Solvers Principles and Architecture (SPA)

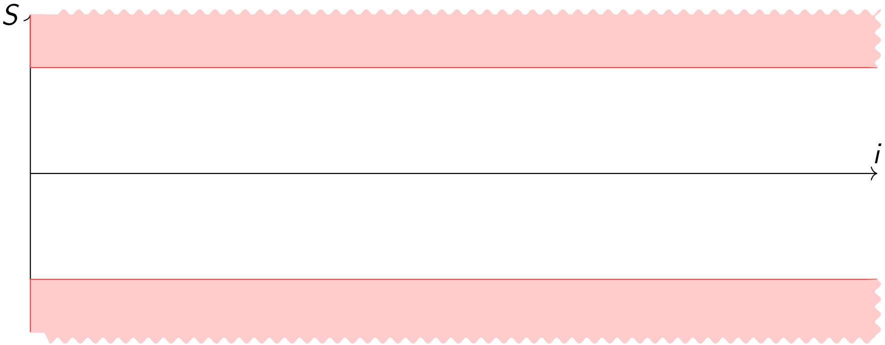
Part 2

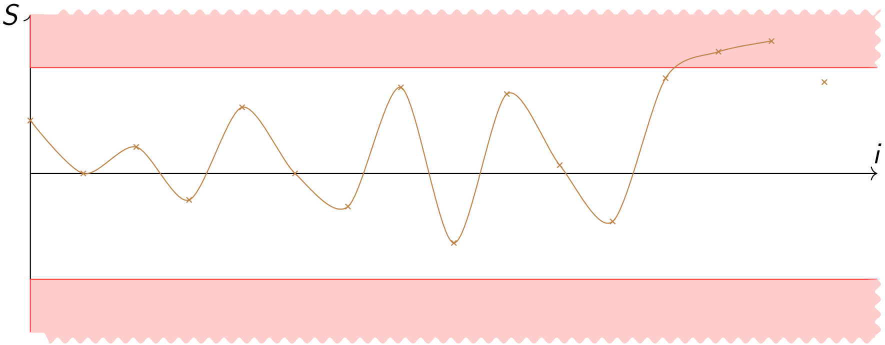
Abstract Interpretation (Basics)

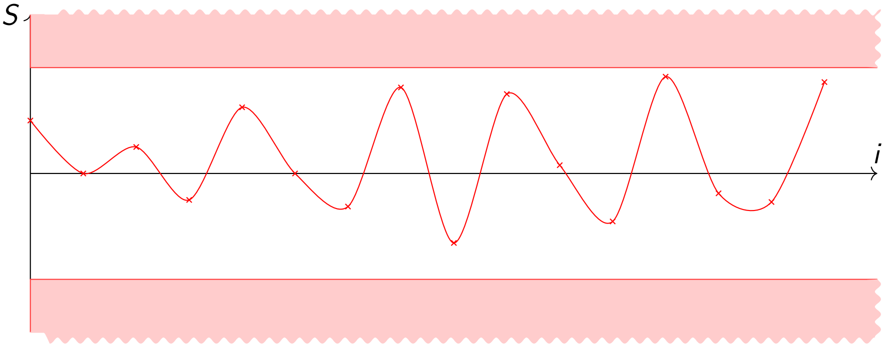
Master Sciences Informatique (Sif)
September, 2019
Rennes

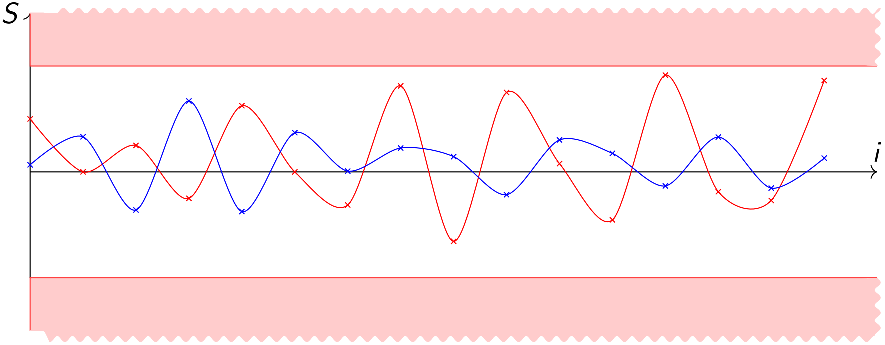
Khalil Ghorbal
khalil.ghorbal@inria.fr

Abstract Interpretation : Intuitions

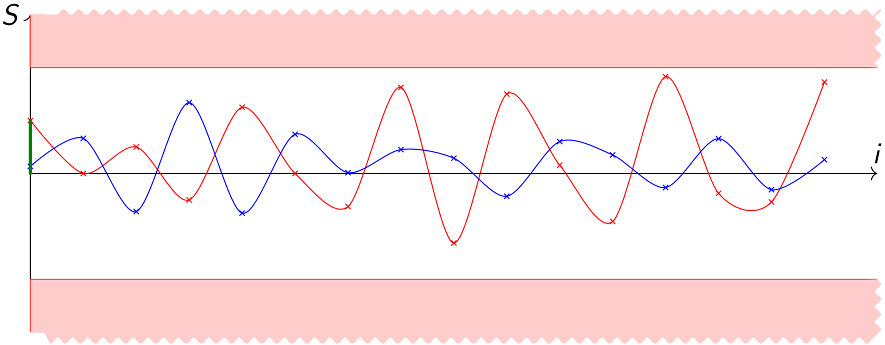


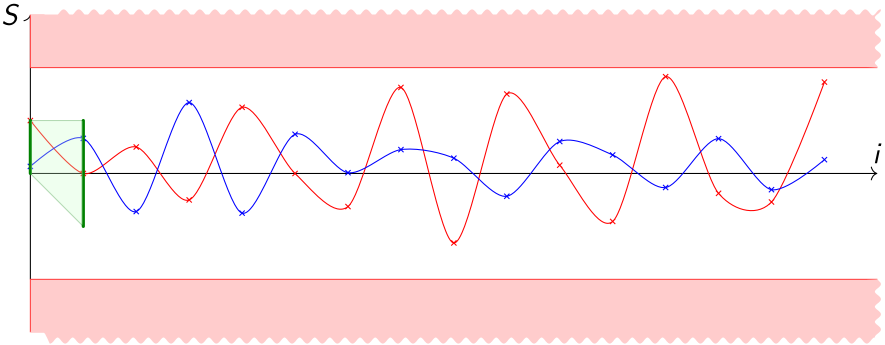

S
i

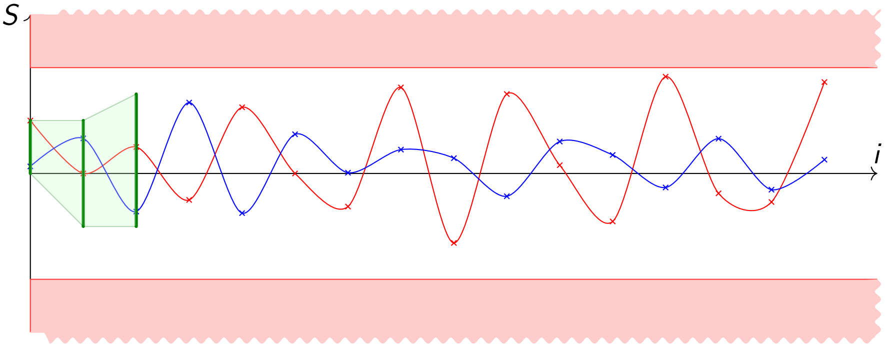

S
i

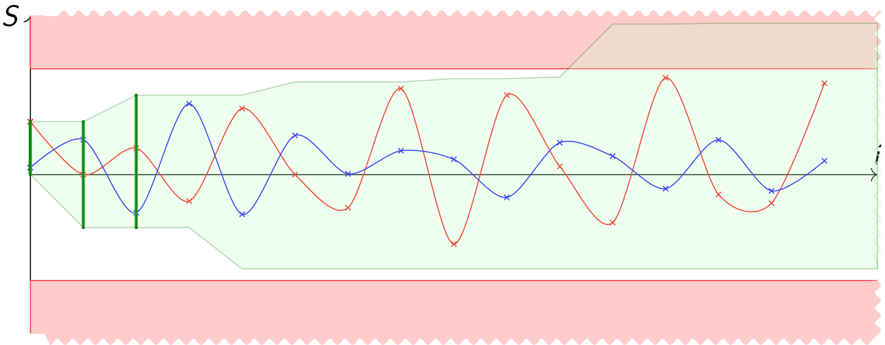


➻ What about the missed bugs ? are they severe ?

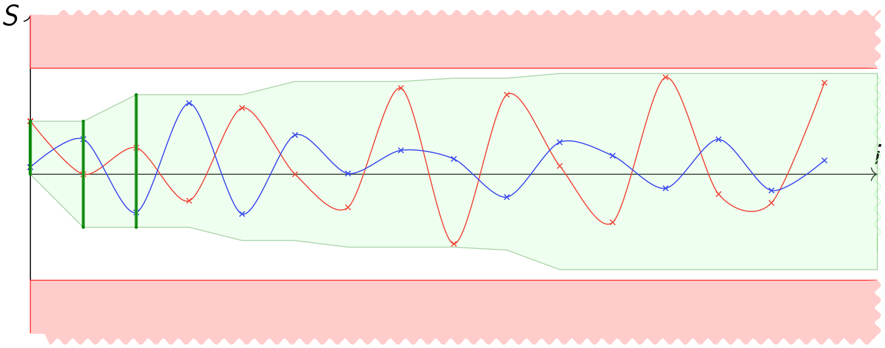

S
i


S
i


S
i



- Over-approximation may lead to **false alarms**. 



- Accurate over-approximation gives a safety **proof**.

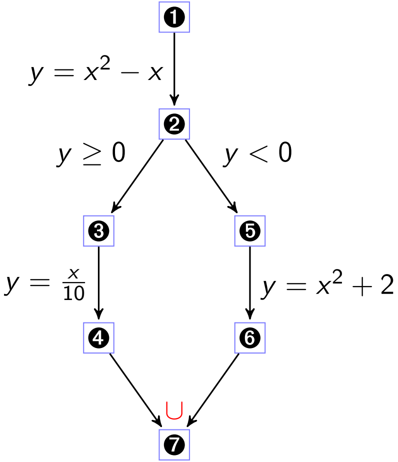
Examples

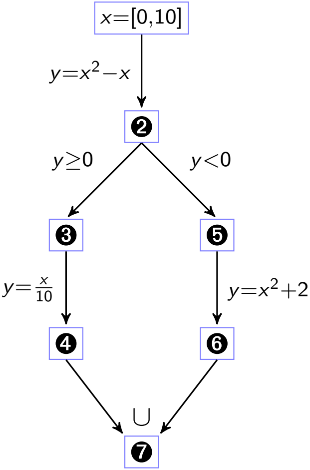
- 1982, The Vancouver stock exchange: after 22 months the index had fallen to $524,811$ instead of $1098,811$
- 1985, Therac 25 (radiation therapy machine) : 5 patients killed (overdoses of radiation)
- 1991, The Patriot Missile: 28 soldiers killed
- 1996, Ariane 5: more than 1 billion $ gone in 40 seconds

E. Dijkstra (1972)

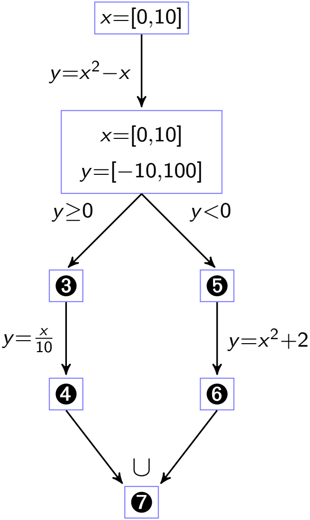
Program testing can be used to show the presence of bugs, but never to show their absence!

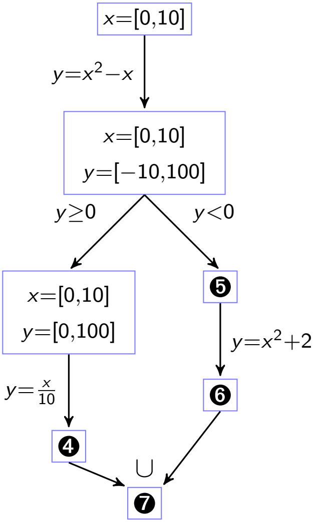
```
begin
x = [0,10]; ❶
y = x*x - x ❷
if (y >= 0)  ❸ then
y = x / 10; ❹
else  ❺
  y = x*x + 2; ❻
done; ❼
end
```

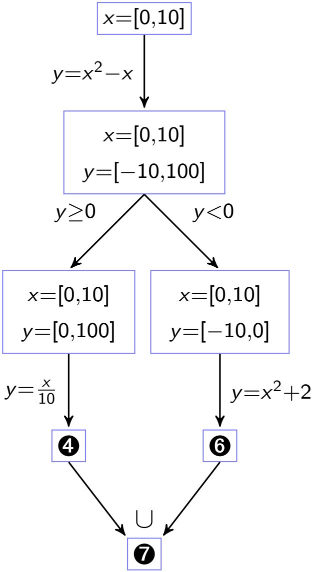


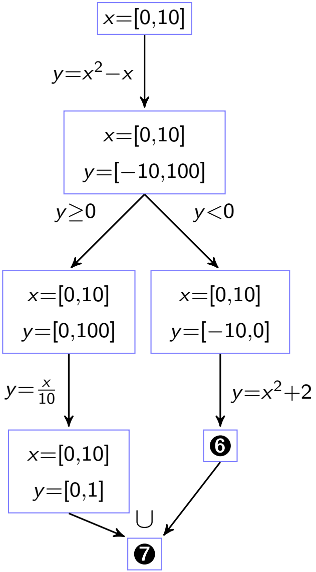

x=[0,10]
y=x²−x
❷
y≥0
y<0
❸
❺
y=x/10
y=x²+2
❹
❻
∪
❼

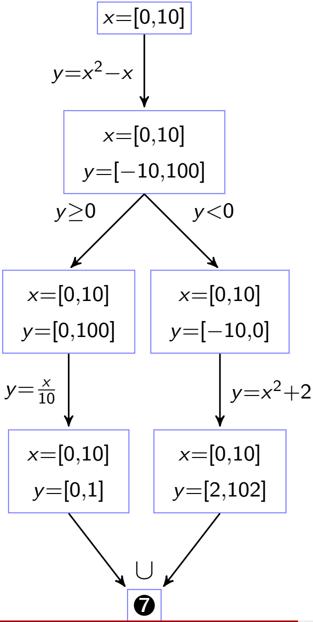
Forward Propagation

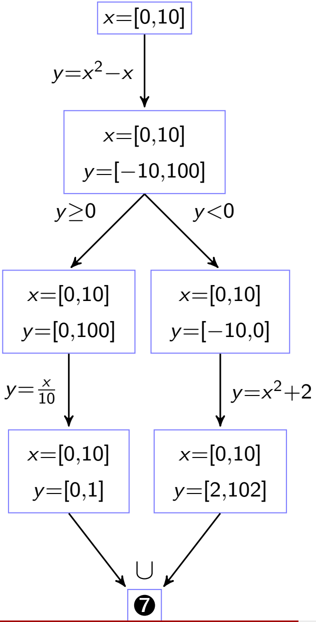


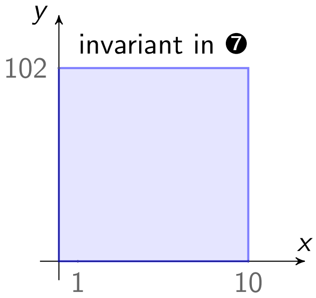

x=[0,10]
y=x^2−x
x=[0,10]
y=[−10,100]
y≥0
y<0
x=[0,10]
y=[0,100]
5
y=x^2+2
y=$\frac{x}{10}$
6
4
∪
7


x=[0,10]
y=x²−x
x=[0,10]
y=[−10,100]
y≥0
y<0
x=[0,10]
y=[0,100]
x=[0,10]
y=[−10,0]
y=x/10
y=x²+2
❹
❻
∪
❼

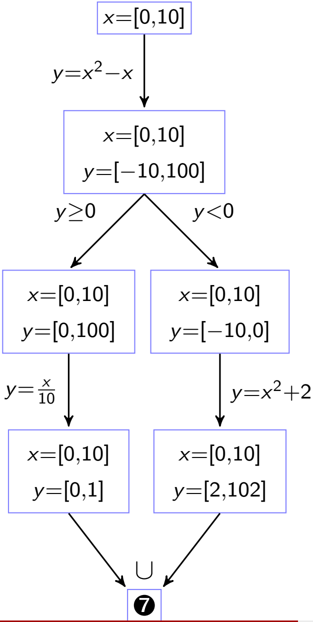

x=[0,10]
y=x^2−x
x=[0,10]
y=[−10,100]
y≥0
y<0
x=[0,10]
y=[0,100]
x=[0,10]
y=[−10,0]
y=x/10
y=x^2+2
x=[0,10]
y=[0,1]
❻
∪
❼

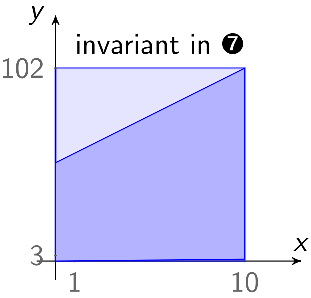

x=[0,10]
y=x²−x
x=[0,10]
y=[−10,100]
y≥0
y<0
x=[0,10]
y=[0,100]
x=[0,10]
y=[−10,0]
y=x/10
y=x²+2
x=[0,10]
y=[0,1]
x=[0,10]
y=[2,102]
∪
7


x=[0,10]
y=x²−x
x=[0,10]
y=[−10,100]
y≥0
y<0
x=[0,10]
y=[0,100]
x=[0,10]
y=[−10,0]
y=x/10
y=x²+2
x=[0,10]
y=[0,1]
x=[0,10]
y=[2,102]
∪
❼

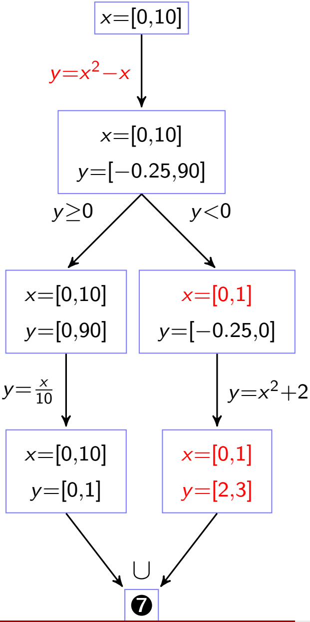

y
invariant in ❼
102
x
1
10

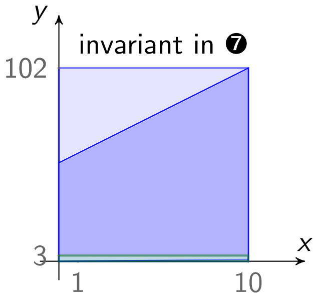
Forward Propagation

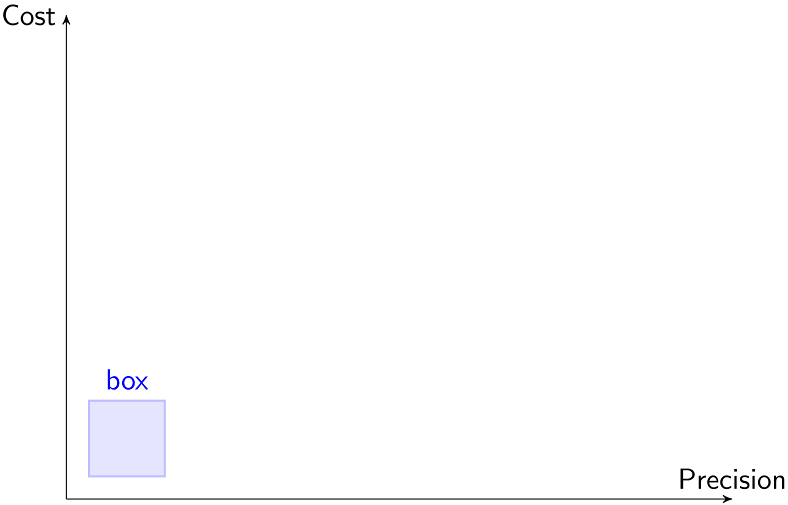


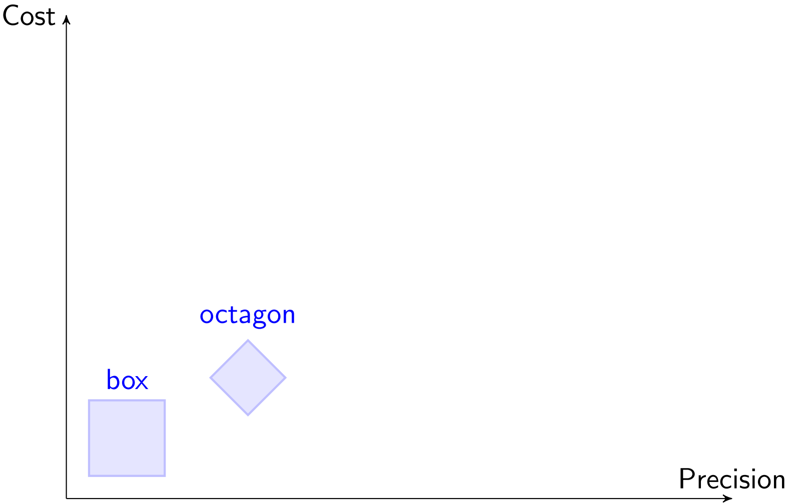


Forward Propagation

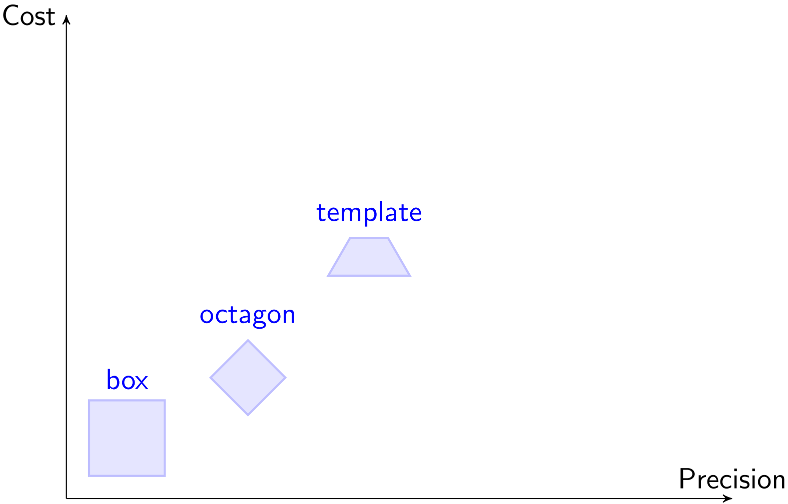


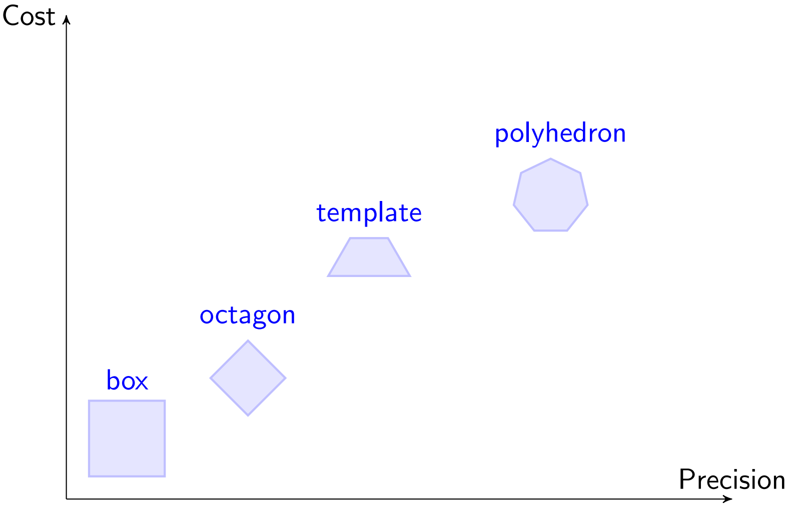



Cost
box
Precision

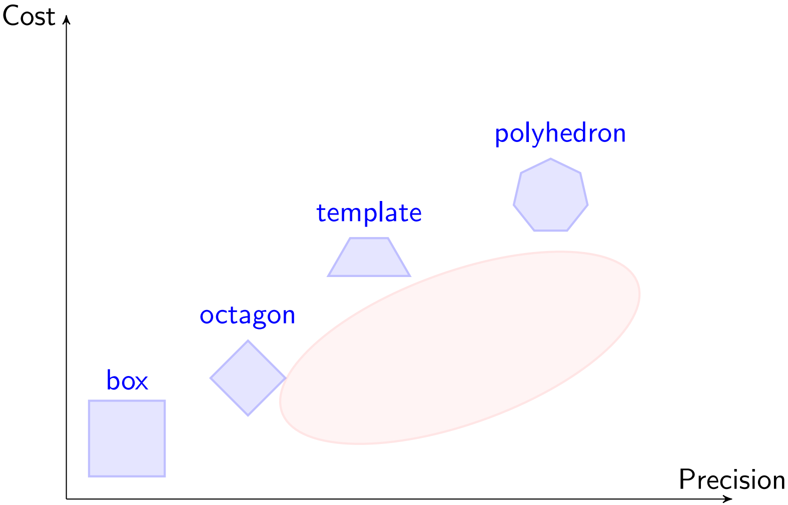

Cost
octagon
box
Precision

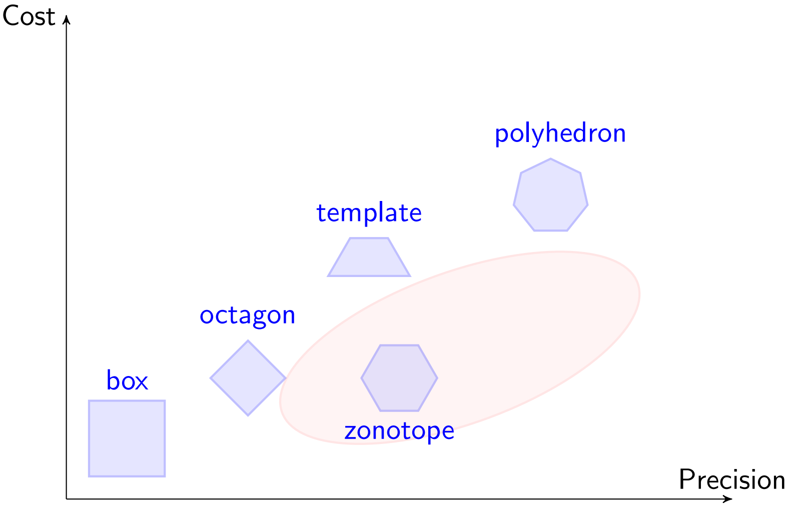
Precision Cost Trade-off

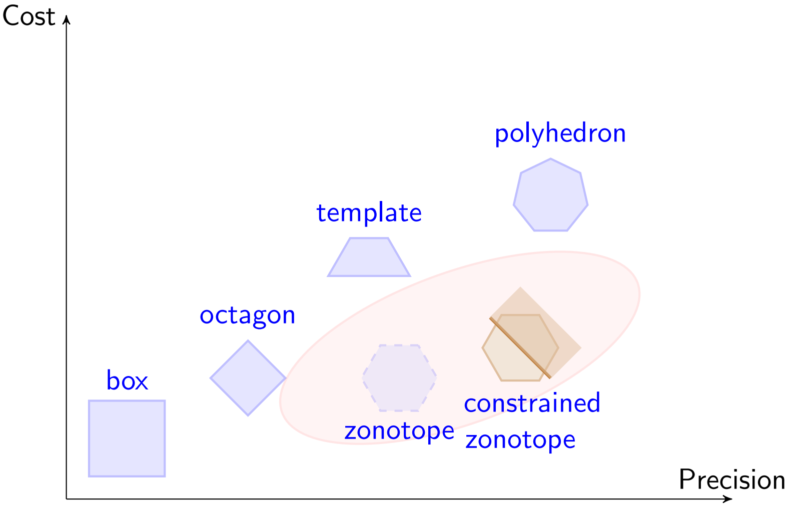



Cost
polyhedron
template
octagon
box
Precision

Precision Cost Trade-off




Cost
polyhedron
template
octagon
box
zonotope
Precision


Cost
polyhedron
template
octagon
box
zonotope
constrained
zonotope
Precision

1. Static Analysis-based Abstract Interpretation

Formal Verification Approaches

- Hoare 1969: wrap the code of interest with preconditions and postconditions, then prove that postconditions are met
- Clarke, Emerson et Sifakis 1974: model checking
- Cousot(s) 1977: Abstract Interpretation

Properties of Interest

- run time errors: overflow, division by zero, square root of negatives, etc.
- robustness and stability of algorithms: linear and non linear recursive schemes, filters, etc.

Formal Verification Approaches

- Hoare 1969: wrap the code of interest with preconditions and postconditions, then prove that postconditions are met
- Clarke, Emerson et Sifakis 1974: model checking
- Cousot(s) 1977: Abstract Interpretation

Properties of Interest

- run time errors: overflow, division by zero, square root of negatives, etc.
- robustness and stability of algorithms: linear and non linear recursive schemes, filters, etc.

Formal Verification Approaches

- Hoare 1969: wrap the code of interest with preconditions and postconditions, then prove that postconditions are met
- Clarke, Emerson et Sifakis 1974: model checking
- Cousot(s) 1977: Abstract Interpretation

Properties of Interest

- run time errors: overflow, division by zero, square root of negatives, etc.
- robustness and stability of algorithms: linear and non linear recursive schemes, filters, etc.

Formal Verification Approaches

- Hoare 1969: wrap the code of interest with preconditions and postconditions, then prove that postconditions are met
- Clarke, Emerson et Sifakis 1974: model checking
- Cousot(s) 1977: Abstract Interpretation

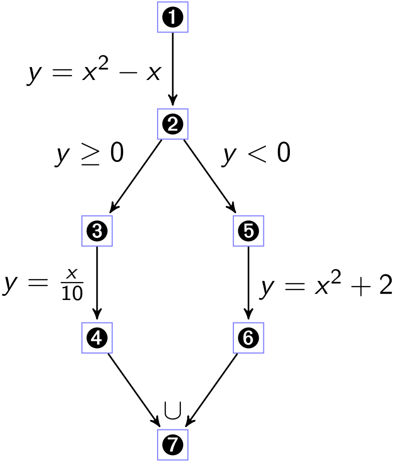
Properties of Interest

- run time errors: overflow, division by zero, square root of negatives, etc.
- robustness and stability of algorithms: linear and non linear recursive schemes, filters, etc.

- Program semantics formalized as a fixpoint of a monotonic operator in a complete partially ordered set (exemplified later),
- Fully automated,
- Industrial tools exists : Polyspace Verifier (MathWorks), AstrÃľe (ENS/ABSINT), Fluctuat (CEA), aIT (ABSINT), F-Soft (Nec Labs) . . .

Challenge

find the *suitable* abstract domain for the properties of interest.



$$
\begin{cases}
\mathcal{X}_1 = [\![\mathcal{V} \to \mathbb{I}]\!]^\flat \\
\mathcal{X}_2 = [\![y \leftarrow x^2 - x]\!]^\flat(\mathcal{X}_1) \\
\mathcal{X}_3 = [\![y \geq 0]\!]^\flat(\mathcal{X}_2) \\
\mathcal{X}_4 = [\![y \leftarrow \frac{x}{10}]\!]^\flat(\mathcal{X}_3) \\
\mathcal{X}_5 = [\![y < 0]\!]^\flat(\mathcal{X}_2) \\
\mathcal{X}_6 = [\![y \leftarrow x^2 + 2]\!]^\flat(\mathcal{X}_5) \\
\mathcal{X}_7 = \mathcal{X}_6 \cup \mathcal{X}_4
\end{cases}
$$

Solving the equations system

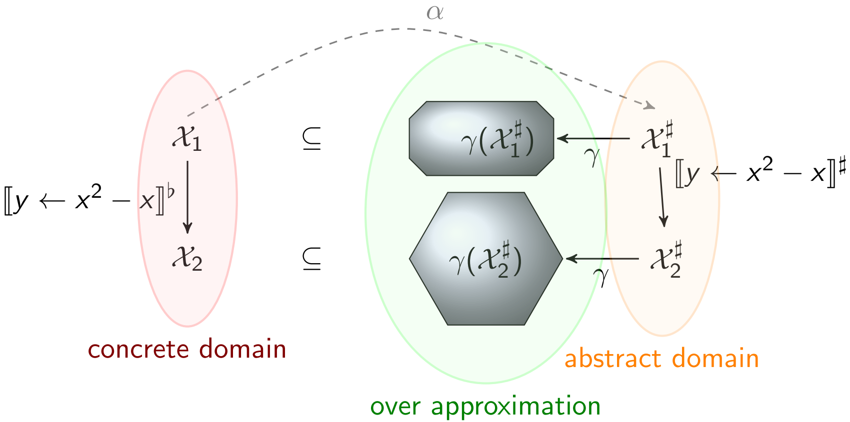
- $D = (\wp(\mathcal{V} \to \mathbb{I}), \subseteq, \cup, \cap, \emptyset, (\mathcal{V} \to \mathbb{I}))$ is a **complete lattice**
- each operator $\mathcal{X} \mapsto \mathcal{F}(\mathcal{X})$ is monotonic

➜ **Tarski Theorem** ensures the existence of a least fixpoint for $\mathcal{F}$

➜ **Kleene Iteration Technique** reaches the least fixpoint

Issues

- $\wp(\mathcal{V} \to \mathbb{I})$ is non representable in finite memory,
- $[\![.]\!]^\flat$ are non computable,
- Iterations over the lattice may be transfinite.



α
$\mathcal{X}_1$
$[\![y \leftarrow x^2 - x]\!]^\flat$
$\mathcal{X}_2$
⊆
⊆
$\gamma(\mathcal{X}_1^\sharp)$
$\gamma(\mathcal{X}_2^\sharp)$
γ
γ
$\mathcal{X}_1^\sharp$
$[\![y \leftarrow x^2 - x]\!]^\sharp$
$\mathcal{X}_2^\sharp$
concrete domain
abstract domain
over approximation

- lattice-like structure:
  - abstract objects
  - order relation (preorder over abstract objects)
  - monotonic concretisation function (γ)
- Transfer Functions
  - evaluation of arithmetic expressions ($[\![x^2 - x]\!]^\sharp$)
  - assignment ($\mathcal{X}_2 = [\![y \leftarrow x^2 - x]\!]^\sharp(\mathcal{X}_1)$)
  - upper bound (join) ($\mathcal{X}_7 = \mathcal{X}_6 \cup \mathcal{X}_4$)
  - over-approximation of lower bounds (meet) ($\mathcal{X}_3 = [\![y \geq 0]\!]^\sharp \mathcal{X}_2 =$ "$\mathcal{X}_3 = \mathcal{X}_2 \cap [\![y \geq 0]\!]^\sharp \top^\sharp$")
- Convergence acceleration (widening)