Gentle introduction to abstract interpretation

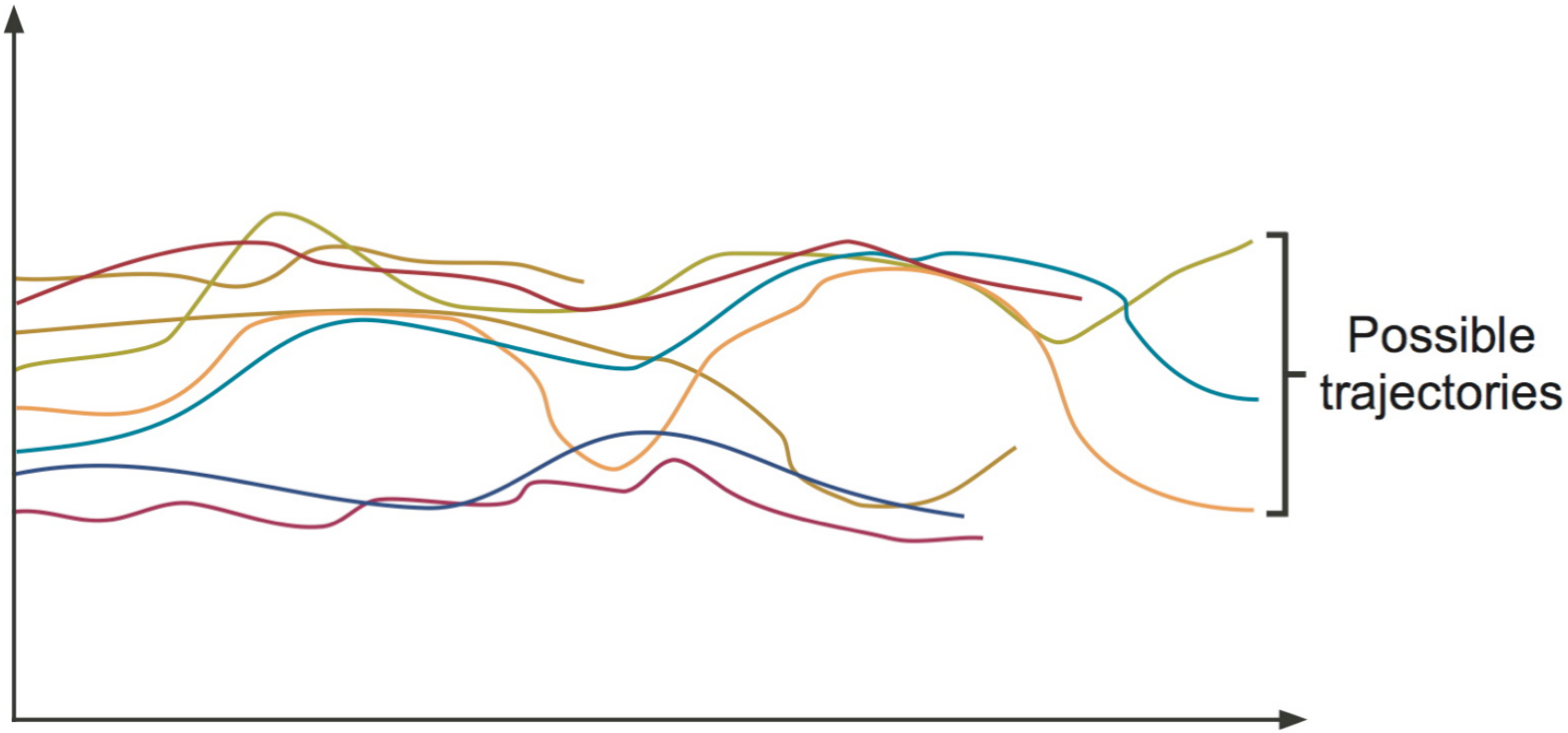
Henrich Lauko

September 25, 2017

Motivation

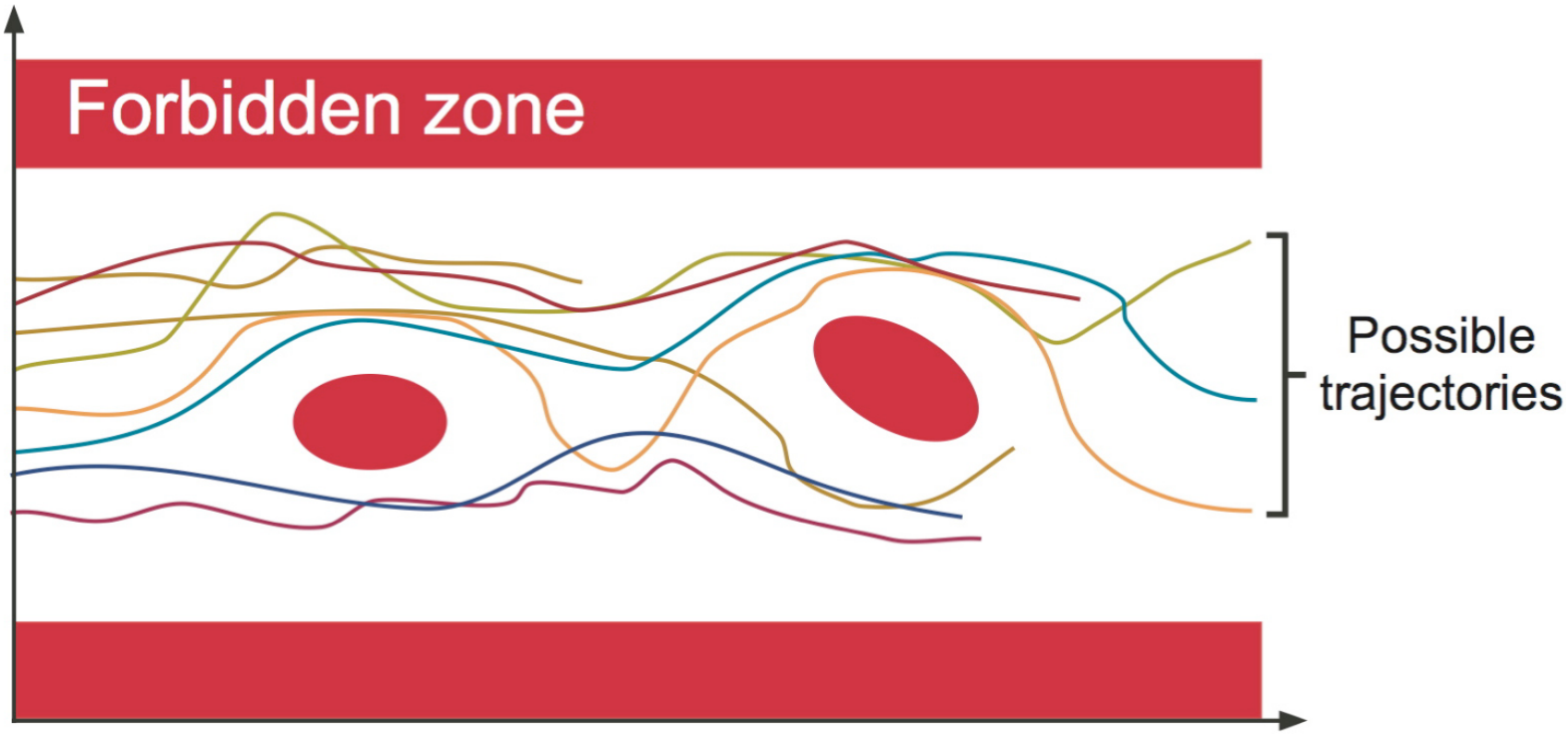
- Often impossible to compute the set of reachable states precisely
- Lets compute them on some level of abstraction

Traces of program

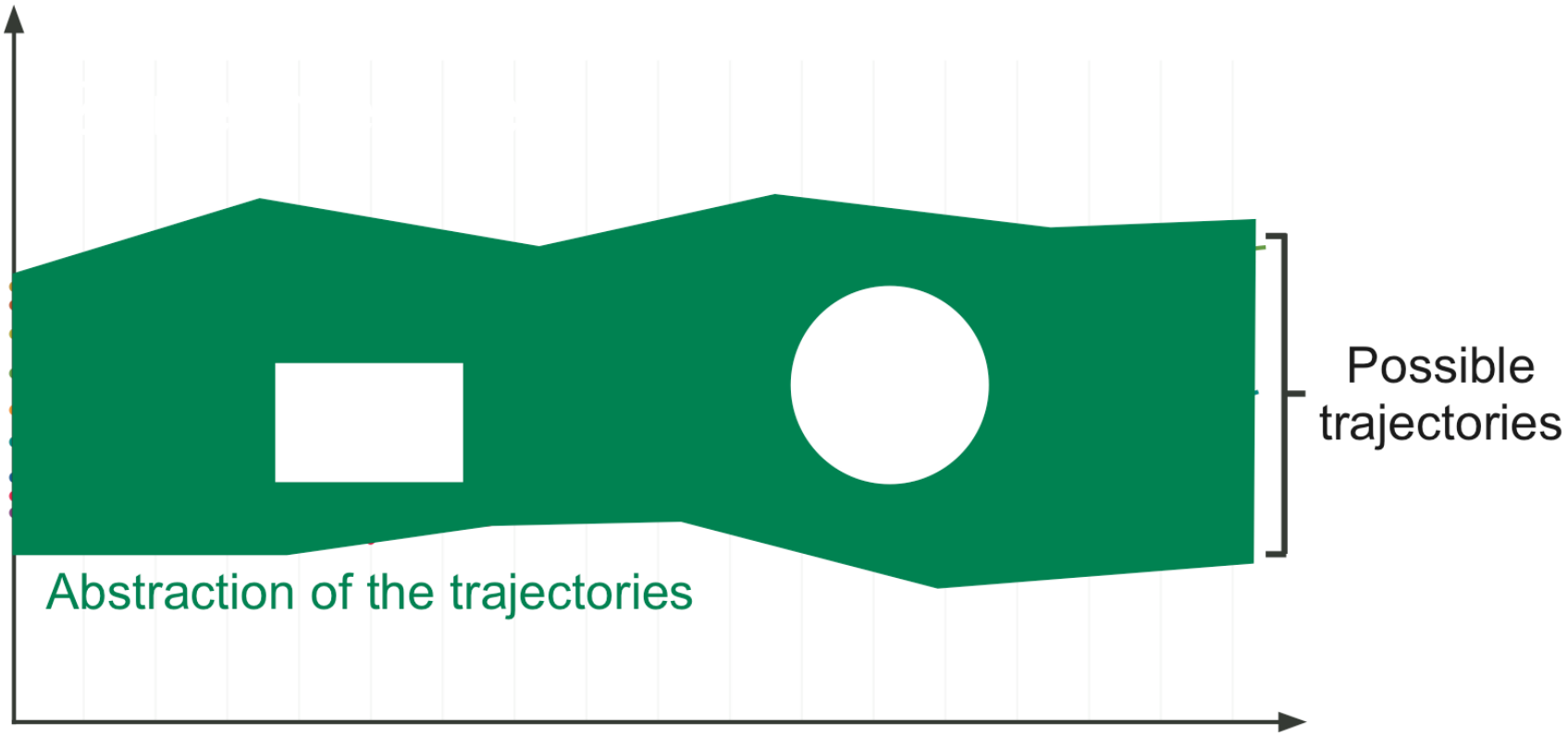


- Evolution of program state through time

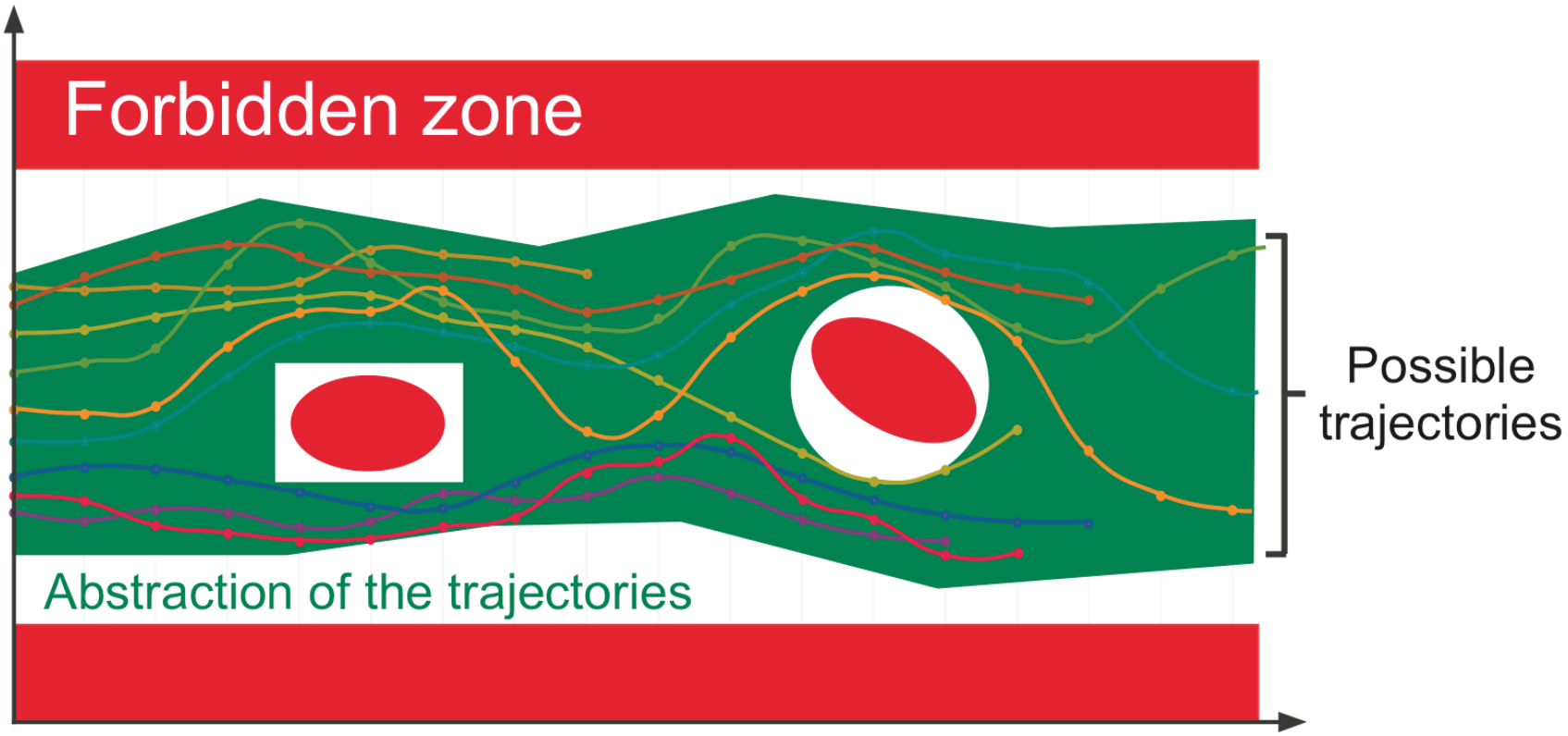
Error states



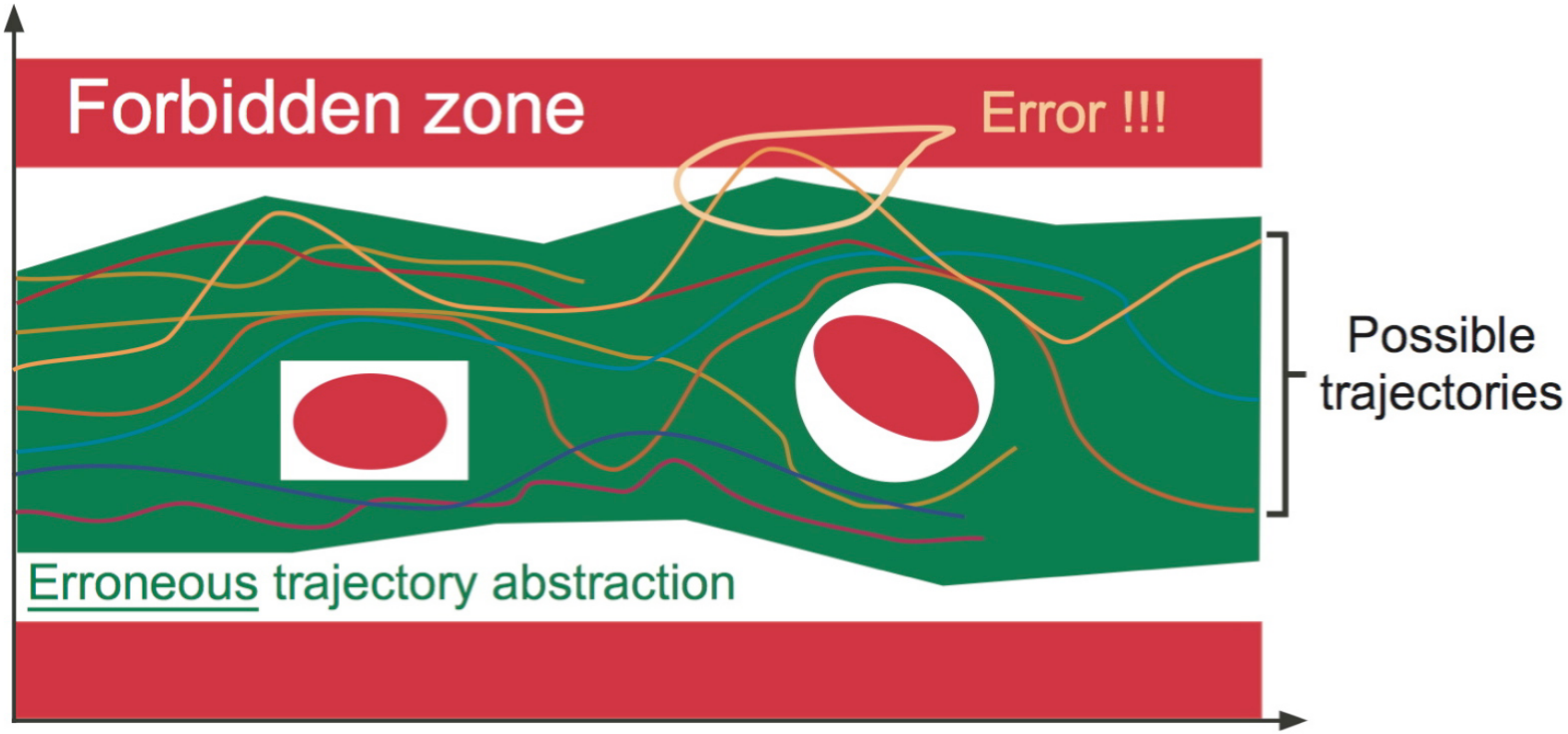
Abstraction of trajectories



Abstraction of trajectories

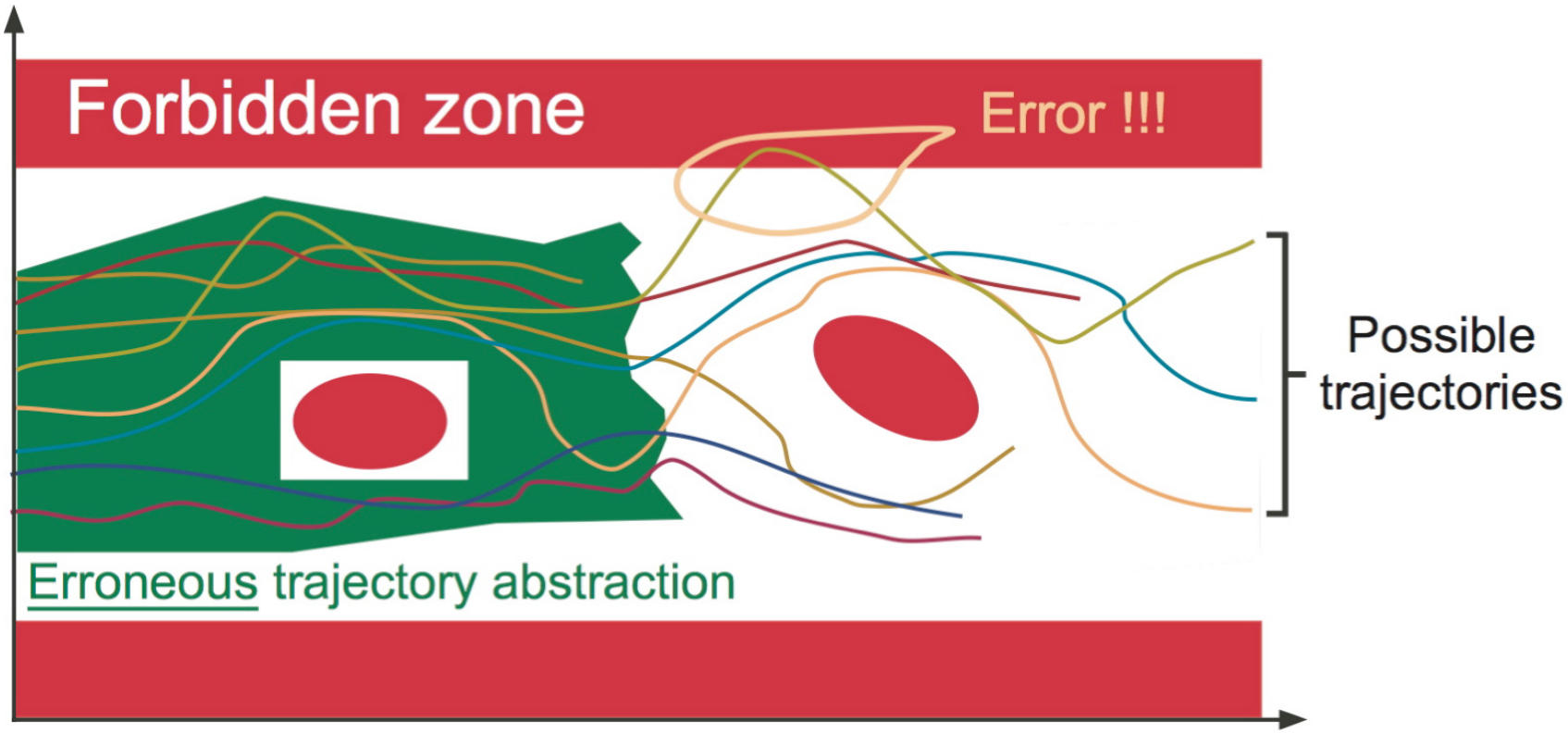


Soundness of abstract interpretation



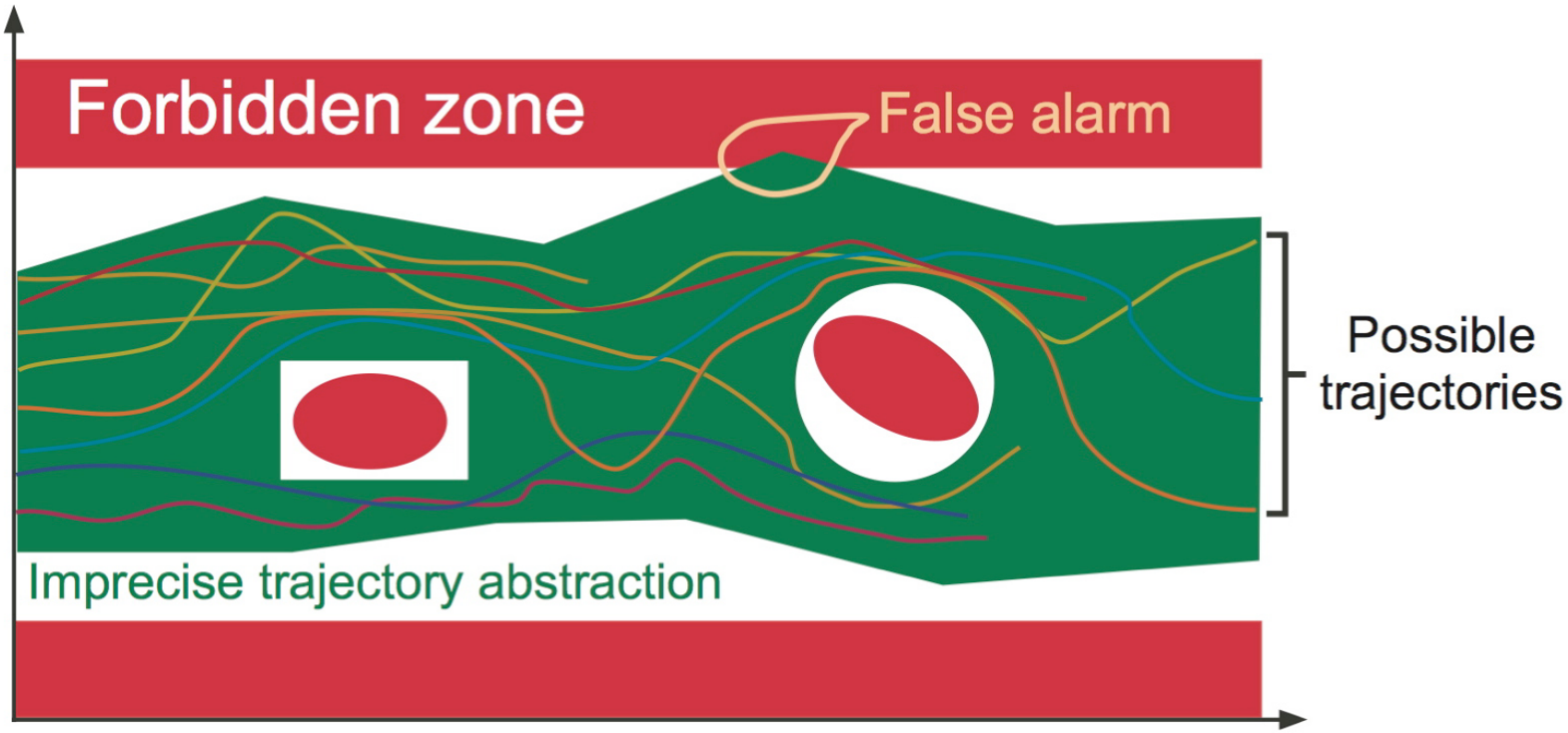
- We want to exclude unsound abstractions

Soundness of abstract interpretation



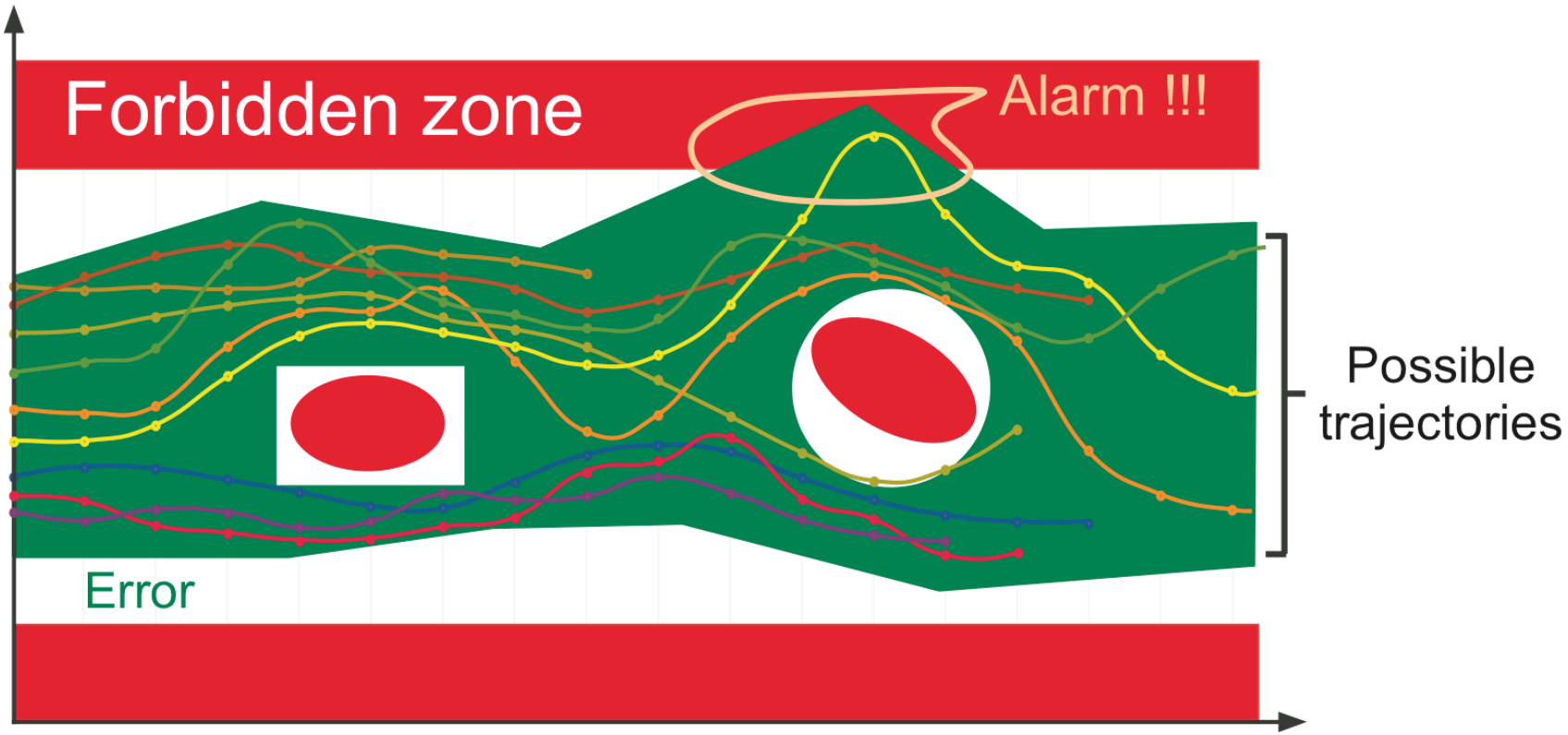
- Bounded model-checking

Over-approximation



- False positive

Good abstraction



Preliminaries

1. Partially ordered sets and lattices
2. Monotone functions
3. Fixpoint computation
4. Strongest postcondition

Abstract intersections and unions

Partially ordered set $(\mathrm{P}, \leqslant)$, where $\leqslant$ is an ordering on P.

Abstract intersections and unions

Partially ordered set $(\mathrm{P}, \leqslant)$, where $\leqslant$ is an ordering on P.

Intuition
$\mathfrak{a} \leqslant \mathfrak{b}$ if $\mathfrak{a}$ contains more information than $\mathfrak{b}$

Abstract intersections and unions

Partially ordered set $(P, \leqslant)$, where $\leqslant$ is an ordering on P.

Intuition
$a \leqslant b$ if a contains more information than b

$$[5, 10] \leqslant [0, 15]$$

Abstract intersections and unions

Partially ordered set $(P, \leqslant)$, where $\leqslant$ is an ordering on P.

Intuition
$a \leqslant b$ if a contains more information than b

$$[5, 10] \leqslant [0, 15]$$

Union $\sim$ Supremum (Join)

a b

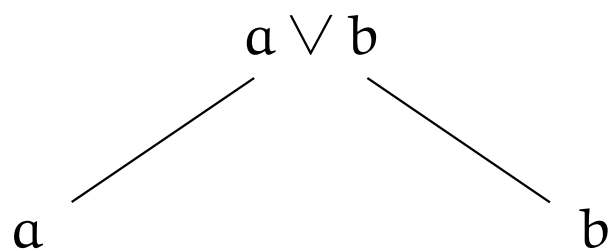
Abstract intersections and unions

Partially ordered set $(P, \leqslant)$, where $\leqslant$ is an ordering on P.

Intuition
$a \leqslant b$ if a contains more information than b

$$[5, 10] \leqslant [0, 15]$$

Union $\sim$ Supremum (Join)



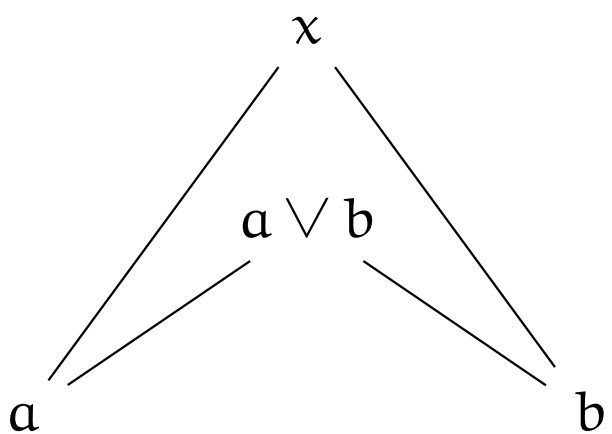
Abstract intersections and unions

Partially ordered set $(P, \leqslant)$, where $\leqslant$ is an ordering on P.

Intuition
$a \leqslant b$ if a contains more information than b

$$[5, 10] \leqslant [0, 15]$$

Union $\sim$ Supremum (Join)



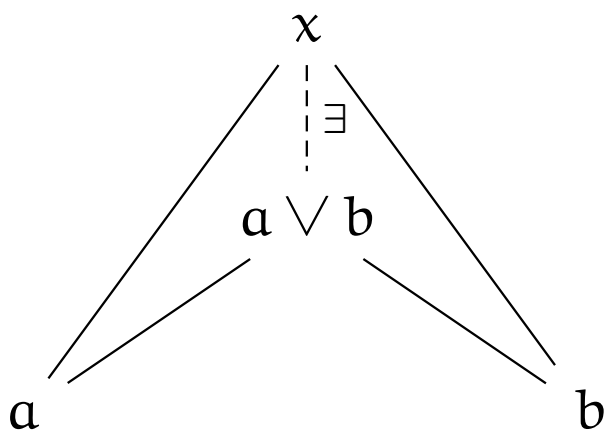
Abstract intersections and unions

Partially ordered set $(P, \leqslant)$, where $\leqslant$ is an ordering on P.

Intuition
$a \leqslant b$ if a contains more information than b

$$[5, 10] \leqslant [0, 15]$$

Union $\sim$ Supremum (Join)



Abstract intersections and unions

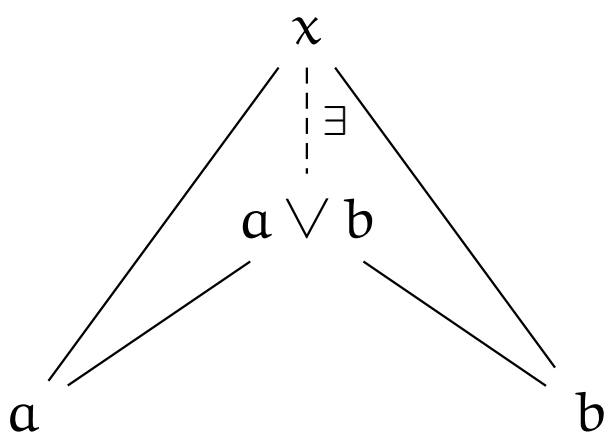
Partially ordered set $(P, \leqslant)$, where $\leqslant$ is an ordering on P.

Intuition

$a \leqslant b$ if a contains more information than b

$$[5, 10] \leqslant [0, 15]$$

Union $\sim$ Supremum (Join)



$$[1, 3] \vee [2, 4] = ???$$

Abstract intersections and unions

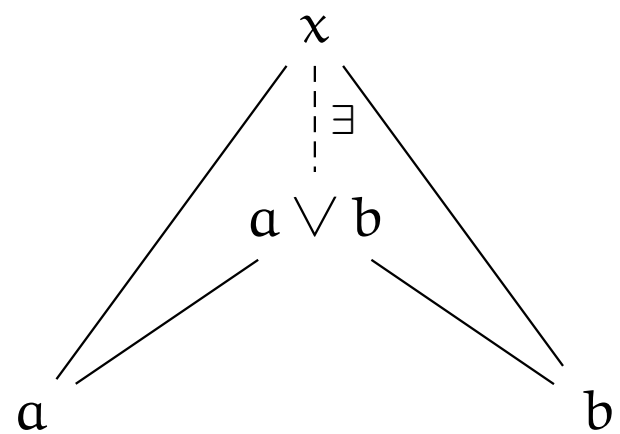
Partially ordered set $(P, \leqslant)$, where $\leqslant$ is an ordering on P.

Intuition

$a \leqslant b$ if a contains more information than b

$$[5, 10] \leqslant [0, 15]$$

Union $\sim$ Supremum (Join)



$$[1, 3] \vee [2, 4] = [1, 4]$$

Abstract intersections and unions

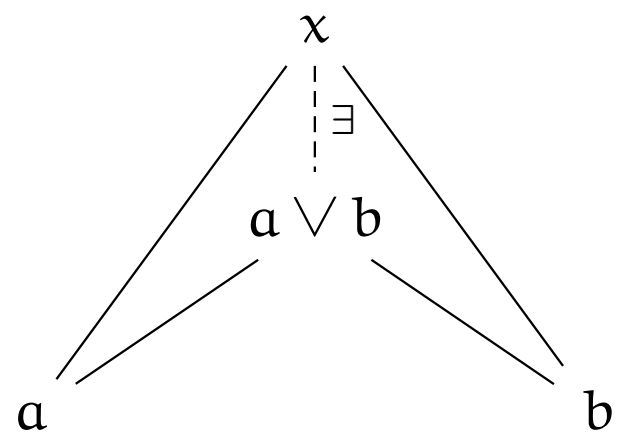
Partially ordered set $(P, \leqslant)$, where $\leqslant$ is an ordering on P.

Intuition

$a \leqslant b$ if a contains more information than b

$$[5, 10] \leqslant [0, 15]$$

Union $\sim$ Supremum (Join)



$$[1, 3] \vee [2, 4] = [1, 4]$$

Intersection $\sim$ Infimum (Meet)

a b

Abstract intersections and unions

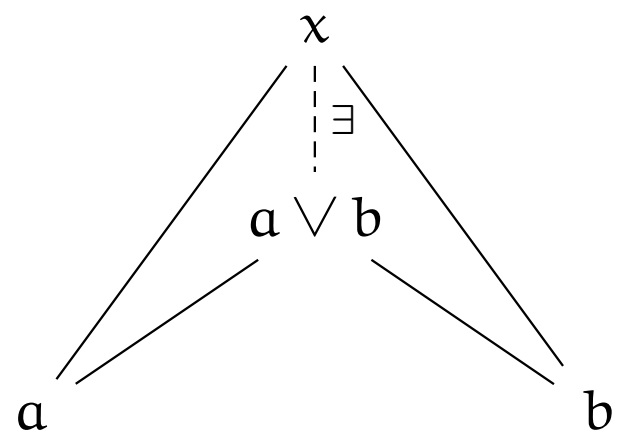
Partially ordered set $(P, \leqslant)$, where $\leqslant$ is an ordering on P.

Intuition

$a \leqslant b$ if a contains more information than b

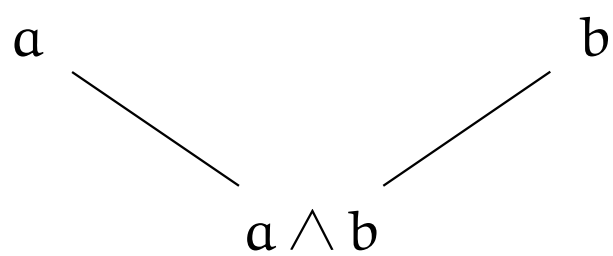
$$[5, 10] \leqslant [0, 15]$$

Union ~ Supremum (Join)



$$[1, 3] \vee [2, 4] = [1, 4]$$

Intersection ~ Infimum (Meet)



Abstract intersections and unions

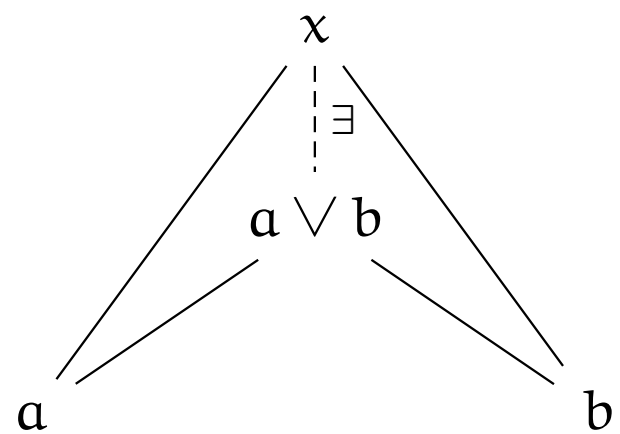
Partially ordered set $(P, \leqslant)$, where $\leqslant$ is an ordering on P.

Intuition

$a \leqslant b$ if a contains more information than b

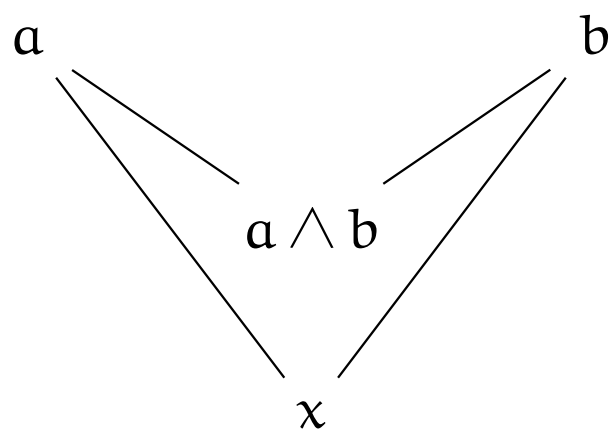
$$[5, 10] \leqslant [0, 15]$$

Union ~ Supremum (Join)



$$[1, 3] \vee [2, 4] = [1, 4]$$

Intersection ~ Infimum (Meet)



Abstract intersections and unions

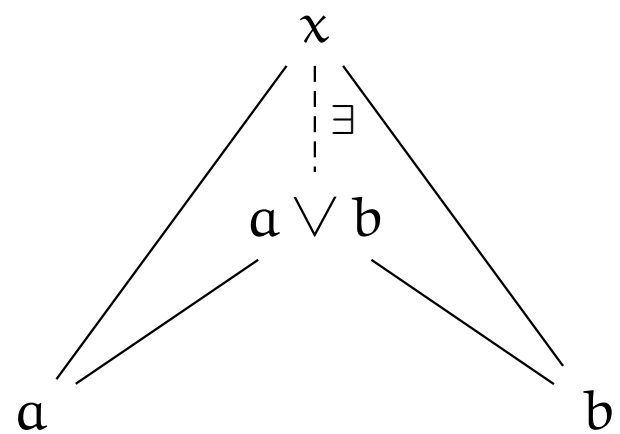
Partially ordered set $(P, \leqslant)$, where $\leqslant$ is an ordering on P.

Intuition

$a \leqslant b$ if a contains more information than b

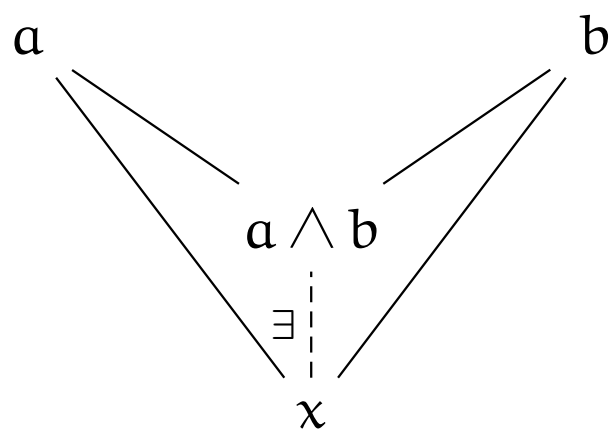
$$[5, 10] \leqslant [0, 15]$$

Union ~ Supremum (Join)



$$[1, 3] \vee [2, 4] = [1, 4]$$

Intersection ~ Infimum (Meet)



Abstract intersections and unions

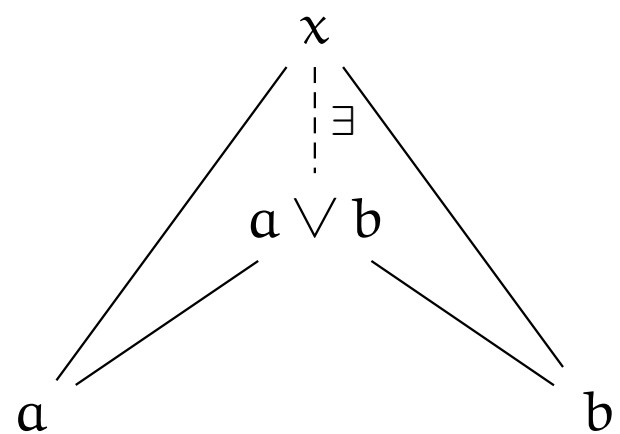
Partially ordered set $(P, \leqslant)$, where $\leqslant$ is an ordering on P.

Intuition

$a \leqslant b$ if a contains more information than b

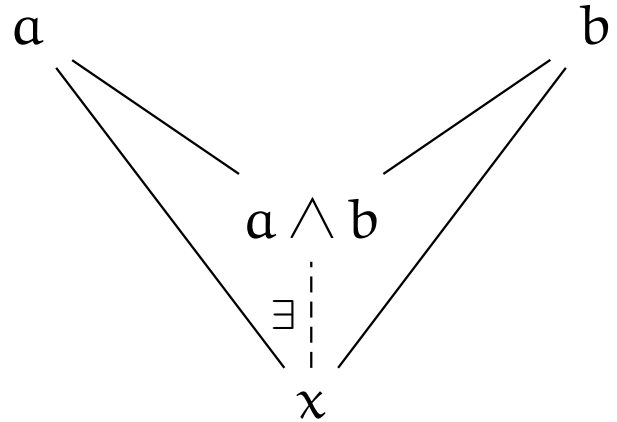
$$[5, 10] \leqslant [0, 15]$$

Union ~ Supremum (Join)



$$[1, 3] \vee [2, 4] = [1, 4]$$

Intersection ~ Infimum (Meet)



$$[1, 3] \wedge [2, 4] = ???$$

Abstract intersections and unions

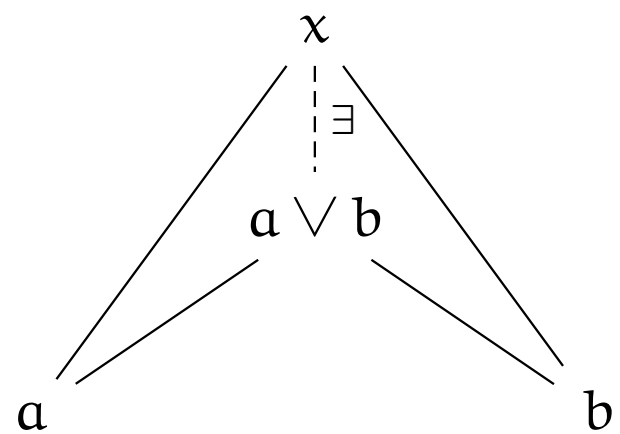
Partially ordered set $(P, \leqslant)$, where $\leqslant$ is an ordering on P.

Intuition

$a \leqslant b$ if a contains more information than b

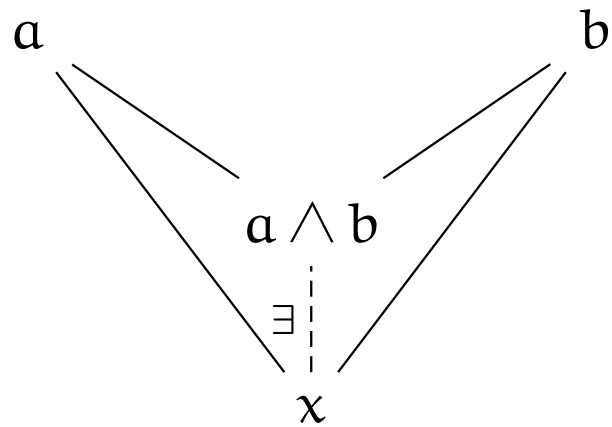
$$[5, 10] \leqslant [0, 15]$$

Union ~ Supremum (Join)



$$[1, 3] \vee [2, 4] = [1, 4]$$

Intersection ~ Infimum (Meet)



$$[1, 3] \wedge [2, 4] = [2, 3]$$

Lattice

A poset in which all pairs of elements have the supremum and the infimum is called a lattice.

Lattice

A poset in which all pairs of elements have the supremum and the infimum is called a lattice.

Are the following posets lattices?

- $(\mathbb{Z}, \leqslant)$

Lattice

A poset in which all pairs of elements have the supremum and the infimum is called a lattice.

Are the following posets lattices?

- $(\mathbb{Z}, \leqslant)$
- $(2^A, \subseteq)$ for a set A

Lattice

A poset in which all pairs of elements have the supremum and the infimum is called a lattice.

Are the following posets lattices?

- $(\mathbb{Z}, \leqslant)$
- $(2^A, \subseteq)$ for a set A
- $(\mathrm{Int}, \subseteq)$ for a set of intervals in $\mathrm{Int} = \{[a, b] \mid a, b \in \mathbb{Z}, a \leqslant b\}$

Lattice

A poset in which all pairs of elements have the supremum and the infimum is called a lattice.

Are the following posets lattices?

- $(\mathbb{Z}, \leqslant)$
- $(2^A, \subseteq)$ for a set A
- $(\mathrm{Int}, \subseteq)$ for a set of intervals in $\mathrm{Int} = \{[a, b] \mid a, b \in \mathbb{Z}, a \leqslant b\}$
- $(\mathrm{Int} \cup \{\emptyset\}, \subseteq)$

Lattice

A poset in which all pairs of elements have the supremum and the infimum is called a lattice.

Are the following posets lattices?

- $(\mathbb{Z}, \leqslant)$
- $(2^A, \subseteq)$ for a set A
- $(\mathrm{Int}, \subseteq)$ for a set of intervals in $\mathrm{Int} = \{[a, b] \mid a, b \in \mathbb{Z}, a \leqslant b\}$
- $(\mathrm{Int} \cup \{\emptyset\}, \subseteq)$
- $(\mathrm{Unit} \cup \{\emptyset\}, \subseteq)$ for a set Unit of unit intervals (i.e. $[a, a+1)$)

Complete lattice

A poset $(\mathrm{L}, \leqslant)$ is a complete lattice, if for each $\mathrm{M} \subseteq \mathrm{L}$ there exists both $\mathrm{sup}(\mathrm{M})$ and $\mathrm{inf}(\mathrm{M})$.

Complete lattice

A poset $(\mathrm{L}, \leqslant)$ is a complete lattice, if for each $\mathrm{M} \subseteq \mathrm{L}$ there exists both $\sup(\mathrm{M})$ and $\inf(\mathrm{M})$.

Are the following lattices complete lattices?

- $(\mathbb{Z}, \leqslant)$

Complete lattice

A poset $(\mathrm{L}, \leqslant)$ is a complete lattice, if for each $\mathrm{M} \subseteq \mathrm{L}$ there exists both $\sup(\mathrm{M})$ and $\inf(\mathrm{M})$.

Are the following lattices complete lattices?

- $(\mathbb{Z}, \leqslant)$
- $(\mathbb{Z} \cup \{-\infty, \infty\}, \leqslant)$

Complete lattice

A poset $(\mathrm{L}, \leqslant)$ is a complete lattice, if for each $\mathrm{M} \subseteq \mathrm{L}$ there exists both $\mathrm{sup}(\mathrm{M})$ and $\mathrm{inf}(\mathrm{M})$.

Are the following lattices complete lattices?

- $(\mathbb{Z}, \leqslant)$
- $(\mathbb{Z} \cup \{-\infty, \infty\}, \leqslant)$
- $(2^{\mathrm{A}}, \subseteq)$ for a set A

Complete lattice

A poset $(\mathrm{L}, \leqslant)$ is a complete lattice, if for each $\mathrm{M} \subseteq \mathrm{L}$ there exists both $\sup(\mathrm{M})$ and $\inf(\mathrm{M})$.

Are the following lattices complete lattices?

- $(\mathbb{Z}, \leqslant)$
- $(\mathbb{Z} \cup \{-\infty, \infty\}, \leqslant)$
- $(2^{\mathrm{A}}, \subseteq)$ for a set A
- $(\mathrm{Int} \cup \{\emptyset, (-\infty, \infty)\}, \subseteq)$

Complete lattice

A poset $(\mathrm{L}, \leqslant)$ is a complete lattice, if for each $\mathrm{M} \subseteq \mathrm{L}$ there exists both $\sup(\mathrm{M})$ and $\inf(\mathrm{M})$.

Are the following lattices complete lattices?

- $(\mathbb{Z}, \leqslant)$
- $(\mathbb{Z} \cup \{-\infty, \infty\}, \leqslant)$
- $(2^{\mathrm{A}}, \subseteq)$ for a set A
- $(\mathrm{Int} \cup \{\emptyset, (-\infty, \infty)\}, \subseteq)$

Complete lattice

A poset $(\mathrm{L}, \leqslant)$ is a complete lattice, if for each $\mathrm{M} \subseteq \mathrm{L}$ there exists both $\sup(\mathrm{M})$ and $\inf(\mathrm{M})$.

Are the following lattices complete lattices?

- $(\mathbb{Z}, \leqslant)$
- $(\mathbb{Z} \cup \{-\infty, \infty\}, \leqslant)$
- $(2^{\mathrm{A}}, \subseteq)$ for a set A
- $(\mathrm{Int} \cup \{\emptyset, (-\infty, \infty)\}, \subseteq) = \mathrm{Int}_{\mathrm{L}}$

Complete lattice

A poset $(\mathrm{L}, \leqslant)$ is a complete lattice, if for each $\mathrm{M} \subseteq \mathrm{L}$ there exists both $\sup(\mathrm{M})$ and $\inf(\mathrm{M})$.

Are the following lattices complete lattices?

- $(\mathbb{Z}, \leqslant)$
- $(\mathbb{Z} \cup \{-\infty, \infty\}, \leqslant)$
- $(2^{\mathrm{A}}, \subseteq)$ for a set A
- $(\mathrm{Int} \cup \{\emptyset, (-\infty, \infty)\}, \subseteq) = \mathrm{Int}_{\mathrm{L}}$

A complete lattice always has

- the greatest element ($\top$) called top
- the least element ($\bot$) called bottom

Monotonic function

Let $(P, \leqslant), (R, \sqsubseteq)$ are posets

- a function $f : P \to R$ is monotone if for all $x, y \in P$ it holds

$$x \leqslant y \implies f(x) \sqsubseteq f(y)$$

Monotonic function

Let $(P, \leqslant), (R, \sqsubseteq)$ are posets

- a function $f : P \to R$ is monotone if for all $x, y \in P$ it holds

$$x \leqslant y \implies f(x) \sqsubseteq f(y)$$

Are following functions monotone?

- $\mathrm{sign} : \mathbb{Z} \to \{-1, 0, 1\}$

Monotonic function

Let $(P, \leqslant), (R, \sqsubseteq)$ are posets

- a function $f : P \to R$ is monotone if for all $x, y \in P$ it holds

$$x \leqslant y \implies f(x) \sqsubseteq f(y)$$

Are following functions monotone?

- $\mathrm{sign} : \mathbb{Z} \to \{-1, 0, 1\}$
- $\mathrm{abs} : \mathbb{Z} \to \mathbb{N}$

Monotonic function

Let $(P, \leqslant), (R, \sqsubseteq)$ are posets

- a function $f : P \to R$ is monotone if for all $x, y \in P$ it holds

$$x \leqslant y \implies f(x) \sqsubseteq f(y)$$

Are following functions monotone?

- $\mathrm{sign} : \mathbb{Z} \to \{-1, 0, 1\}$
- $\mathrm{abs} : \mathbb{Z} \to \mathbb{N}$
- $\mathrm{middle} : \mathrm{Int}_L \to \mathbb{R}$

Monotonic function

Let $(P, \leqslant), (R, \sqsubseteq)$ are posets

- a function $f : P \to R$ is monotone if for all $x, y \in P$ it holds

$$x \leqslant y \implies f(x) \sqsubseteq f(y)$$

Are following functions monotone?

- $\mathrm{sign} : \mathbb{Z} \to \{-1, 0, 1\}$
- $\mathrm{abs} : \mathbb{Z} \to \mathbb{N}$
- $\mathrm{middle} : \mathrm{Int}_L \to \mathbb{R}$
- $\mathrm{size} : \mathrm{Int}_L \to \mathbb{N}$

Monotonic function

Let $(P, \leqslant), (R, \sqsubseteq)$ are posets

- a function $f : P \to R$ is monotone if for all $x, y \in P$ it holds

$$x \leqslant y \implies f(x) \sqsubseteq f(y)$$

Are following functions monotone?

- $\mathrm{sign} : \mathbb{Z} \to \{-1, 0, 1\}$
- $\mathrm{abs} : \mathbb{Z} \to \mathbb{N}$
- $\mathrm{middle} : \mathrm{Int}_L \to \mathbb{R}$
- $\mathrm{size} : \mathrm{Int}_L \to \mathbb{N}$

Monotonic function

Let $(P, \leqslant), (R, \sqsubseteq)$ are posets

- a function $f : P \to R$ is monotone if for all $x, y \in P$ it holds

$$x \leqslant y \implies f(x) \sqsubseteq f(y)$$

Are following functions monotone?

- $\mathrm{sign} : \mathbb{Z} \to \{-1, 0, 1\}$
- $\mathrm{abs} : \mathbb{Z} \to \mathbb{N}$
- $\mathrm{middle} : \mathrm{Int}_L \to \mathbb{R}$
- $\mathrm{size} : \mathrm{Int}_L \to \mathbb{N}$

A monotone function $f : P \to P$ on a poset $(P, \leqslant)$ is called a transformer.

Fixpoint

Let $(P, \leqslant)$ be a poset:

- $x \in P$ is called a fixpoint of transformer f if $f(x) = x$

Fixpoint

Let $(P, \leqslant)$ be a poset:

- $x \in P$ is called a fixpoint of transformer f if $f(x) = x$

What are fixpoints of following functions?

- $\lambda x.(x + 1)$ on $(\mathbb{Z}, \leqslant)$

Fixpoint

Let $(P, \leqslant)$ be a poset:

- $x \in P$ is called a fixpoint of transformer f if $f(x) = x$

What are fixpoints of following functions?

- $\lambda x.(x + 1)$ on $(\mathbb{Z}, \leqslant)$
- $\lambda[x, y].([x + 1, y + 1])$ on Int_L

Fixpoint

Let $(P, \leqslant)$ be a poset:

- $x \in P$ is called a fixpoint of transformer f if $f(x) = x$

What are fixpoints of following functions?

- $\lambda x.(x + 1)$ on $(\mathbb{Z}, \leqslant)$
- $\lambda[x, y].([x + 1, y + 1])$ on Int_L
- $\lambda x.(-x)$ on $\mathrm{Sig}_L = (\{-1, 0, 1, \top, \bot\}, \leqslant)$

Fixpoint theorems

Theorem (Knaster-Tarski)

A set of fixpoints of a transformer on a complete lattice forms a complete lattice.

Fixpoint theorems

Theorem (Knaster-Tarski)

A set of fixpoints of a transformer on a complete lattice forms a complete lattice.

As a corollary, there is

- the greatest fixed point $\mathtt{gfp}(\mathrm{f})$
- the least fixed point $\mathtt{lfp}(\mathrm{f})$

Fixpoint theorems

Theorem (Knaster-Tarski)

A set of fixpoints of a transformer on a complete lattice forms a complete lattice.

As a corollary, there is

- the greatest fixed point $\mathtt{gfp}(\mathrm{f})$
- the least fixed point $\mathtt{lfp}(\mathrm{f})$

What is $\mathtt{gfp}$ and $\mathtt{lfp}$ in $\mathrm{Sig}_{\mathrm{L}}$?

Fixpoint theorems

Theorem (Knaster-Tarski)

A set of fixpoints of a transformer on a complete lattice forms a complete lattice.

As a corollary, there is

- the greatest fixed point $\mathrm{gfp}(\mathrm{f})$
- the least fixed point $\mathrm{lfp}(\mathrm{f})$

What is gfp and lfp in $\mathrm{Sig}_{\mathrm{L}}$?

Theorem (Kleene)

Let $(\mathrm{L}, \leqslant)$ *be a complete lattice of finite height and transformer* f. *Then there exists* $\mathrm{n} \in \mathbb{N}$ *such that for all* $\mathrm{k} \in \mathbb{N}$ *it is* $\mathrm{f}^{\mathrm{n}}(\bot) = \mathrm{f}^{\mathrm{n}+\mathrm{k}}(\bot)$ *and* $\mathrm{f}^{\mathrm{n}}(\bot) = \mathrm{lfp}(\mathrm{f})$.

Fixpoint computation

Algorithm

```
1    x := ⊥;
2    do {
3       t := x;
4       x := f(x);
5    } while (x ≠ t);
```

Strongest postcondition

Given expression e and programs state s, then $\mathrm{sp}(s, e)$ is a strongest postcondition

- postcondition that implies any postcondition satisfied by the final state of any execution from s

Strongest postcondition

Given expression e and programs state s, then $sp(s, e)$ is a strongest postcondition

- postcondition that implies any postcondition satisfied by the final state of any execution from s

What is sp of of state S, where state is discribed as set of possible values of x:

- $sp(\{x \mid x \geqslant 5\}, x := x + 3)$

Strongest postcondition

Given expression e and programs state s, then $sp(s, e)$ is a strongest postcondition

- postcondition that implies any postcondition satisfied by the final state of any execution from s

What is sp of of state S, where state is discribed as set of possible values of x:

- $sp(\{x \mid x \geqslant 5\}, x := x + 3)$
- $sp(S, x := x + 3)$

Strongest postcondition

Given expression e and programs state s, then $sp(s, e)$ is a strongest postcondition

- postcondition that implies any postcondition satisfied by the final state of any execution from s

What is sp of of state S, where state is discribed as set of possible values of x:

- $sp(\{x \mid x \geqslant 5\}, x := x + 3)$
- $sp(S, x := x + 3)$
- $sp(S, x := 0)$

Strongest postcondition

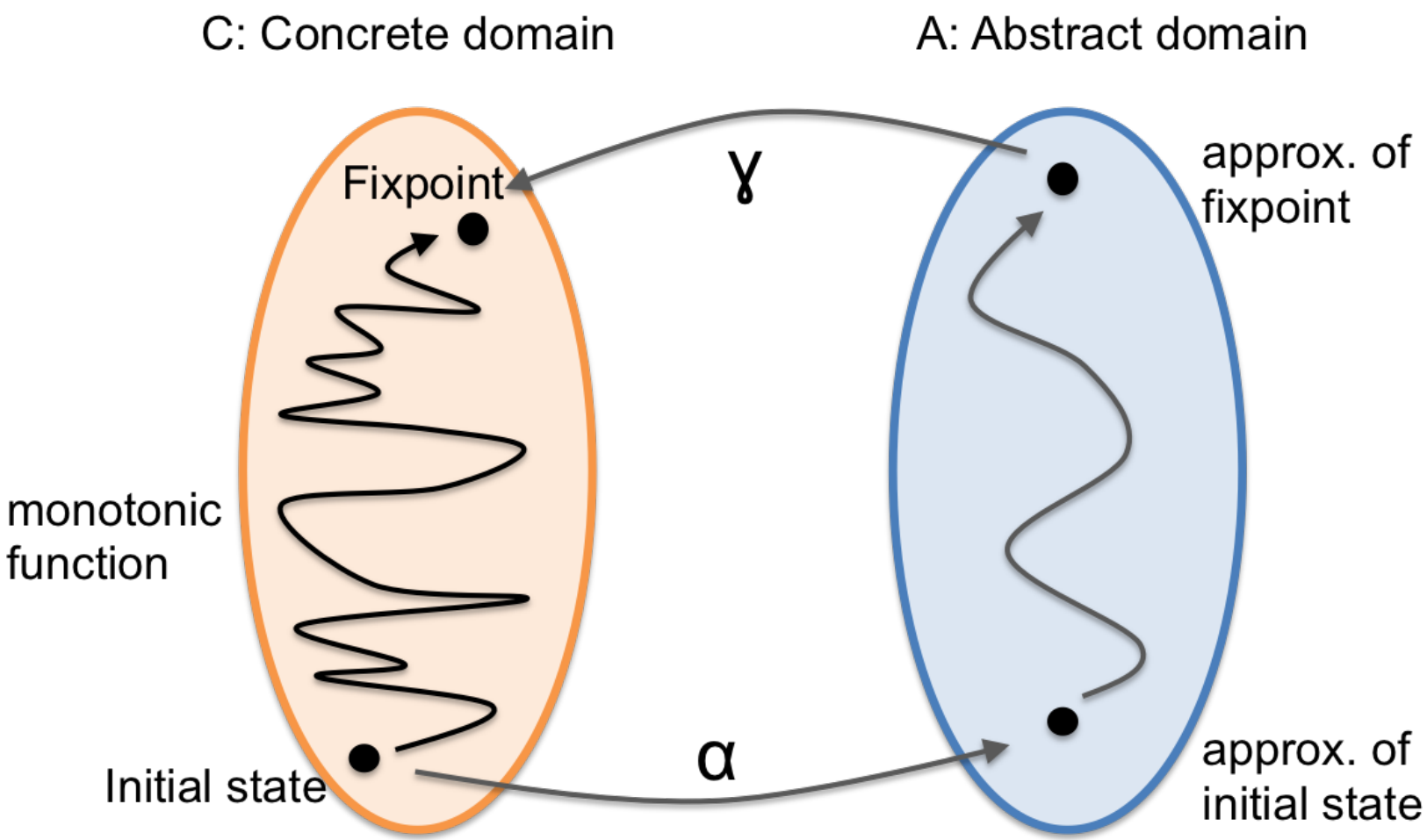
Given expression e and programs state s, then $sp(s, e)$ is a strongest postcondition

- postcondition that implies any postcondition satisfied by the final state of any execution from s

What is sp of of state S, where state is discribed as set of possible values of x:

- $sp(\{x \mid x \geqslant 5\}, x := x + 3)$
- $sp(S, x := x + 3)$
- $sp(S, x := 0)$
- $sp(S, assume(x < 10))$

Concrete and abstract domains



Concrete and abstract domains

Except meet and join operations we can equip lattice by other transformers.

Concrete domain – $(C, \leqslant, \wedge, \vee, \{f_1, \ldots, f_k\})$
where $f_1, \ldots, f_k$ are concrete transformers

Abstract domain – $(A, \sqsubseteq, \sqcap, \sqcup, \{af_1, \ldots, af_k\})$
where $af_1, \ldots, af_k$ are abstract transformers

Idea of abstract interpretation

Consider the assignment: `c = a + b`

Idea of abstract interpretation

Consider the assignment: `c = a + b`

Interpreter (concrete domain):

$$\begin{bmatrix} \texttt{a:10} \\ \texttt{b:-1} \\ \texttt{c:3} \end{bmatrix} \xrightarrow{\texttt{c = a + b}} \begin{bmatrix} \texttt{a:10} \\ \texttt{b:-1} \\ \texttt{c:9} \end{bmatrix}$$

Idea of abstract interpretation

Consider the assignment: `c = a + b`

Interpreter (concrete domain):

$$\begin{bmatrix} \texttt{a}:10 \\ \texttt{b}:-1 \\ \texttt{c}:3 \end{bmatrix} \xrightarrow{\texttt{c = a + b}} \begin{bmatrix} \texttt{a}:10 \\ \texttt{b}:-1 \\ \texttt{c}:9 \end{bmatrix}$$

Abstract interpreter (interval domain):

$$\begin{bmatrix} \texttt{a} \in [0,10] \\ \texttt{b} \in [-5,5] \\ \texttt{c} \in [0,10] \end{bmatrix} \xrightarrow{\texttt{c = a + b}} \begin{bmatrix} \texttt{a} \in [0,10] \\ \texttt{b} \in [-5,5] \\ \texttt{c} \in [-5,15] \end{bmatrix}$$

Idea of abstract interpretation

Consider the assignment: `c = a + b`

Interpreter (concrete domain):

$$\begin{bmatrix} \texttt{a}:10 \\ \texttt{b}:-1 \\ \texttt{c}:3 \end{bmatrix} \xrightarrow{\texttt{c = a + b}} \begin{bmatrix} \texttt{a}:10 \\ \texttt{b}:-1 \\ \texttt{c}:9 \end{bmatrix}$$

Abstract interpreter (interval domain):

$$\begin{bmatrix} \texttt{a} \in [0,10] \\ \texttt{b} \in [-5,5] \\ \texttt{c} \in [0,10] \end{bmatrix} \xrightarrow{\texttt{c = a + b}} \begin{bmatrix} \texttt{a} \in [0,10] \\ \texttt{b} \in [-5,5] \\ \texttt{c} \in [-5,15] \end{bmatrix}$$

Each abstract state represents set of concrete states.

Abstract Interpretation Recipe: Setup

Given control-flow graph (V, E, r), where

- $V = \{v_1, \ldots, v_n\}$ is set of program locations
- $E \subseteq V \times V$ are program transitions
- $r : E \to \mathrm{Expr}$, so each $r(u, v)$ is labeled by expresion doing transformation of state u to v

Abstract Interpretation Recipe: Setup

Given control-flow graph (V, E, r), where

- $V = \{v_1, \ldots, v_n\}$ is set of program locations
- $E \subseteq V \times V$ are program transitions
- $r : E \to \mathrm{Expr}$, so each $r(u, v)$ is labeled by expresion doing transformation of state u to v

1. Design abstract domain A that represents sets of program states. Example: Sig_L domain.

Abstract Interpretation Recipe: Setup

Given control-flow graph (V, E, r), where

- $V = \{v_1, \ldots, v_n\}$ is set of program locations
- $E \subseteq V \times V$ are program transitions
- $r : E \to \mathrm{Expr}$, so each $r(u, v)$ is labeled by expresion doing transformation of state u to v

1. Design abstract domain A that represents sets of program states. Example: Sig_L domain.
2. Define $\gamma : A \to C$ giving meaning to elements of A

Abstract Interpretation Recipe: Setup

Given control-flow graph (V, E, r), where

- $V = \{v_1, \ldots, v_n\}$ is set of program locations
- $E \subseteq V \times V$ are program transitions
- $r : E \to \mathrm{Expr}$, so each $r(u, v)$ is labeled by expresion doing transformation of state u to v

1. Design abstract domain A that represents sets of program states. Example: Sig_L domain.
2. Define $\gamma : A \to C$ giving meaning to elements of A
3. define lattice ordering $\sqsubseteq$ on A such that $a_1 \sqsubseteq a_2 \to \gamma(a_1) \subseteq \gamma(a_2)$

Abstract Interpretation Recipe: Setup

Given control-flow graph (V, E, r), where

- $V = \{v_1, \ldots, v_n\}$ is set of program locations
- $E \subseteq V \times V$ are program transitions
- $r : E \to \mathrm{Expr}$, so each $r(u, v)$ is labeled by expresion doing transformation of state u to v

1. Design abstract domain A that represents sets of program states. Example: Sig_L domain.
2. Define $\gamma : A \to C$ giving meaning to elements of A
3. define lattice ordering $\sqsubseteq$ on A such that $a_1 \sqsubseteq a_2 \to \gamma(a_1) \subseteq \gamma(a_2)$
4. Define $sp_A : A \times \mathrm{Expr} \to A$ that maps an abstract element and a CFG statement to new abstract element, such that $sp(\gamma(a), e) \subseteq \gamma(sp_A(a, e))$

Abstract Interpretation Recipe: Setup

Given control-flow graph (V, E, r), where

- $V = \{v_1, \ldots, v_n\}$ is set of program locations
- $E \subseteq V \times V$ are program transitions
- $r : E \to \mathrm{Expr}$, so each $r(u, v)$ is labeled by expresion doing transformation of state u to v

1. Design abstract domain A that represents sets of program states. Example: Sig_L domain.
2. Define $\gamma : A \to C$ giving meaning to elements of A
3. define lattice ordering $\sqsubseteq$ on A such that $a_1 \sqsubseteq a_2 \to \gamma(a_1) \subseteq \gamma(a_2)$
4. Define $sp_A : A \times \mathrm{Expr} \to A$ that maps an abstract element and a CFG statement to new abstract element, such that $sp(\gamma(a), e) \subseteq \gamma(sp_A(a, e))$

Abstract Interpretation Recipe: Setup

Given control-flow graph (V, E, r), where

- $V = \{v_1, \ldots, v_n\}$ is set of program locations
- $E \subseteq V \times V$ are program transitions
- $r : E \to \mathrm{Expr}$, so each $r(u, v)$ is labeled by expresion doing transformation of state u to v

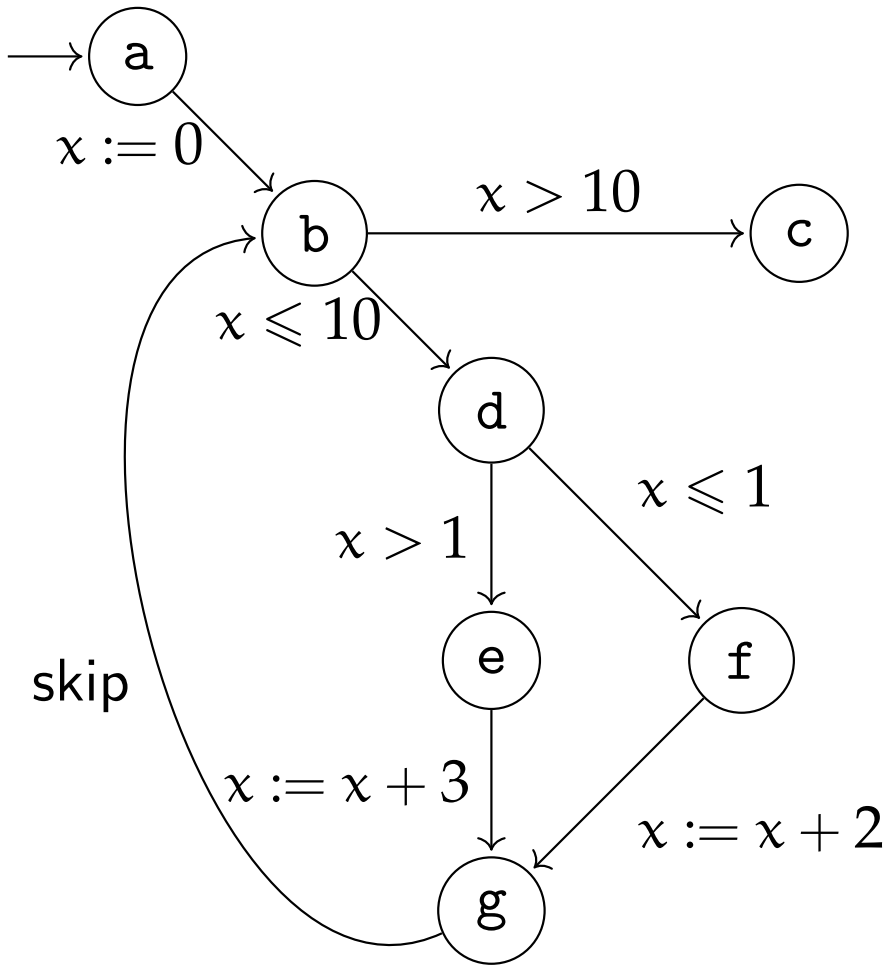
1. Design abstract domain A that represents sets of program states. Example: Sig_L domain.
2. Define $\gamma : A \to C$ giving meaning to elements of A
3. define lattice ordering $\sqsubseteq$ on A such that $a_1 \sqsubseteq a_2 \to \gamma(a_1) \subseteq \gamma(a_2)$
4. Define $sp_A : A \times \mathrm{Expr} \to A$ that maps an abstract element and a CFG statement to new abstract element, such that $sp(\gamma(a), e) \subseteq \gamma(sp_A(a, e))$

Example: $a = 0$, $e =$ `x := x + 1`

Running Abstract Interpretation

Compute lfp for each program state, where sp_A computes one iteration of interpretation in abstract domain.

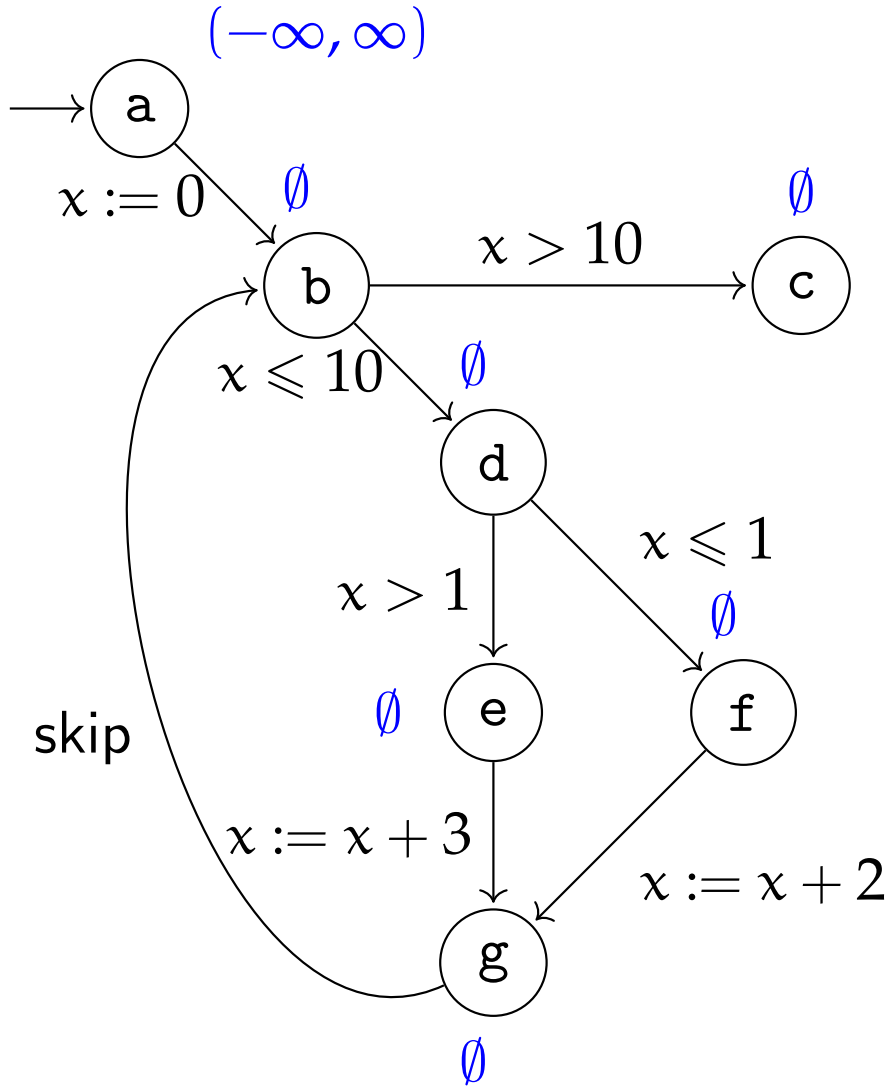
```
1        // a
2        x := 0;
3        // b
4        while (x ⩽ 10) {
5          // d
6          if (x > 1)
7               // e
8               x := x + 3
9          else
10              // f
11              x := x + 2
12         // g
13       }
14       // c
```



Running Abstract Interpretation

Compute lfp for each program state, where sp_A computes one iteration of interpretation in abstract domain.

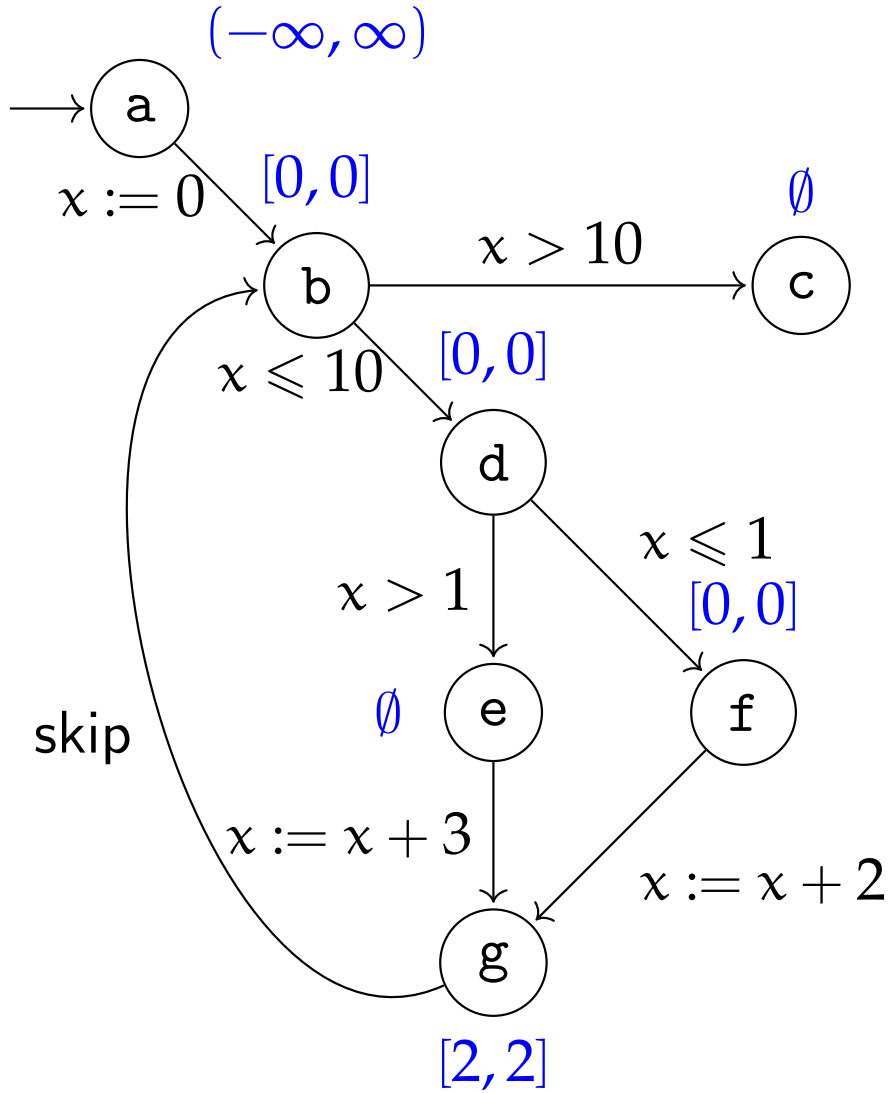
```
1        // a
2        x := 0;
3        // b
4        while (x ⩽ 10) {
5          // d
6          if (x > 1)
7              // e
8              x := x + 3
9          else
10             // f
11             x := x + 2
12       // g
13       }
14       // c
```



Running Abstract Interpretation

Compute lfp for each program state, where sp_A computes one iteration of interpretation in abstract domain.

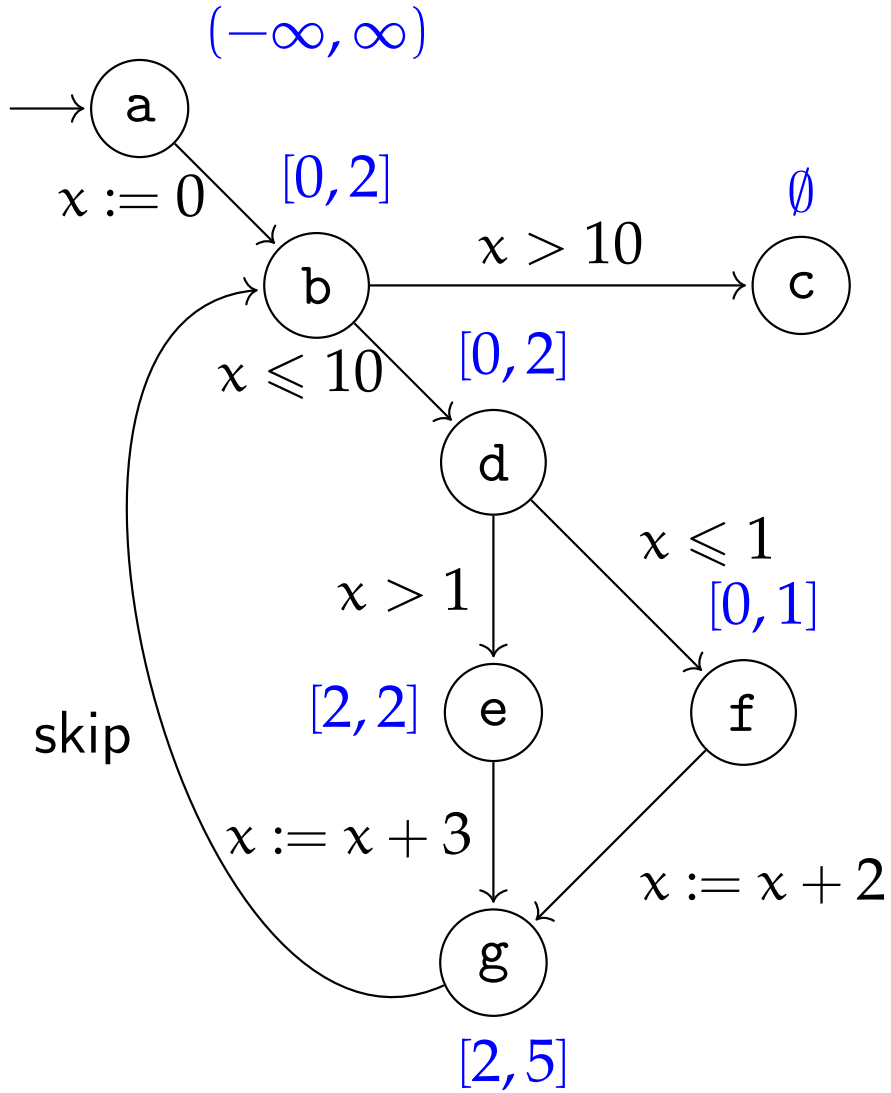
```
1        // a
2        x := 0;
3        // b
4        while (x ⩽ 10) {
5          // d
6          if (x > 1)
7               // e
8               x := x + 3
9          else
10              // f
11              x := x + 2
12         // g
13       }
14       // c
```



Running Abstract Interpretation

Compute lfp for each program state, where sp_A computes one iteration of interpretation in abstract domain.

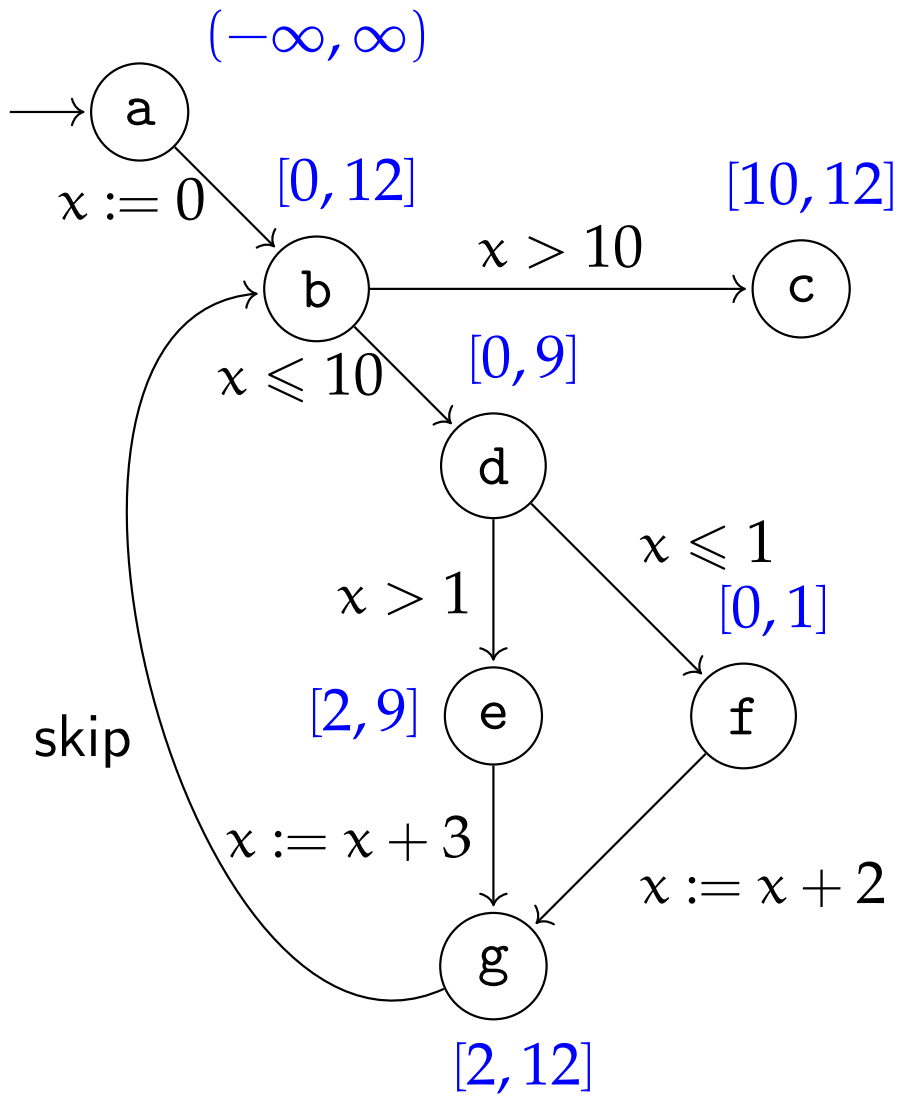
```
1        // a
2        x := 0;
3        // b
4        while (x ⩽ 10) {
5          // d
6          if (x > 1)
7               // e
8               x := x + 3
9          else
10              // f
11              x := x + 2
12         // g
13       }
14       // c
```



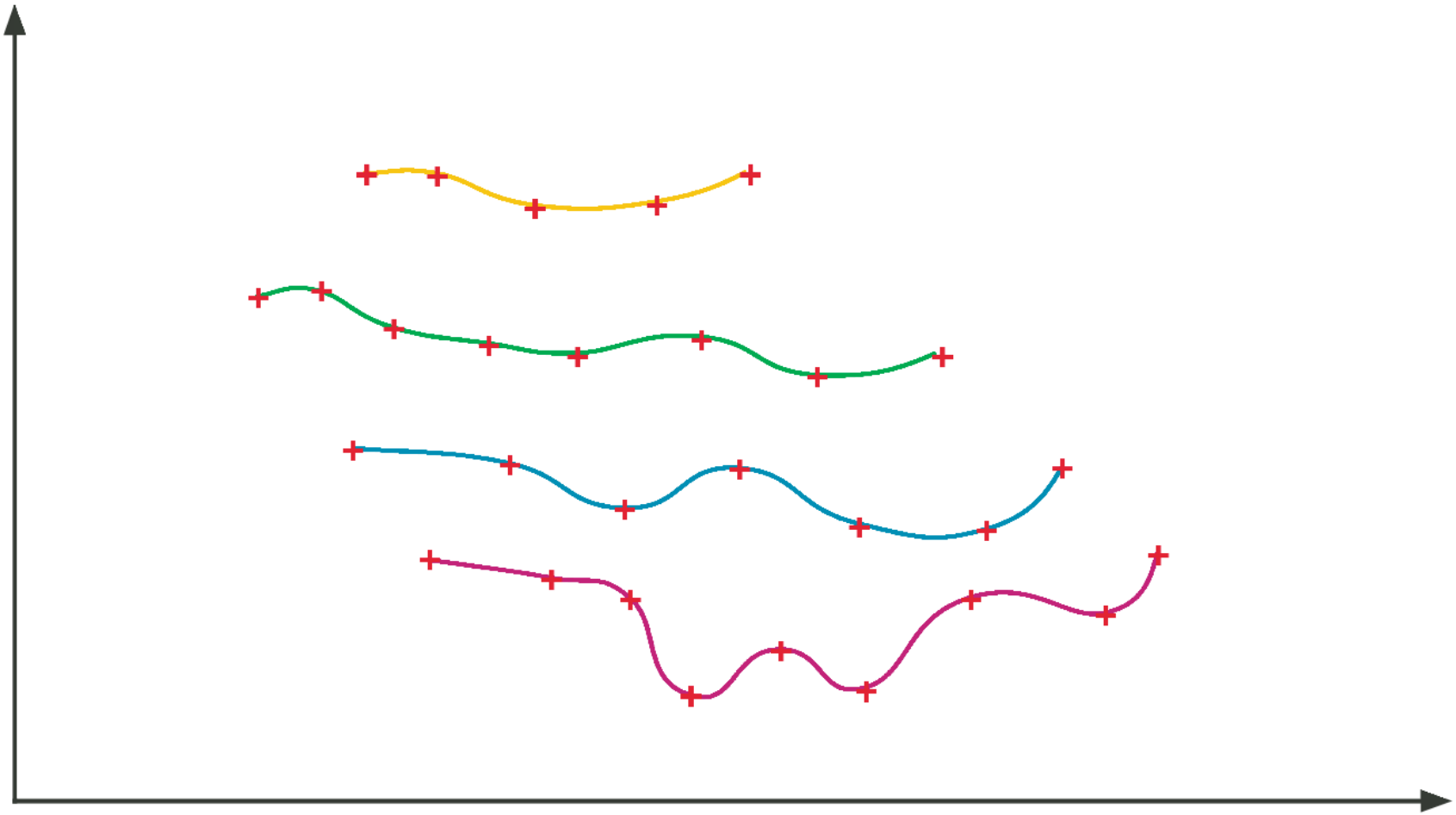
Running Abstract Interpretation

The resulting fixpoint describes an inductive program invariant.

```
1        // a
2        x := 0;
3        // b
4        while (x ⩽ 10) {
5          // d
6          if (x > 1)
7              // e
8              x := x + 3
9          else
10             // f
11             x := x + 2
12       // g
13       }
14       // c
```

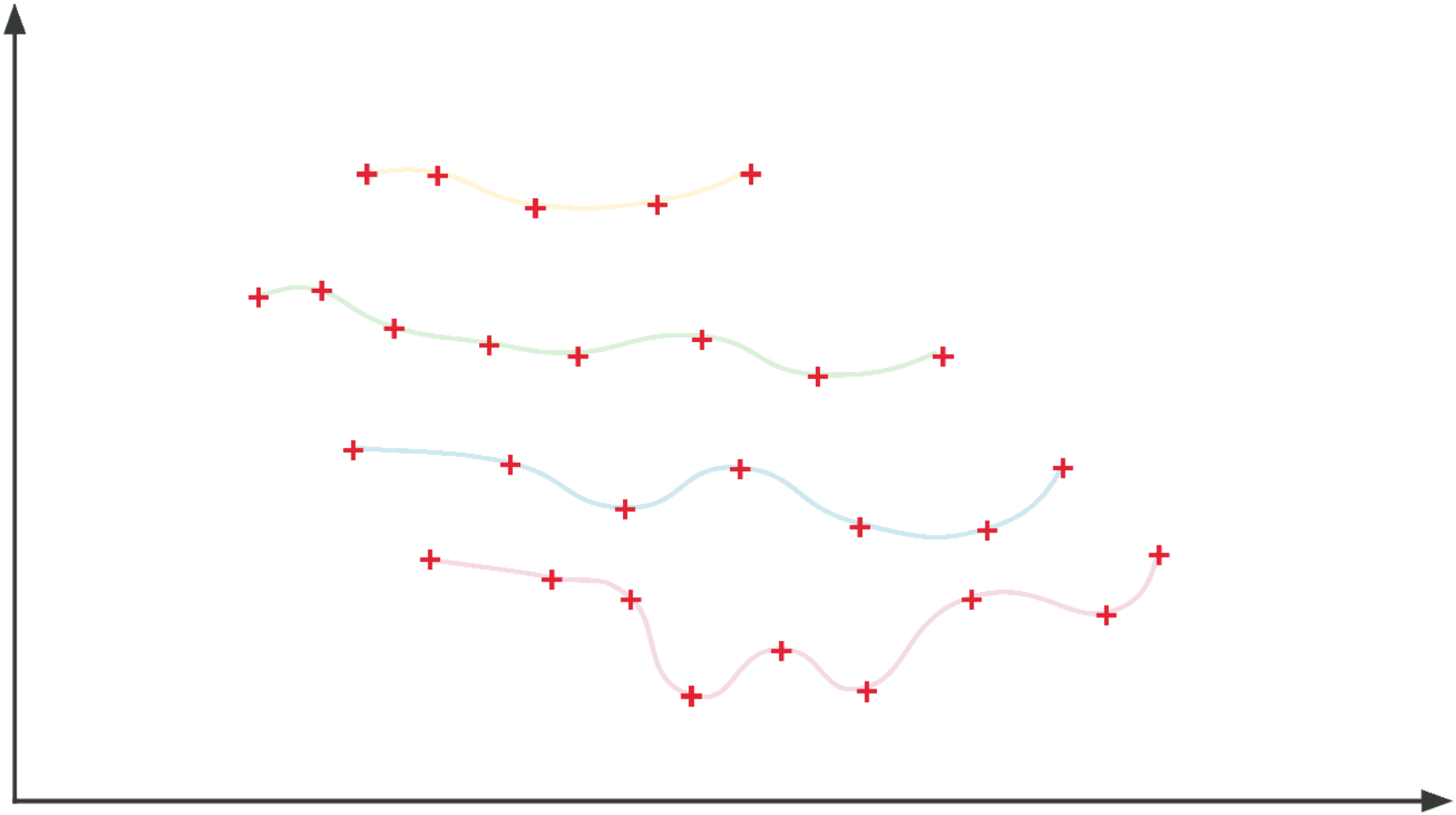


Abstractions



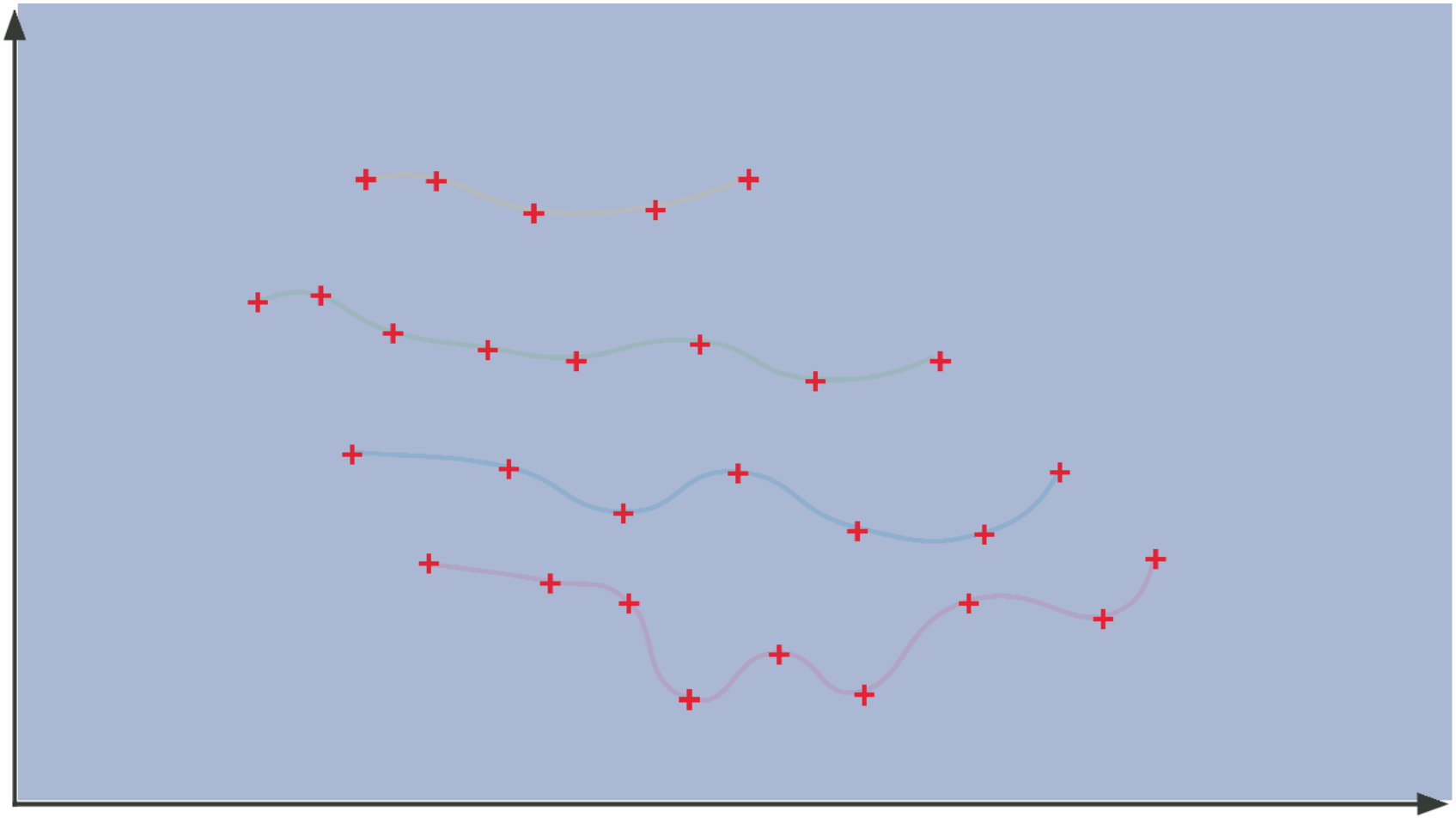
- We want to abstract this set of traces with two variables x and y

Abstractions



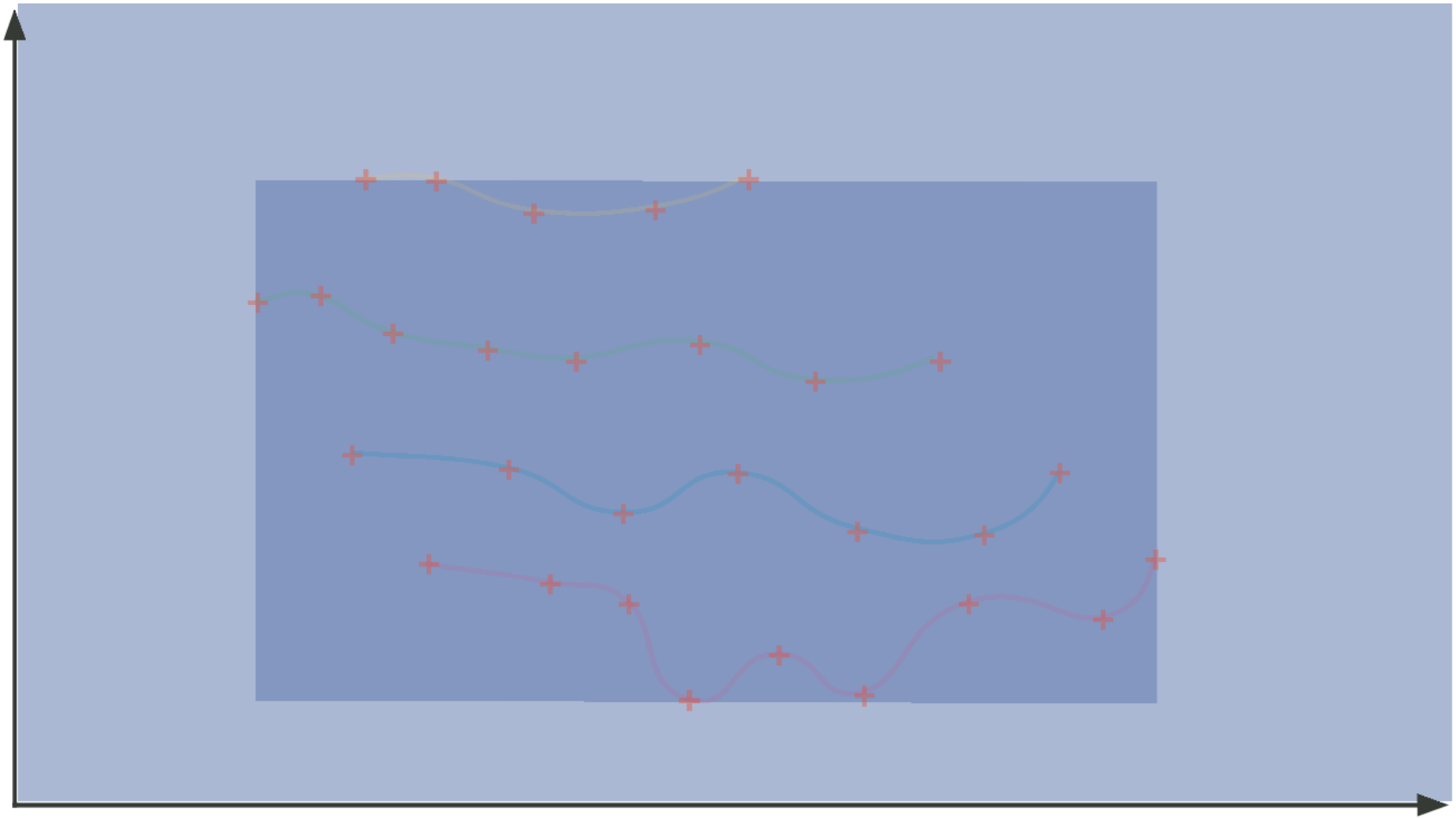
- Decomposition into set of local invariants on memory states attached to each program point

Abstractions: Sign analysis



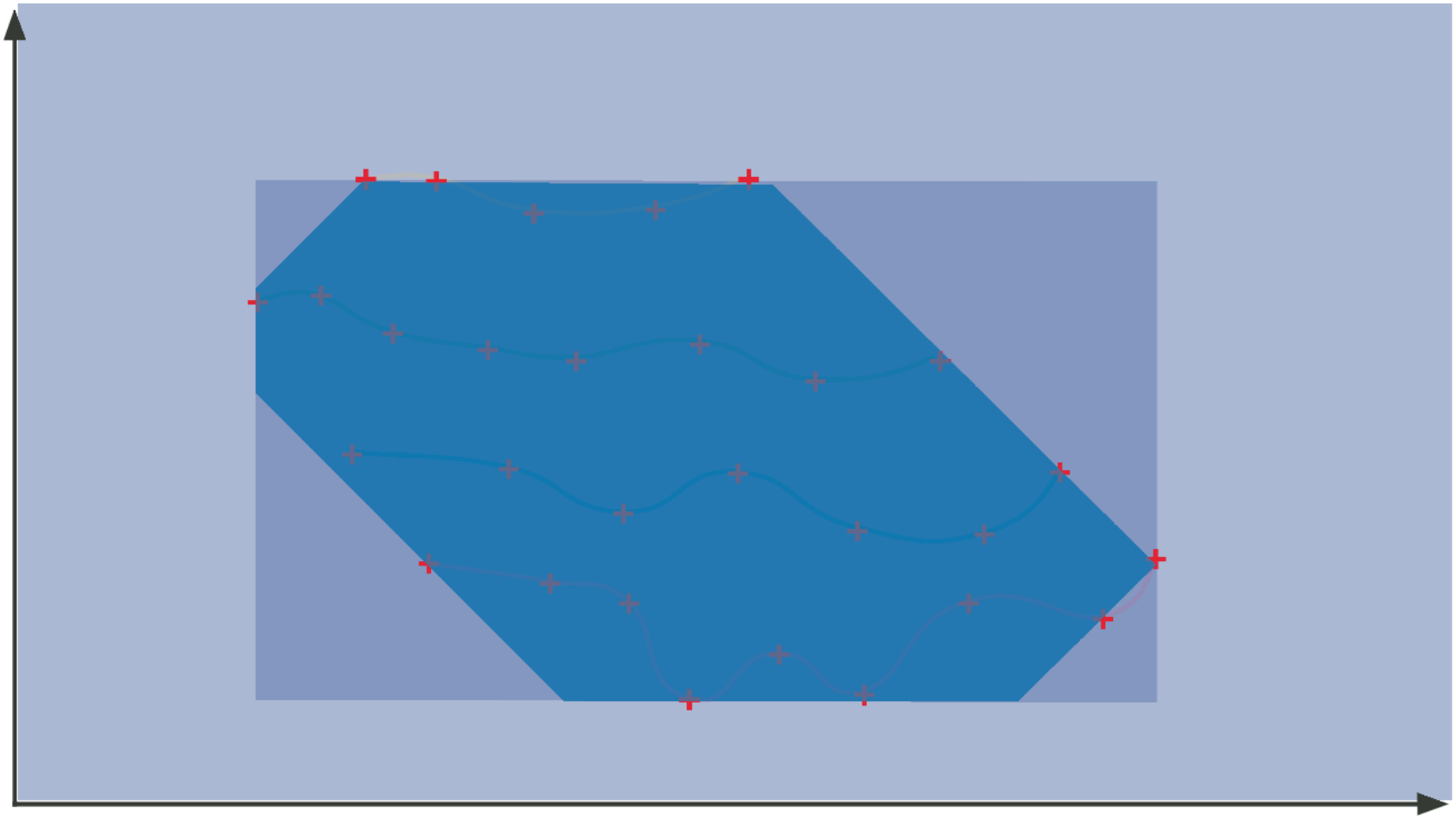
- Local invariants: $x \geqslant 0, y \geqslant 0$

Abstractions: Interval



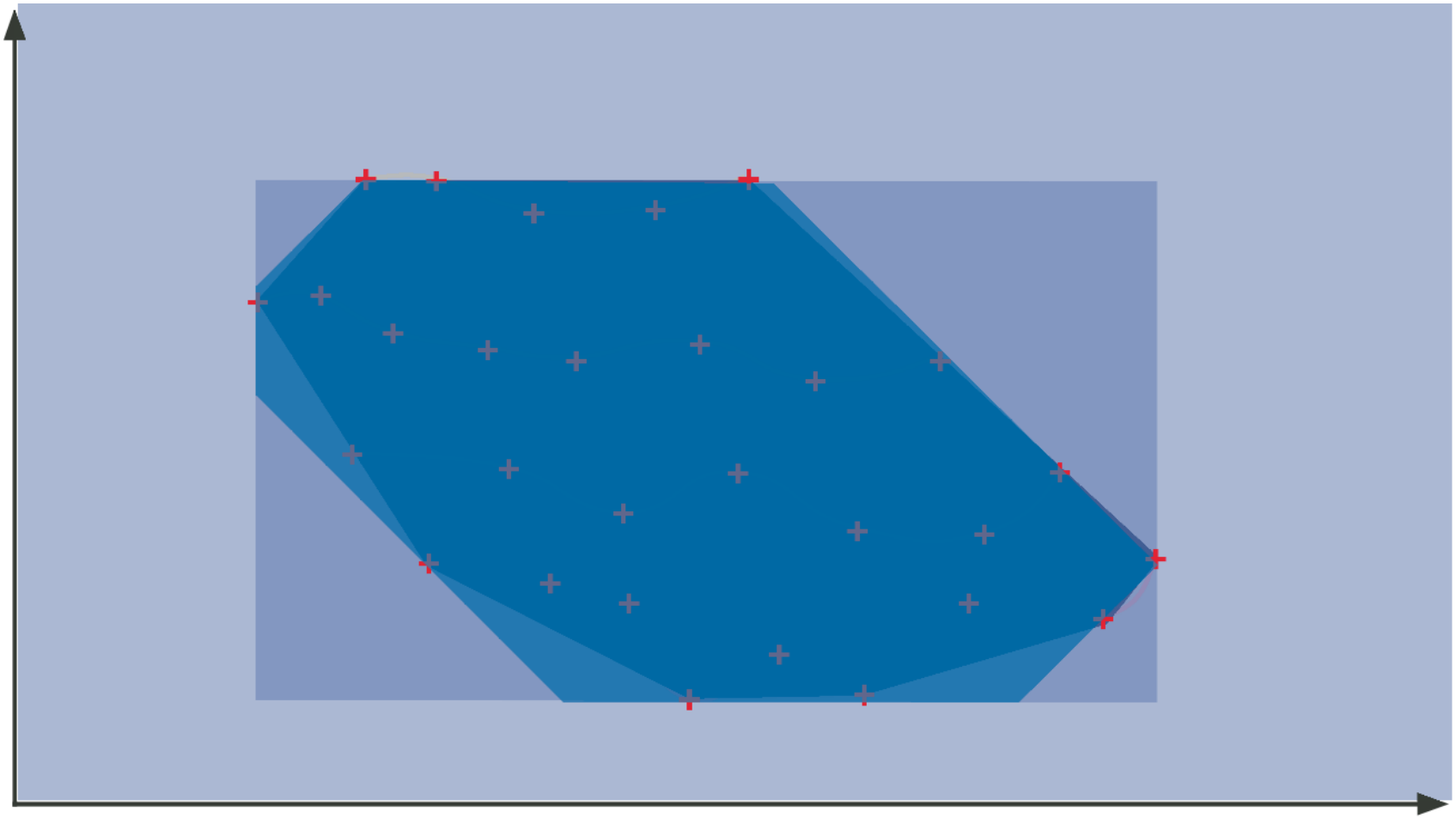
- Local invariants: $a \leqslant x \leqslant b, c \leqslant y \leqslant d$, where a, b, c, d are discovered by analysis
- Cannot prove invariant $0 \leqslant x \leqslant y$

Abstractions: Octagons



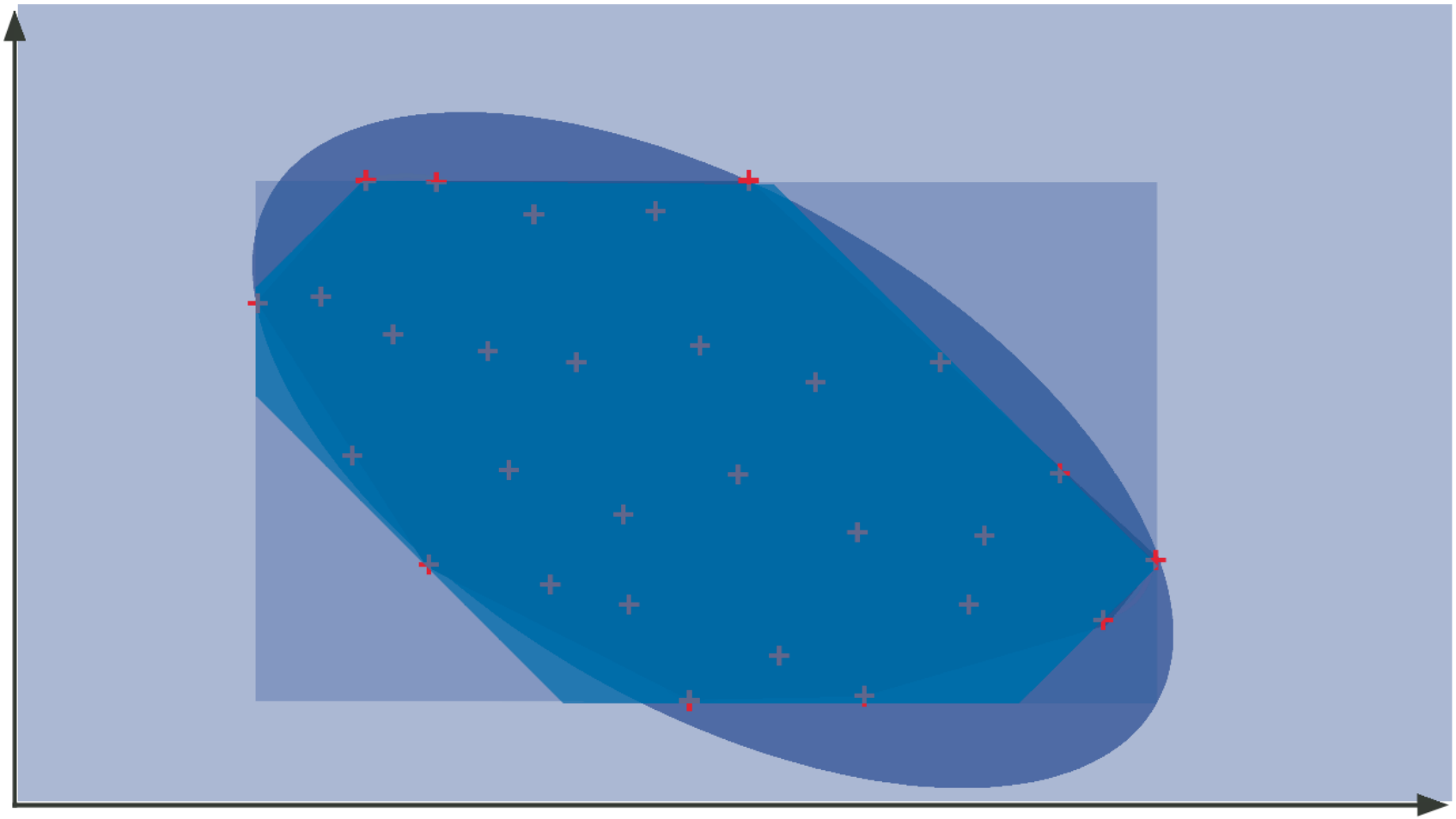
- Local invariants: $x \leqslant a, x - y \leqslant b, x + y \leqslant c$

Abstractions: Convex Polyhedra



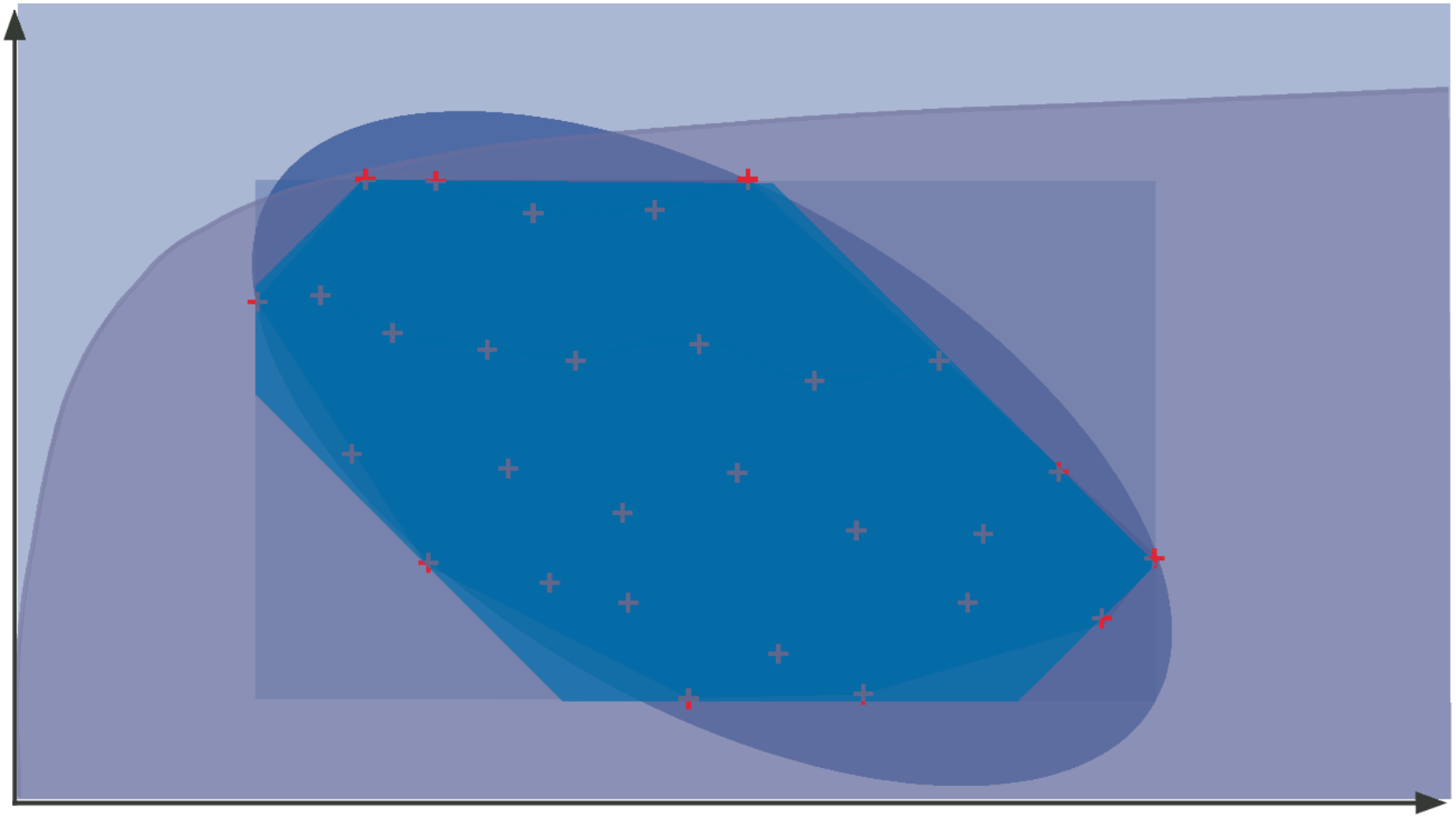
- Local invariants: $a \cdot x + b \cdot y \leqslant c$

Abstractions: Ellipsoids



- Local invariants: $(x - a)^2 + (y - b)^2 \leqslant c$

Abstractions: Exponential



- Local invariants: $a^x \leqslant y$