leq z$



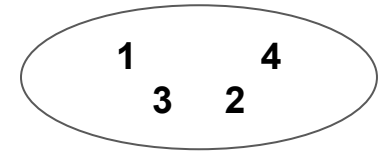
Join semilattice

Meet semilattice

Set, semilattice, lattice

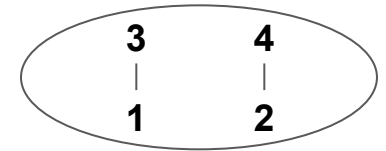
Set

- unordered collection of distinct objects



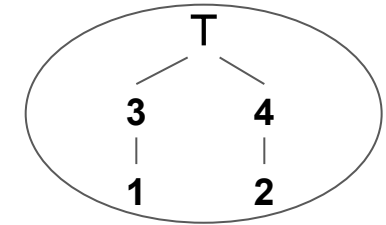
Partially ordered set

- Binary relationship \leq :
 - Reflexive: $x \leq x$
 - Anti-symmetric: $x \leq y \wedge y \leq x \Rightarrow x = y$
 - Transitive: $x \leq y \wedge y \leq z \Rightarrow x \leq z$



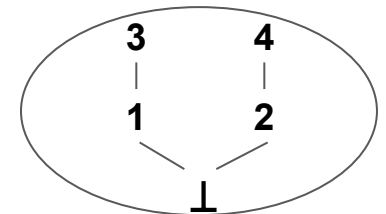
Join semilattice

- Partially ordered set with least upper bound (join)



Meet semilattice

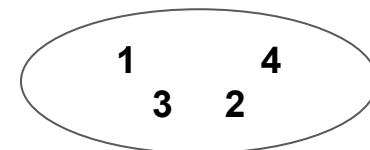
- Partially ordered set with greatest lower bound (meet)



Set, semilattice, lattice

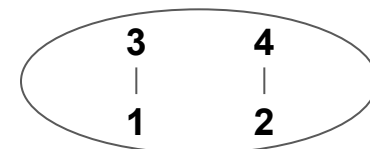
Set

- unordered collection of distinct objects



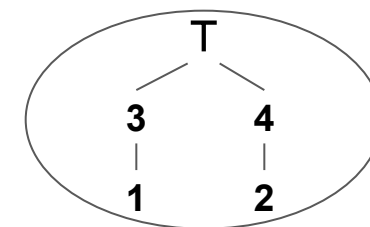
Partially ordered set

- Binary relationship \leq :
 - Reflexive: $x \leq x$
 - Anti-symmetric: $x \leq y \wedge y \leq x \Rightarrow x = y$
 - Transitive: $x \leq y \wedge y \leq z \Rightarrow x \leq z$



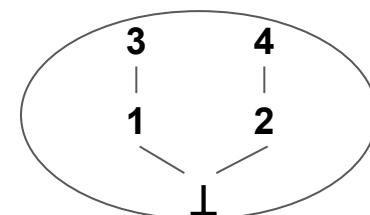
Join semilattice

- Partially ordered set with least upper bound (join)



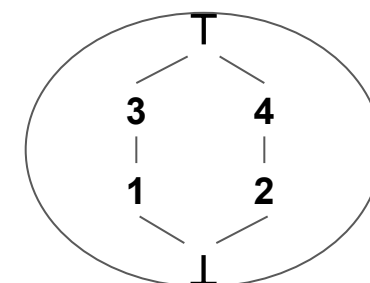
Meet semilattice

- Partially ordered set with greatest lower bound (meet)



Lattice

- Both a join semilattice and a meet semilattice

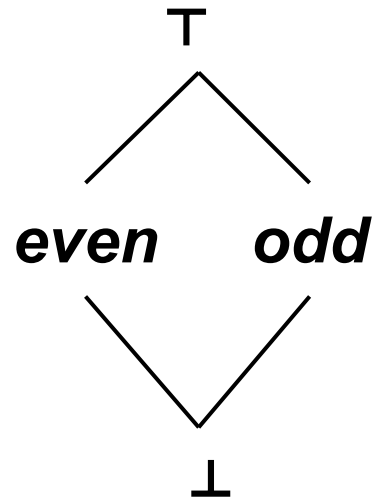


Lattice: example

Abstract domain: *even, odd, unknown, {}*

Lattice: example

Abstract domain: *even, odd, unknown* (\top), $\{\}$ (\perp)



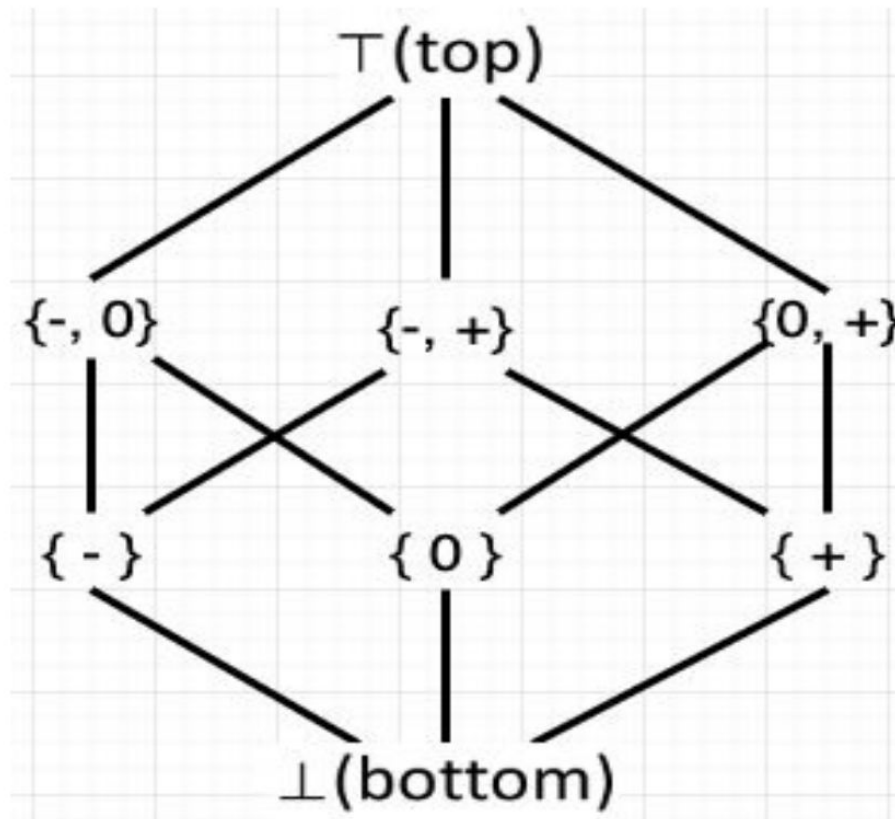
Lattice: example



Abstract domain: -, 0, +, *unknown*, }

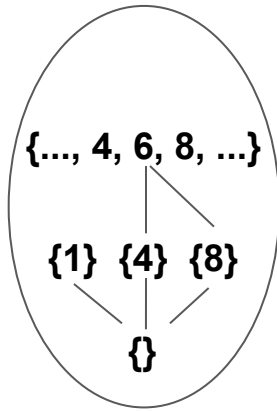
Lattice: example

Abstract domain: $-$, 0 , $+$, *unknown*, $\{\}$

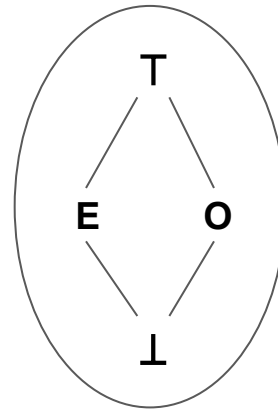


Abstraction function

Concrete ($P(\mathbb{N})$)



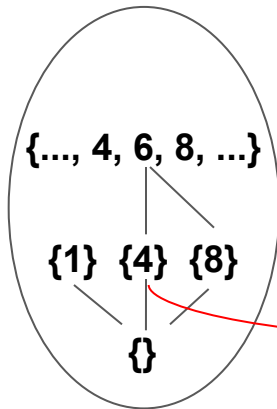
Abstract



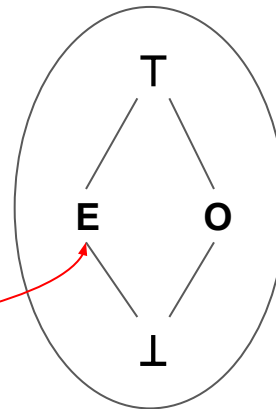
What is the abstraction (α) of $\{4\}$?

Abstraction function

Concrete ($P(\mathbb{N})$)



Abstract

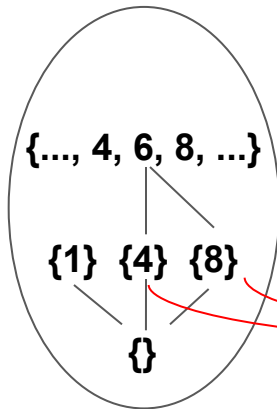


α

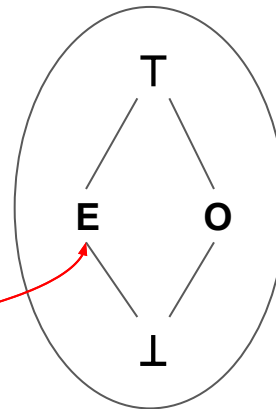
What is the abstraction (α) of $\{8\}$?

Abstraction function

Concrete ($P(\mathbb{N})$)



Abstract

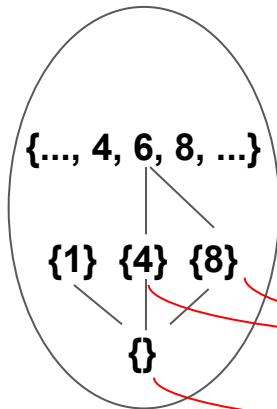


α

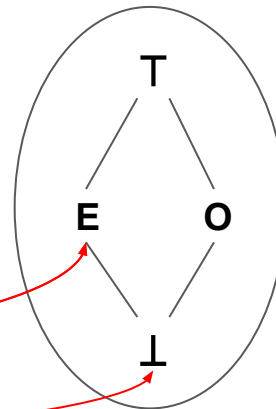
What is the abstraction (α) of $\{\}$?

Abstraction function

Concrete ($P(\mathbb{N})$)



Abstract



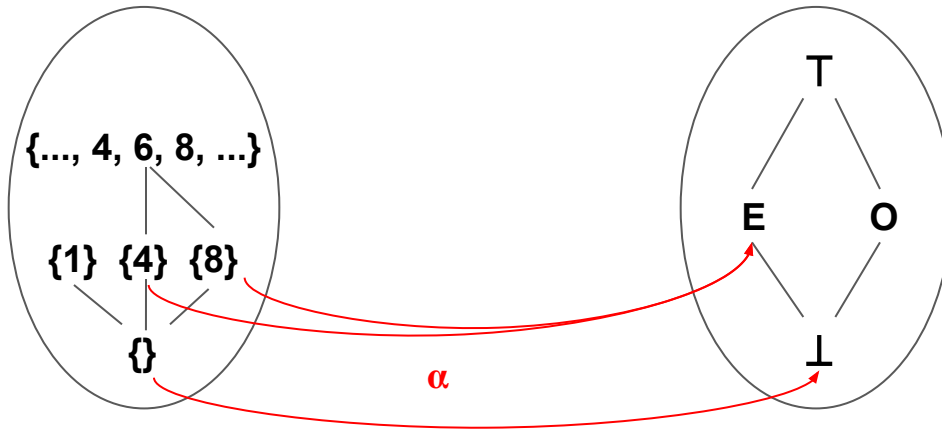
α

Why do we need an abstraction function?

Concretization function

Concrete ($P(\mathbb{N})$)

Abstract

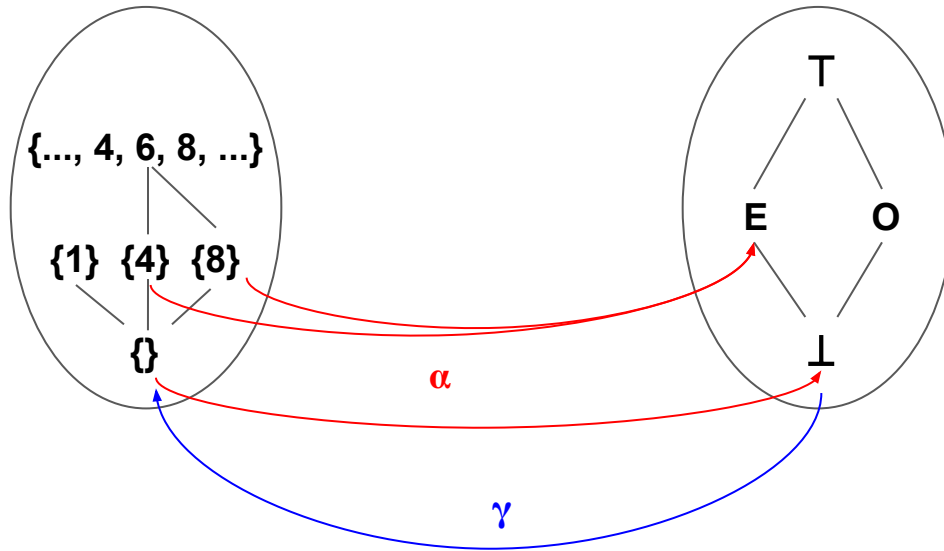


What is the concretization (γ) of \perp ?

Concretization function

Concrete ($P(\mathbb{N})$)

Abstract

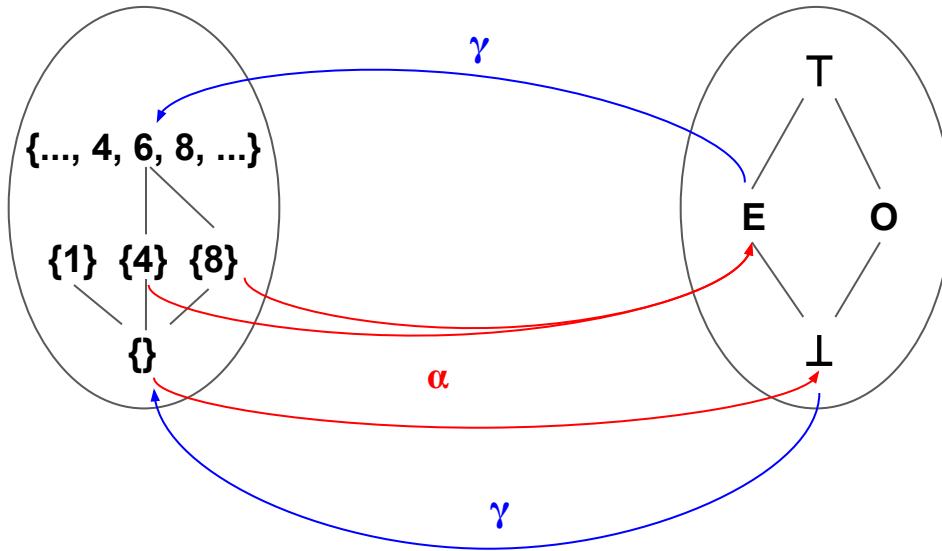


What is the concretization (γ) of E ?

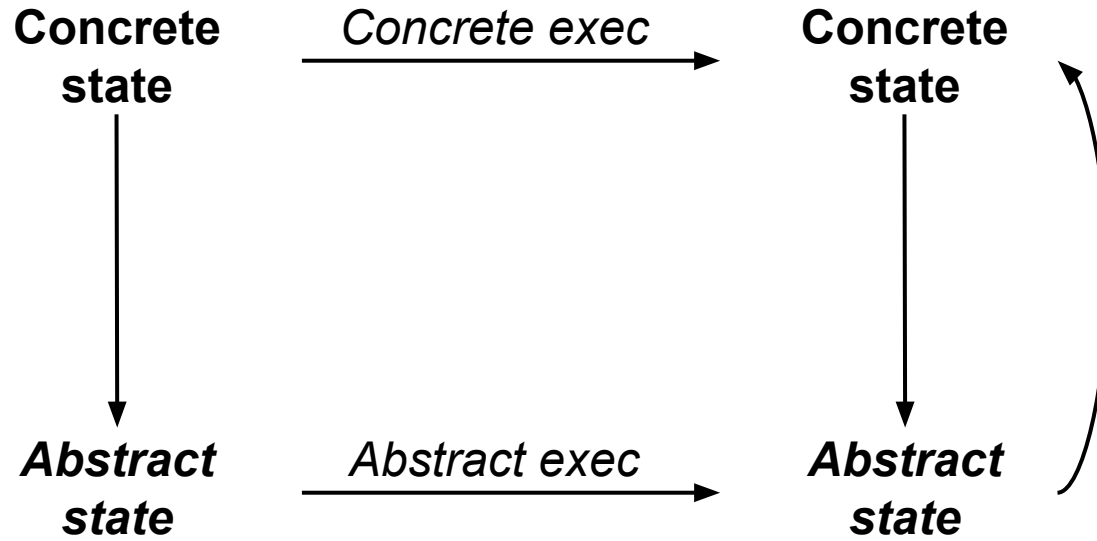
Concretization function

Concrete ($P(\mathbb{N})$)

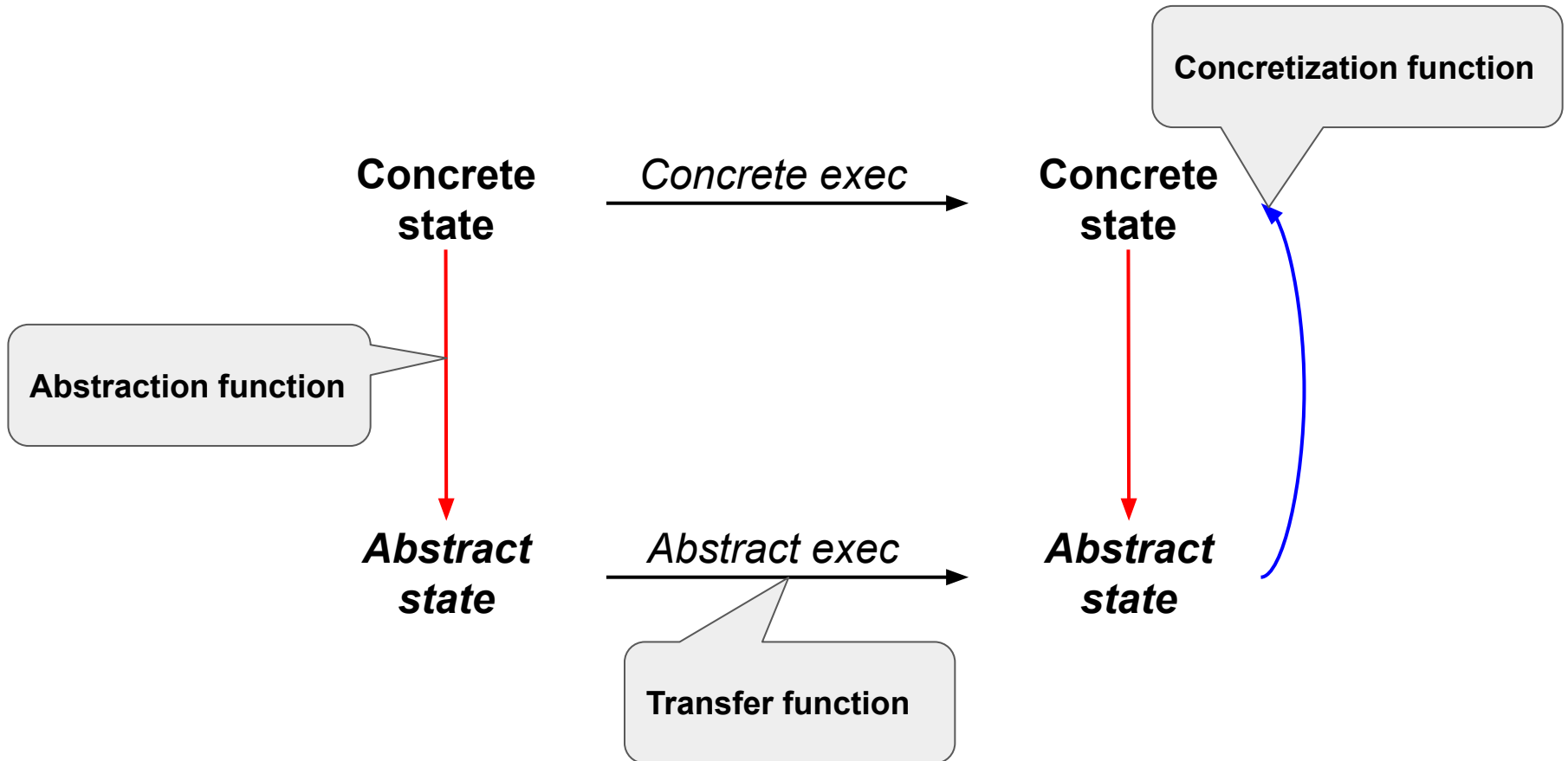
Abstract



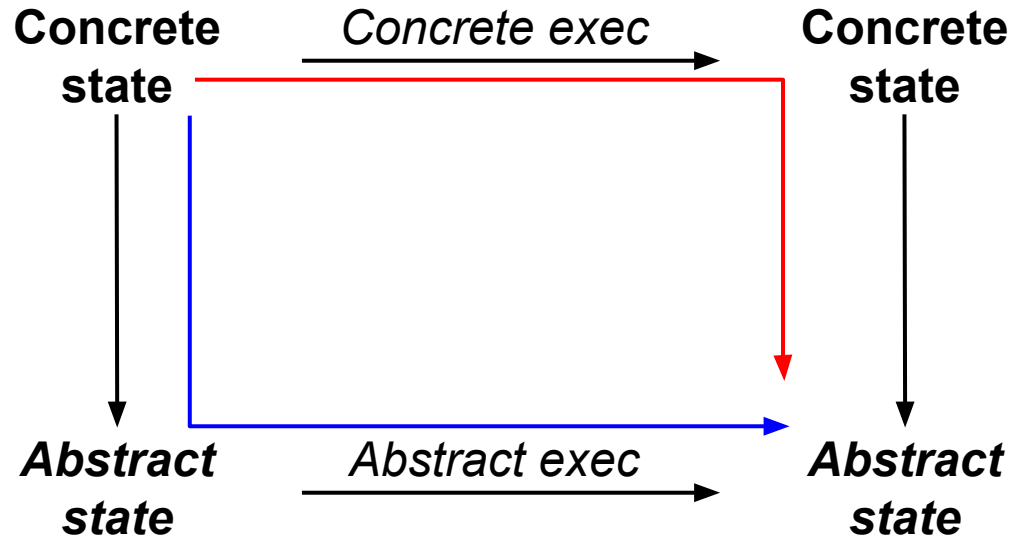
Transfer function



Transfer function

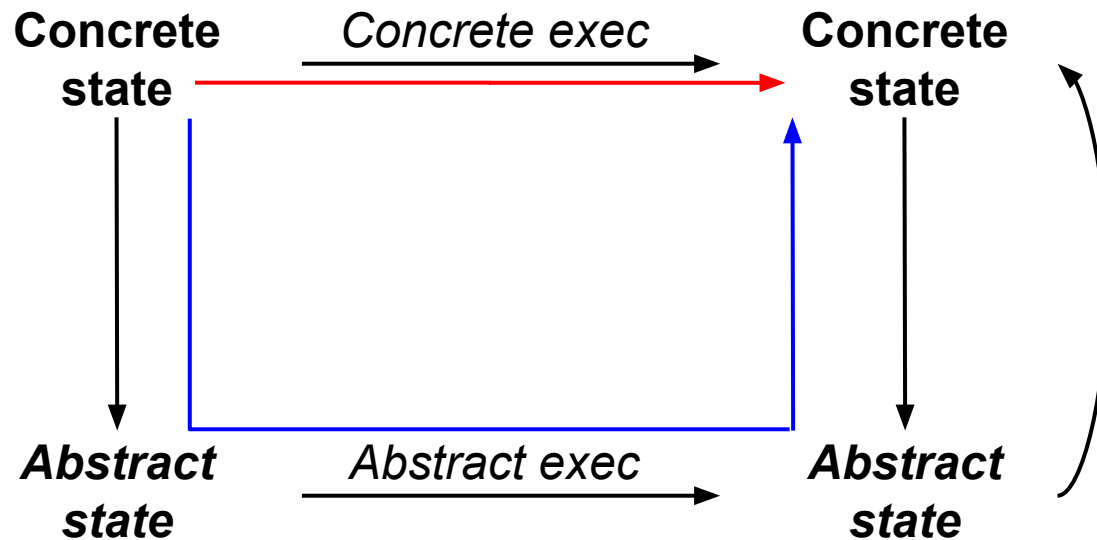


Abstract interpretation: approximation



Do both paths lead to the same abstract state?

Abstract interpretation: approximation



Do both paths lead to the same concrete state?

Abstract interpretation: soundness example

Abstract domain: $\{odd, even_2, even_4, is2, unk\}$

x=16 $\xrightarrow{16 / 2}$

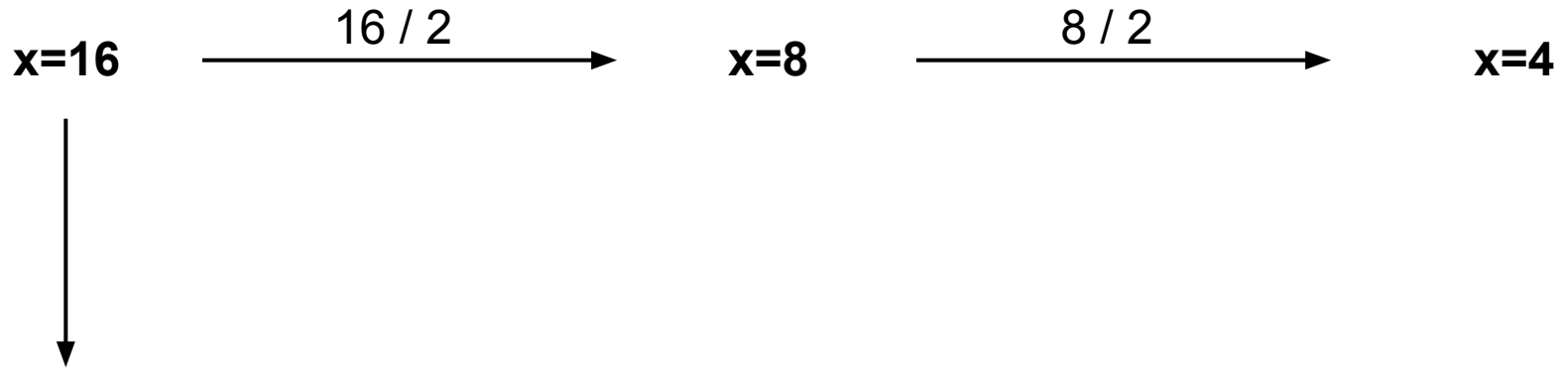
Abstract interpretation: soundness example

Abstract domain: $\{odd, even_2, even_4, is2, unk\}$



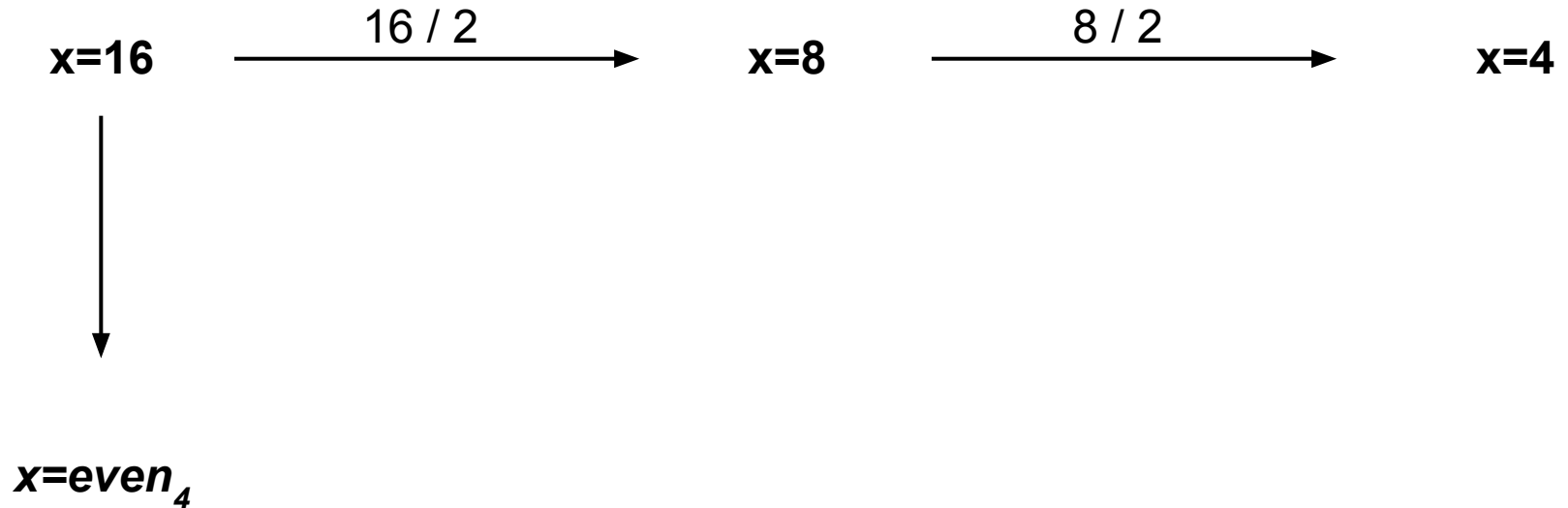
Abstract interpretation: soundness example

Abstract domain: $\{odd, even_2, even_4, is2, unk\}$



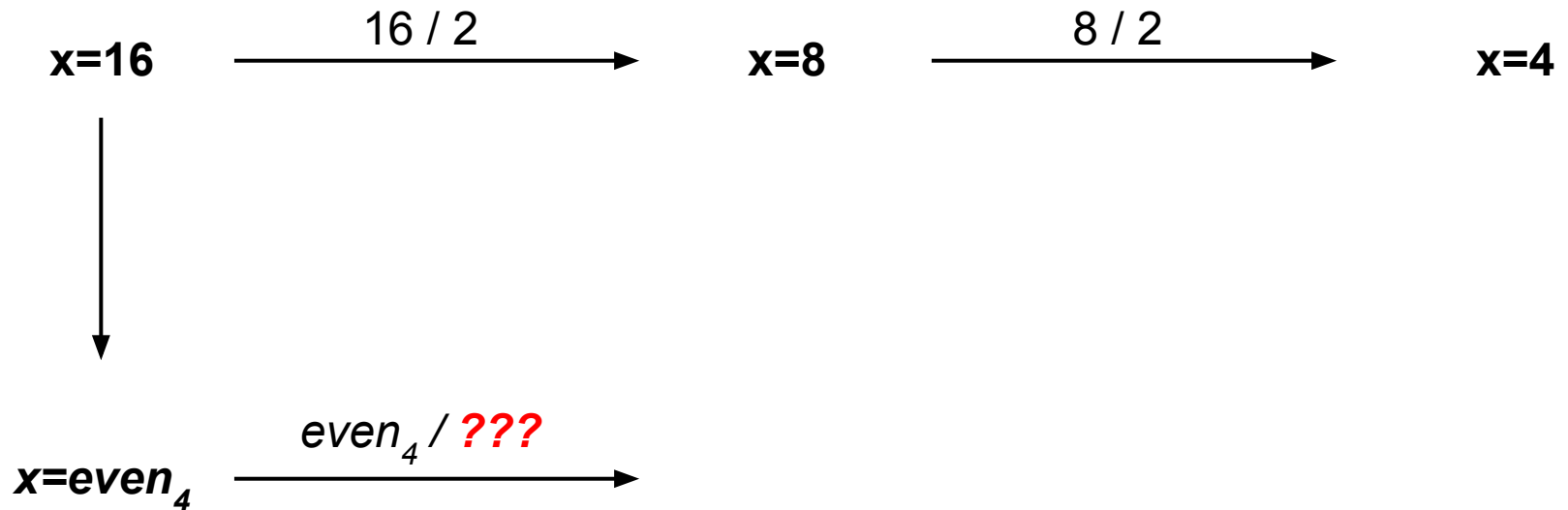
Abstract interpretation: soundness example

Abstract domain: $\{odd, even_2, even_4, is2, unk\}$



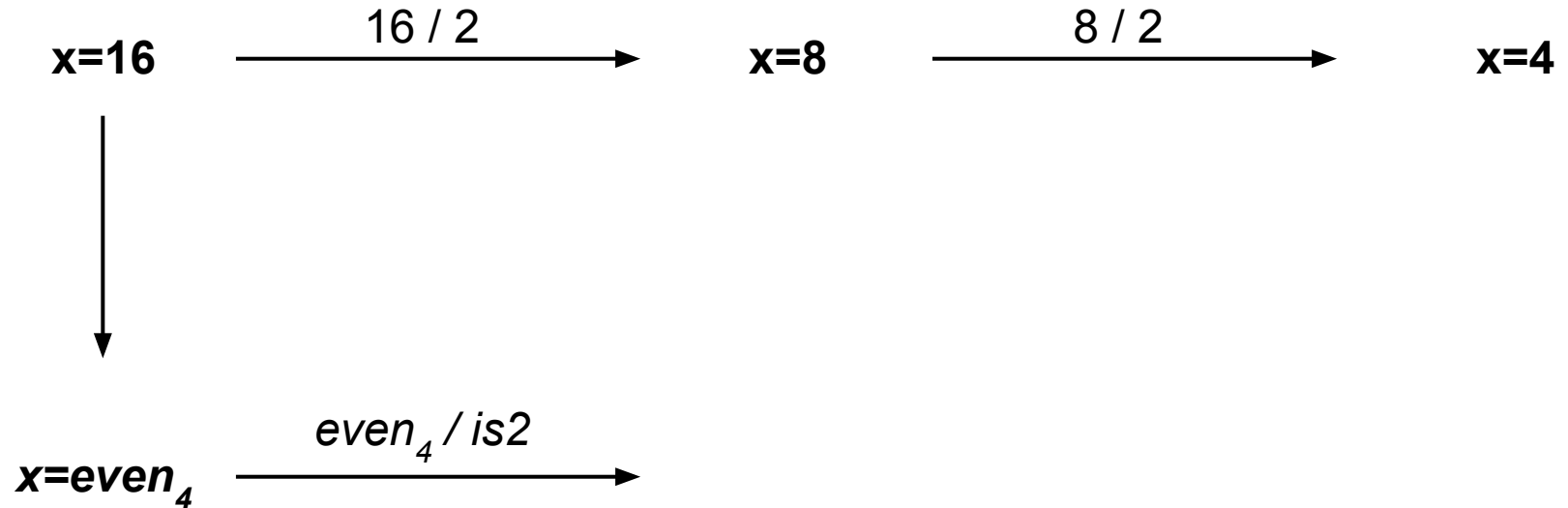
Abstract interpretation: soundness example

Abstract domain: $\{odd, even_2, even_4, is2, unk\}$



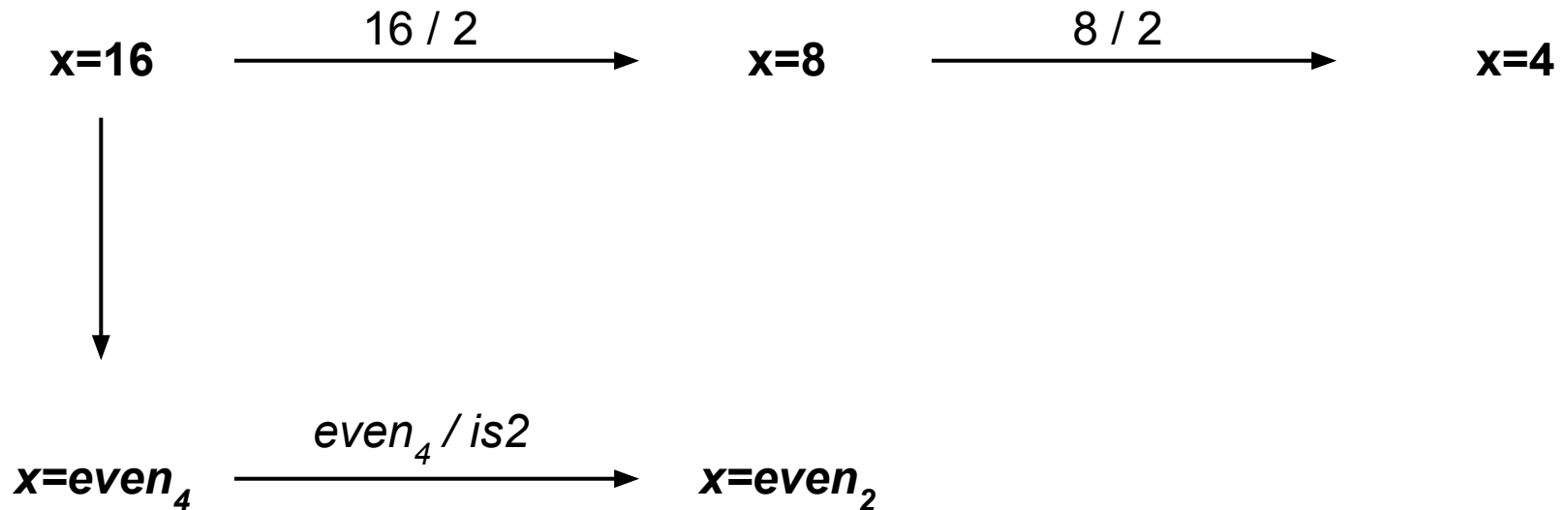
Abstract interpretation: soundness example

Abstract domain: $\{\text{odd}, \text{even}_2, \text{even}_4, \text{is2}, \text{unk}\}$



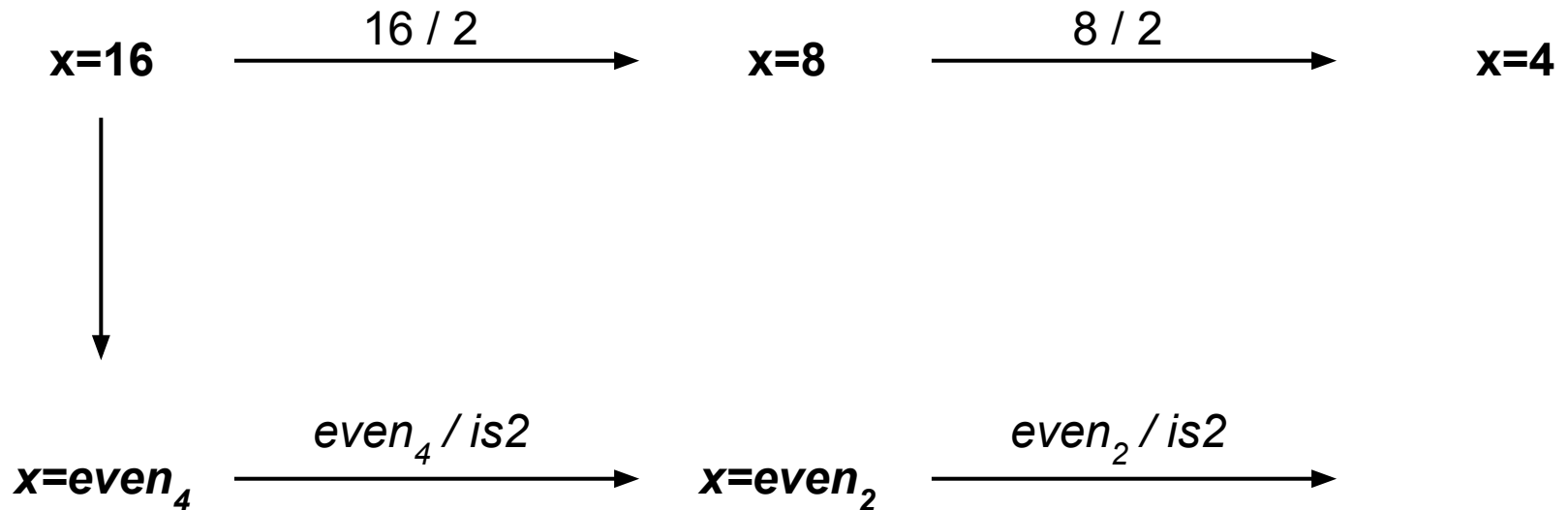
Abstract interpretation: soundness example

Abstract domain: $\{\text{odd}, \text{even}_2, \text{even}_4, \text{is2}, \text{unk}\}$



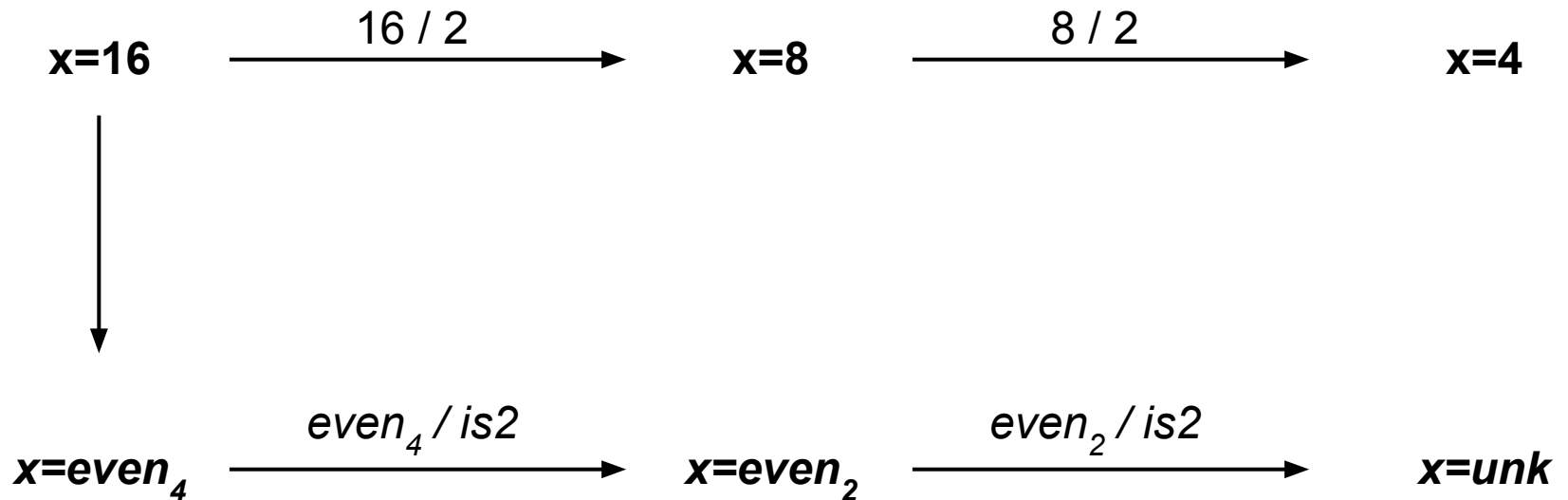
Abstract interpretation: soundness example

Abstract domain: $\{\text{odd}, \text{even}_2, \text{even}_4, \text{is2}, \text{unk}\}$



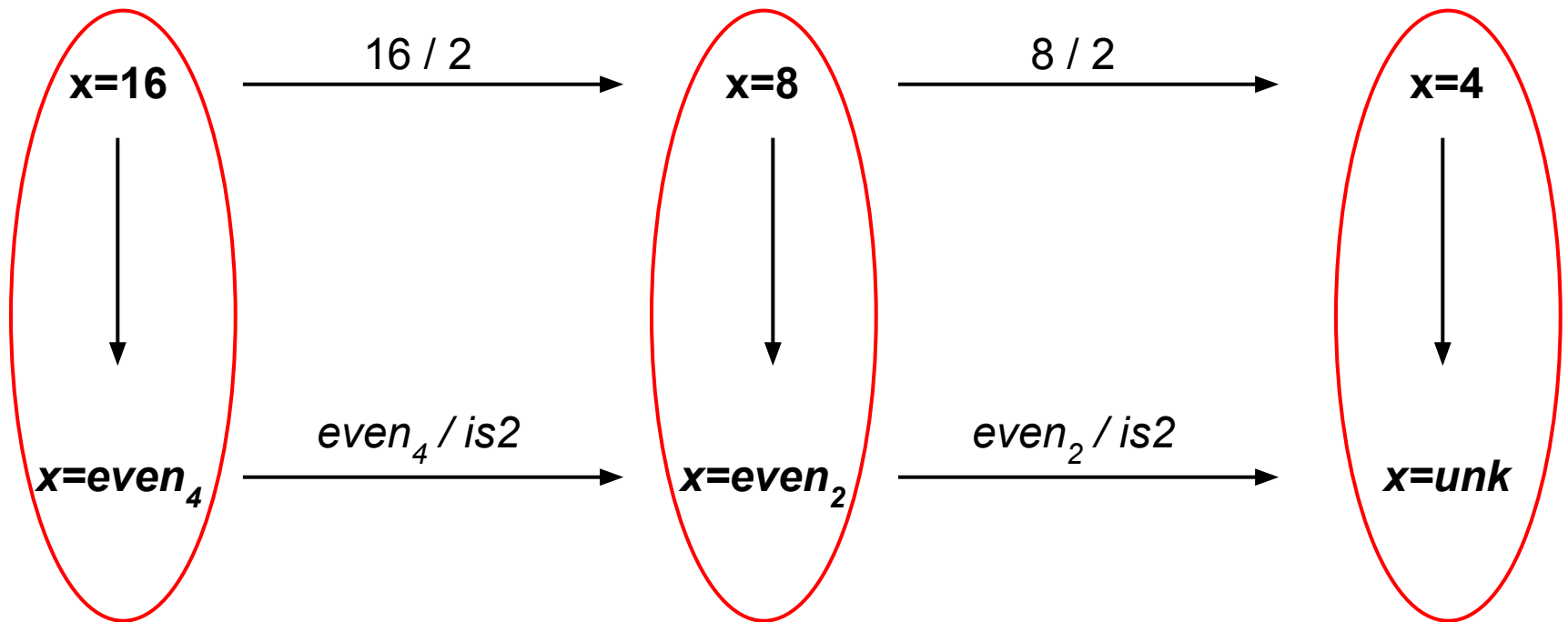
Abstract interpretation: soundness example

Abstract domain: $\{\text{odd}, \text{even}_2, \text{even}_4, \text{is2}, \text{unk}\}$



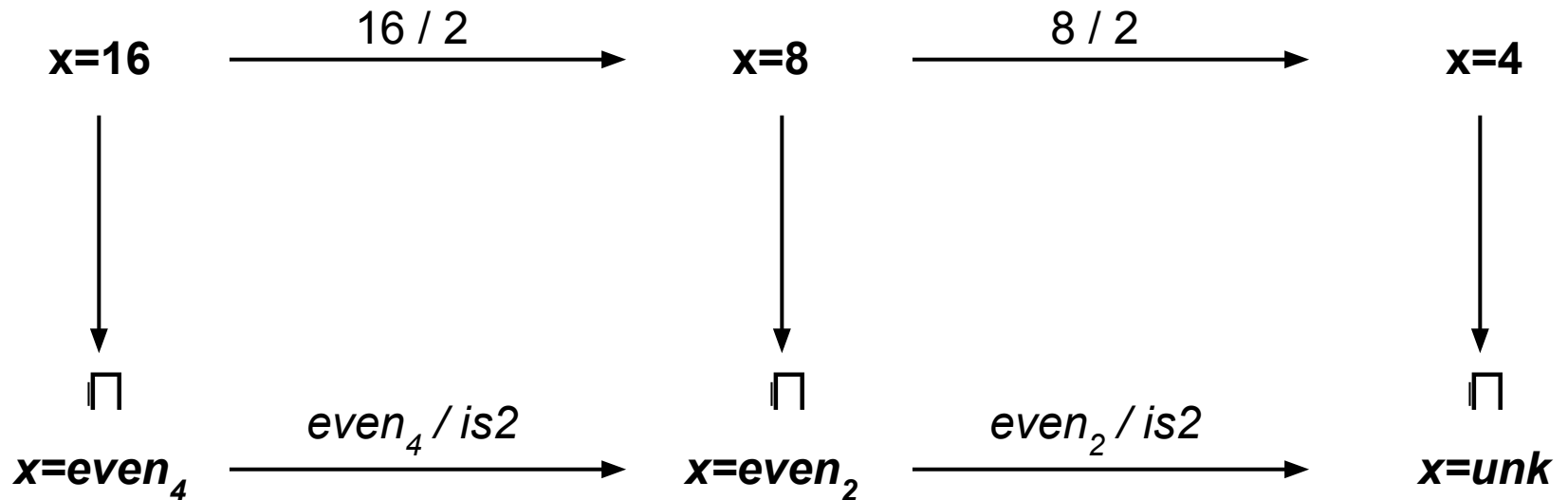
Abstract interpretation: soundness example

Abstract domain: $\{\text{odd}, \text{even}_2, \text{even}_4, \text{is2}, \text{unk}\}$

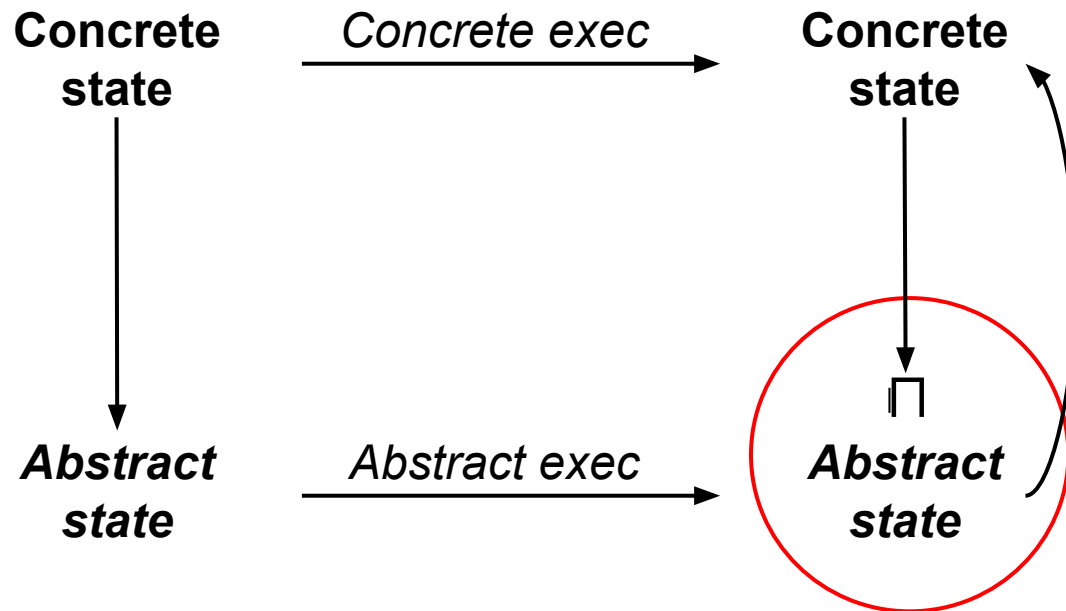


Abstract interpretation: soundness example

Abstract domain: $\{\text{odd}, \text{even}_2, \text{even}_4, \text{is2}, \text{unk}\}$

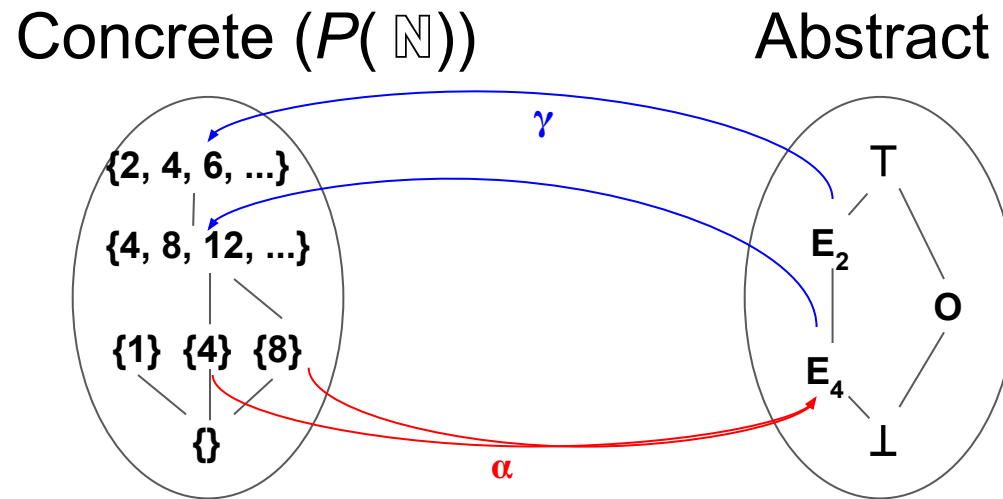


Abstract interpretation: soundness



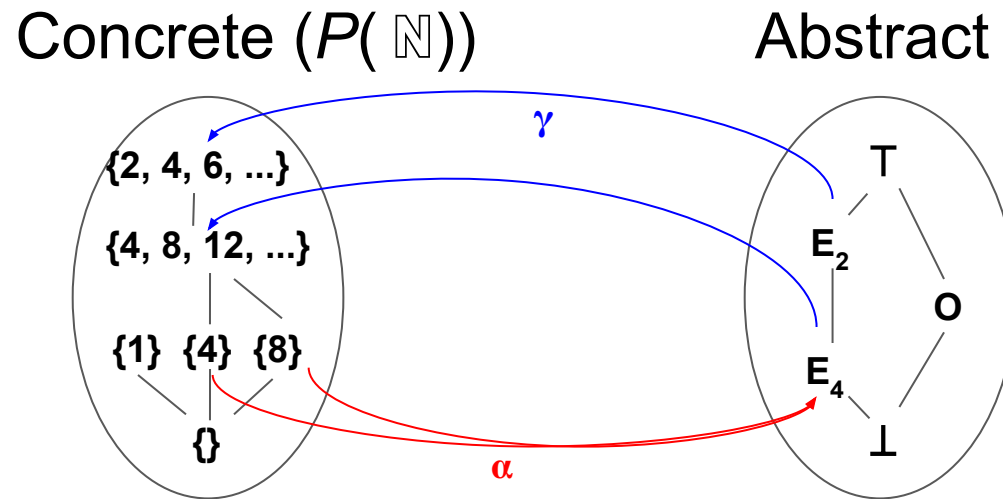
What properties must be satisfied by the abstraction, concretization, and transfer functions?

Sound approximation: properties



What properties must α and γ satisfy?

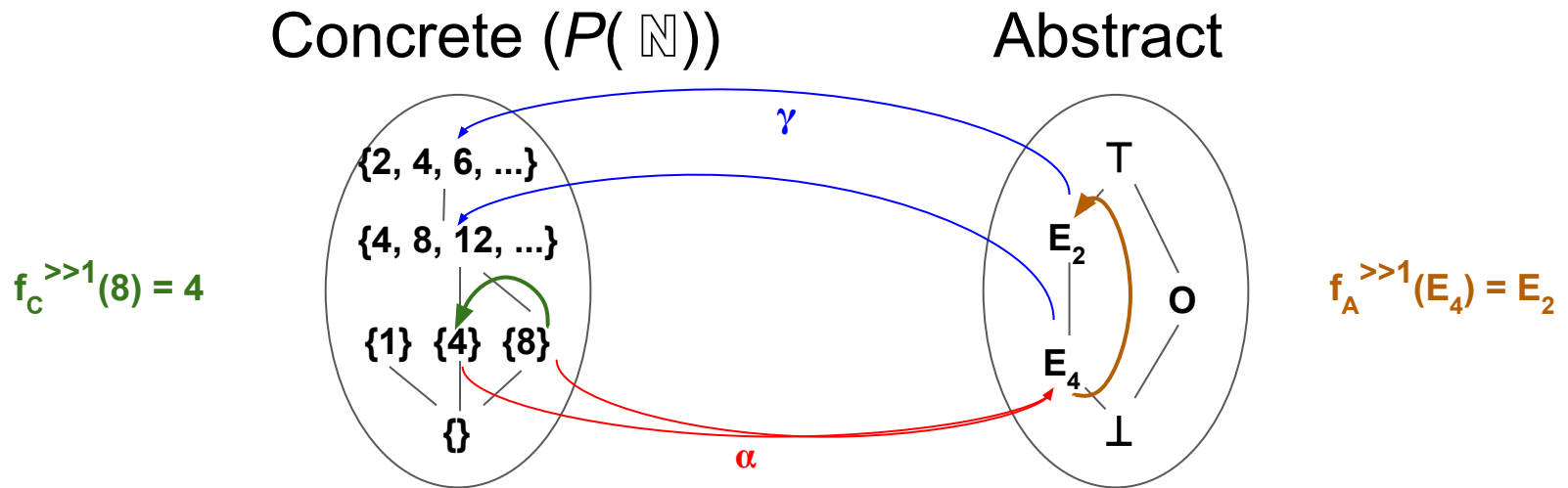
Sound approximation: galois connection



Galois connection

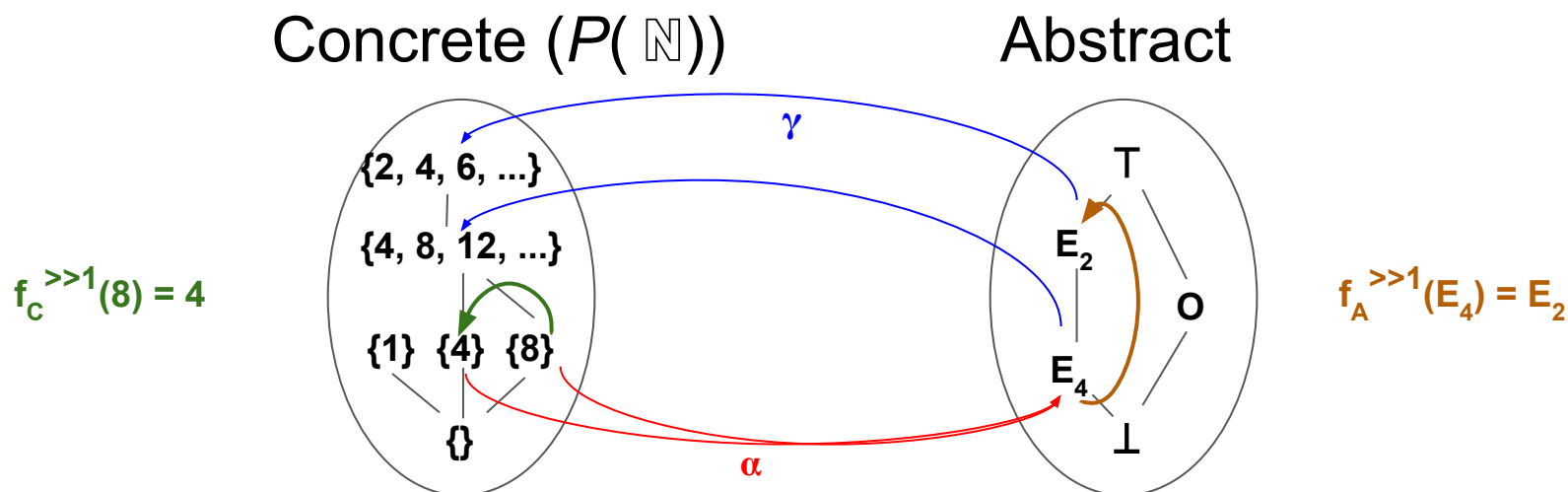
- $\alpha: C \rightarrow A$
- $\gamma: A \rightarrow C$
- $\forall c \in C: c \leq \gamma(\alpha(c))$

Sound approximation: properties



What properties must the transfer function(s) satisfy?

Sound approximation: consistency



Transfer function

- Consistent with concrete execution
 - c : concrete state; $c' = f_C(c)$
 - a : $\alpha(c)$
 - $a' = f_A(a)$
 - $c'' = \gamma(a')$
 - $c' \leq c''$

Sound approximation: properties

Transfer function

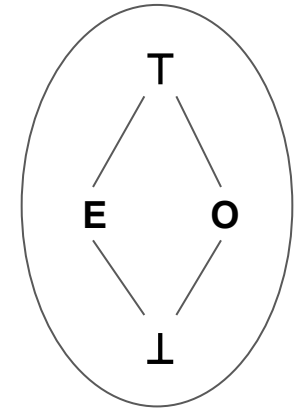
- $f_A^+ : A \times A \rightarrow A$

Lub

- $\text{lub} : A \times A \rightarrow A$

+	E	O	T	...
E	E	O	T	
O	O	E	T	
T	T	T	T	
...				

$$\text{lub}(E, O) = T$$



What properties must the lub function satisfy?

Sound approximation: monotonicity

Transfer function

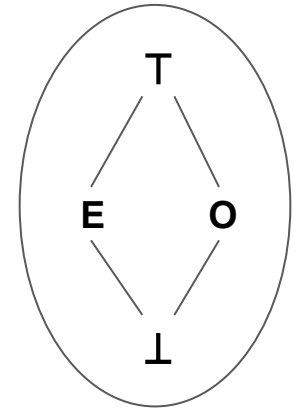
- $f_A^+ : A \times A \rightarrow A$
- may not be monotone

Lub

- $\text{lub} : A \times A \rightarrow A$
- must be monotone

+	E	O	T	...
E	E	O	T	
O	O	E	T	
T	T	T	T	
...				

$$\text{lub}(E, O) = T$$



Sound approximation: join (lub) vs. meet (glb)

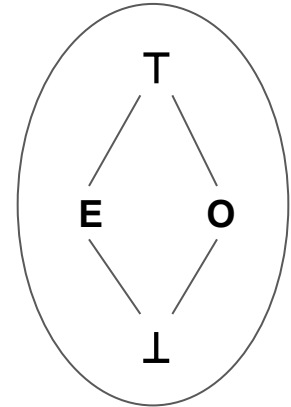
Transfer function

- f_A^+ : $A \times A \rightarrow A$
- may not be monotone

Lub

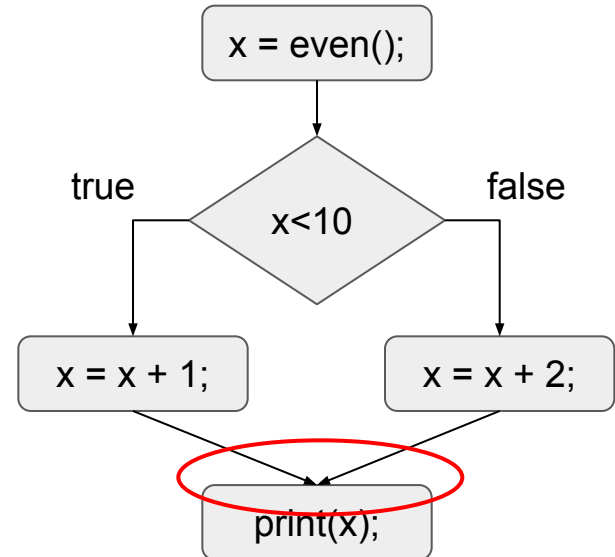
- $\text{lub}: A \times A \rightarrow A$
- must be monotone

+	E	O	T	...
E	E	O	T	
O	O	E	T	
T	T	T	T	
...				



$$\text{lub}(E, O) = T$$

```
int x = even();  
  
if (x < 10) {  
    x = x + 1;  
} else {  
    x = x + 2;  
}  
print(x);
```



Small-group exercise



- Work through two examples:

- Join vs. meet operation ($f(\text{int } a, \text{int } b, \text{int } c): \text{int}$)

```
if (cond) {
  x = a * b;
} else {
  x = a * c;
}
return(x);
```

Which parameters (a, b, c)

- will definitely be used?
- may be used?

(cond is independent of the parameters)

- Termination/fix point iteration

```
int x = 2;
while (x < 10) {
  x = x + 2;
}
```

Is the value of x after the loop an even number? Use an abstract domain with $\{\text{odd}, 2, \text{even}_2, \text{and } \text{even}_4\}$

Small-group exercise



- Work through two examples:

- Join vs. meet operation ($f(\text{int } a, \text{int } b, \text{int } c): \text{int}$)

```
if (cond) {
  x = a * b;
} else {
  x = a * c;
}
return(x);
```

Which parameters (a, b, c)

- will definitely be used?
- may be used?

(cond is independent of the parameters)

- Termination/fix point iteration

```
int x = 2;
while (x < 10) {
  x = x + 2;
}
```

Is the value of x after the loop an even number? Use an abstract domain with $\{\text{odd}, 2, \text{even}_2, \text{and } \text{even}_4\}$

See Q&A write-up:

<https://docs.google.com/document/d/1VEWmFIJVtD2F9ZkXIZ9xeOXGAtkRZATIX13wc1NYmtw>

CheckerFramework live demo