Abstract Interpretation

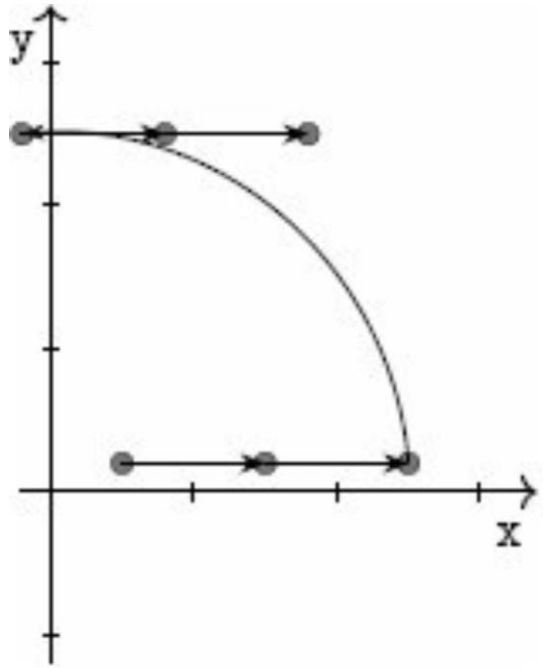
Sergey Mechtaev
mechtaev@pku.edu.cn
Peking University

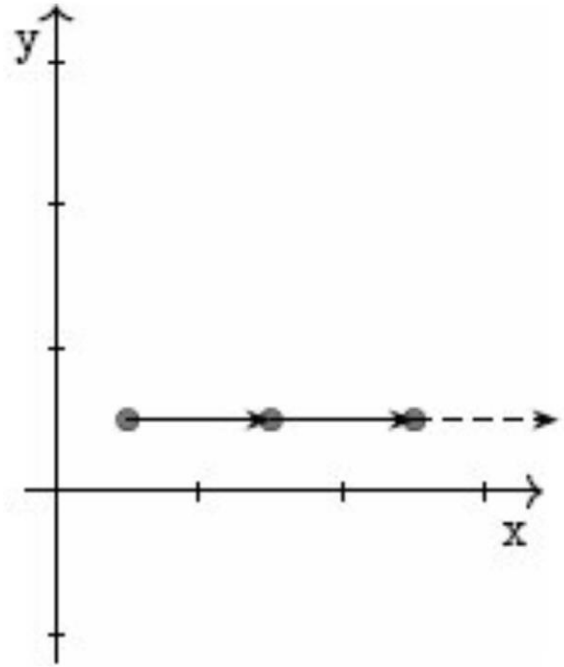
Example

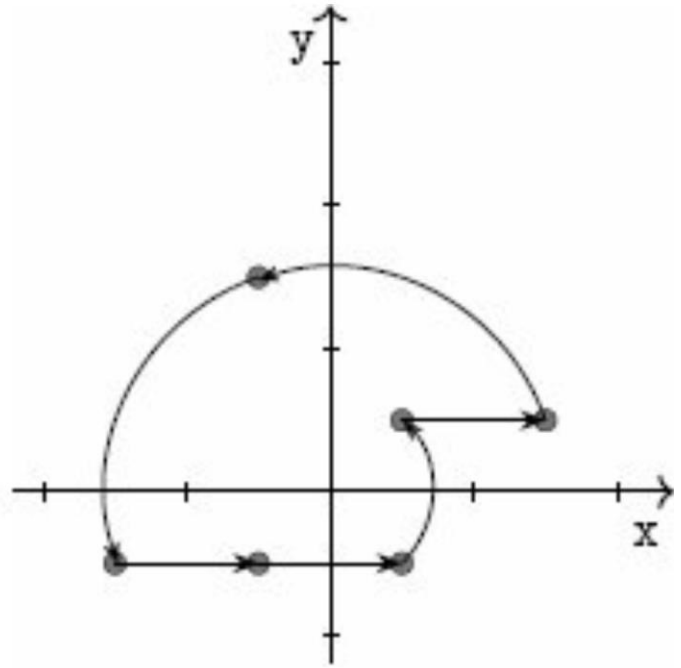
Consider programs manipulating points in two-dimensional space:

```
init([0,1] × [0,1]);
translation(1,0);
iter{
    {
        translation(1,0)
    }or{
        rotation(0,0,90°)
    }
}
```

Possible Executions

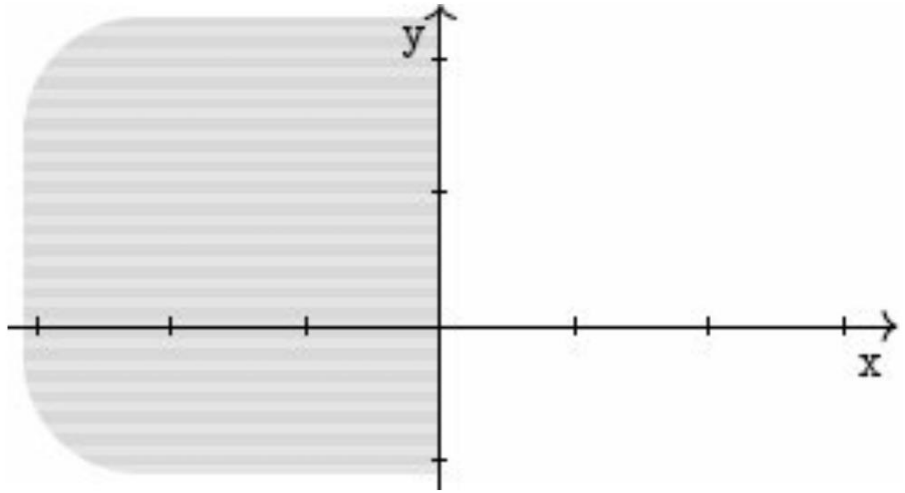






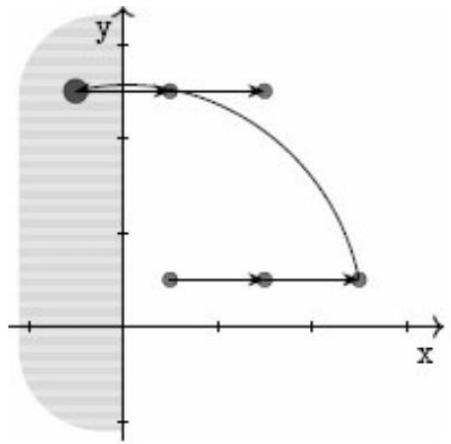
State Reachability Problem

Can program reach a state where x < 0?

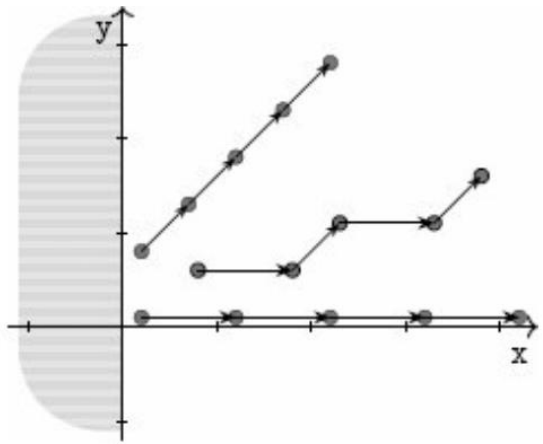


Correct and Incorrect Executions

```
init([0,1] × [0,1]);
iter{
    {
        translation(1,0);
    }or{
        translation(0.5,0.5);
    }
}
```



(a) An incorrect execution



(b) Correct executions

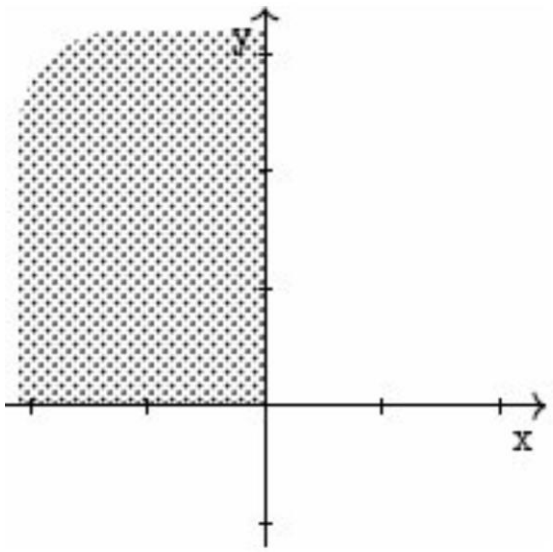
Abstraction and Concretisation

Abstraction is a set of logical properties of program states, which are called **abstract elements**. A set of abstract elements is **abstract domain**.

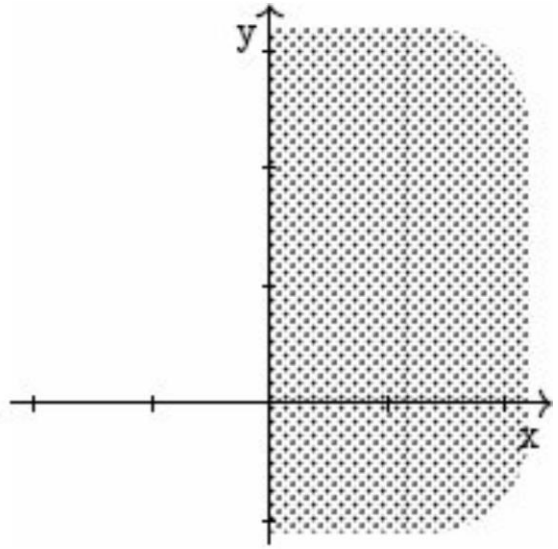
Covent an abstract element a, the set of program states that satisfy it is called concretisation, denoted as $\gamma(a)$

Sign Abstraction

Abstract elements: $[x \geq 0]$, $[x \leq 0, y \geq 0]$, etc



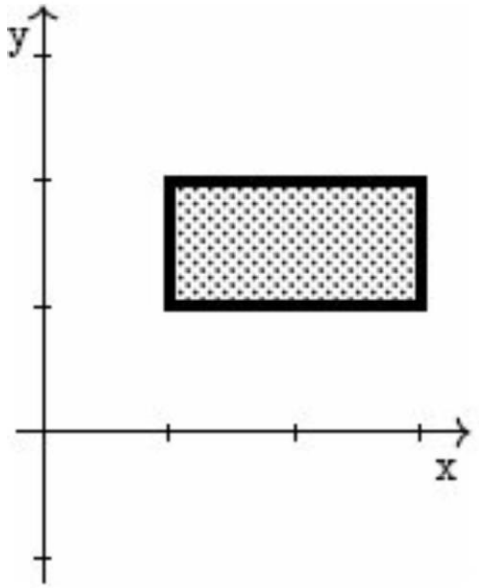
(a) Concretization of $[\mathbf{x} \leq 0, \mathbf{y} \geq 0]$



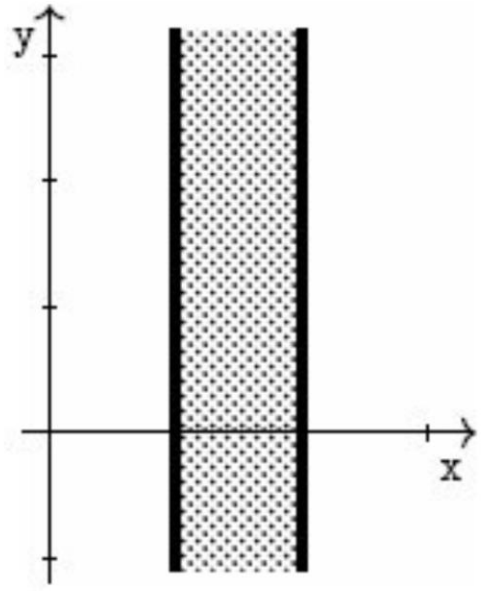
(b) Concretization of $[\mathbf{x} \geq 0]$

Interval Abstraction

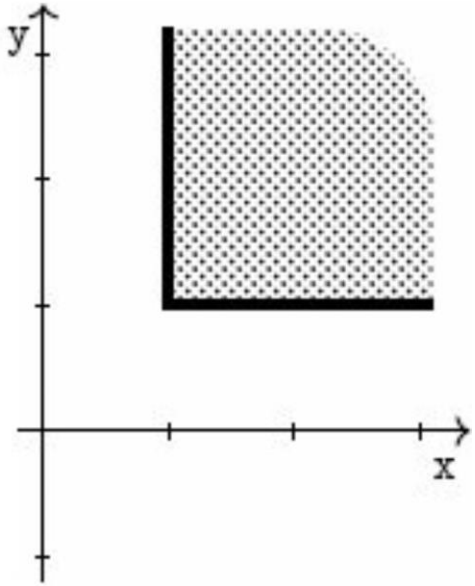
- a_0 corresponds to $1 \leq x \leq 3$ and $1 \leq y \leq 2$
- a_1 corresponds to $1 \leq x \leq 2$
- a_2 corresponds to $1 \leq x$ and $1 \leq y$



(a) Concretization of a_0

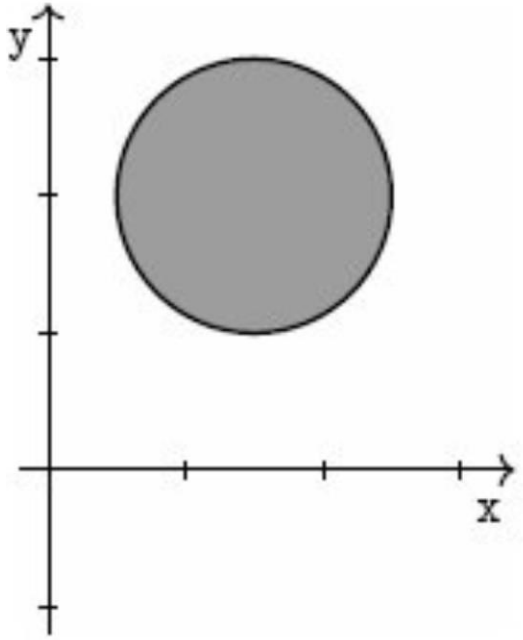


(b) Concretization of a_1

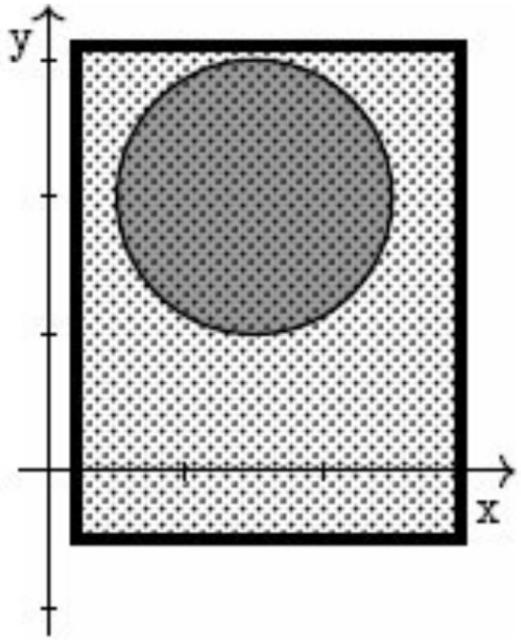


(c) Concretization of a_2

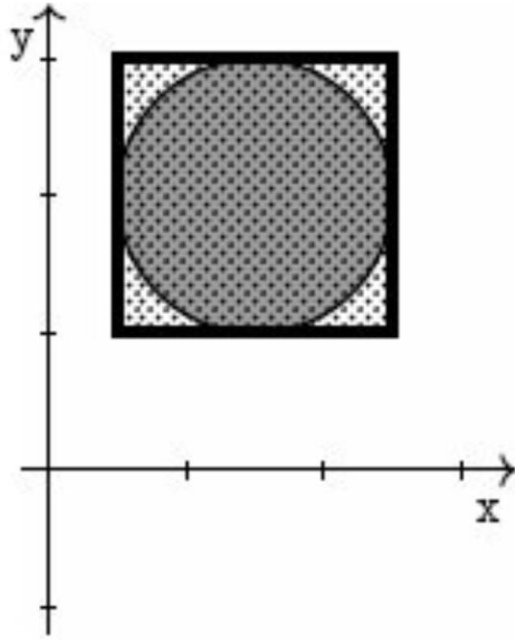
Best Abstraction



(a) A concrete set



(b) Abstractions

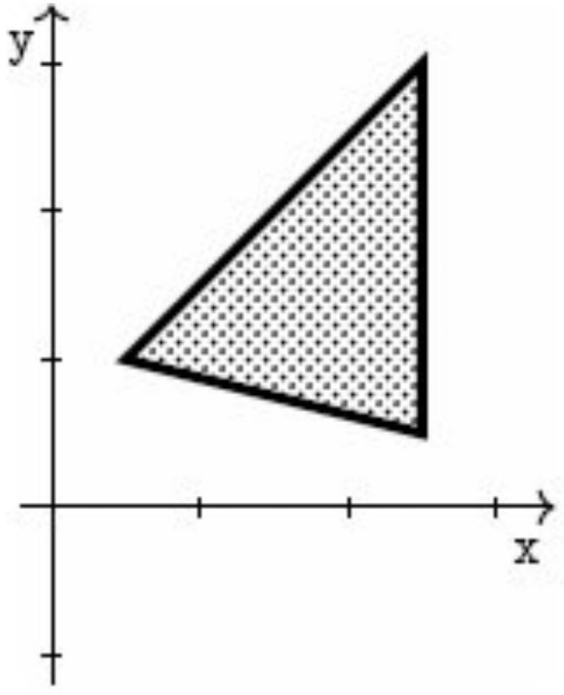


(c) Best abstraction

Convex Polyhedra Abstraction

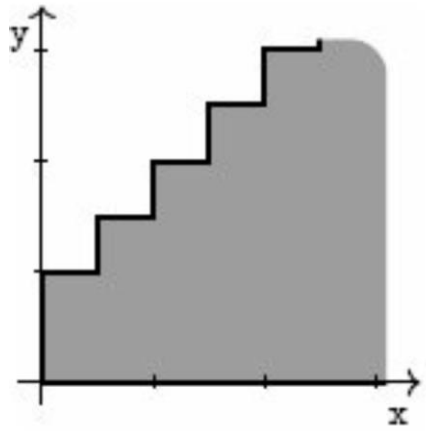
Defined as conjunction of linear inequalities

$x - y \geq -0.5$
$x \leq 2.5$
$x + 4y \geq 4.5$

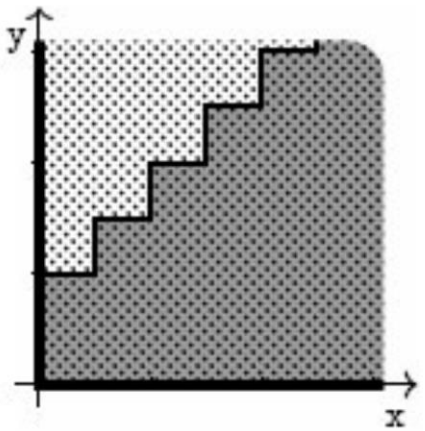


Comparing Abstract Domains

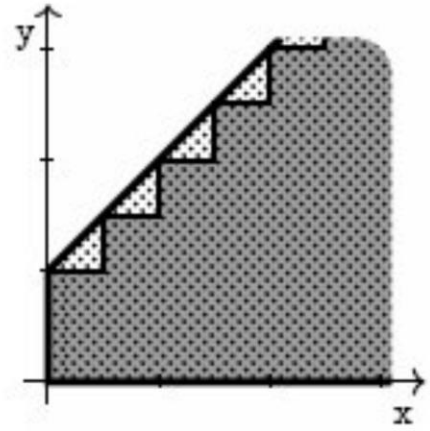
```
init([0,1] × [0,1]);
iter{
    {
        translation(1,0);
    }or{
        translation(0.5,0.5);
    }
}
```



(a) Reachable states



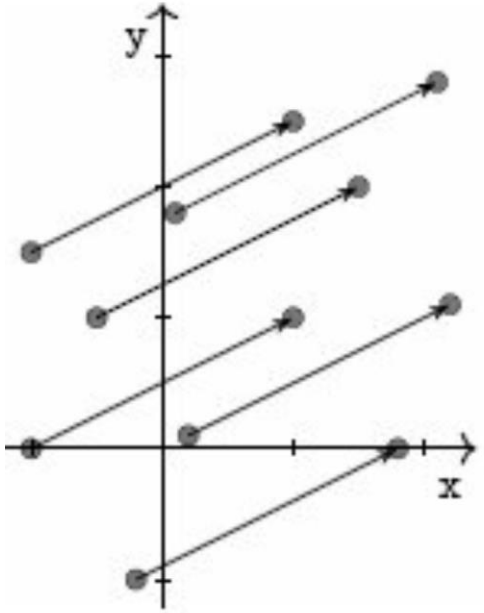
(b) Intervals abstraction



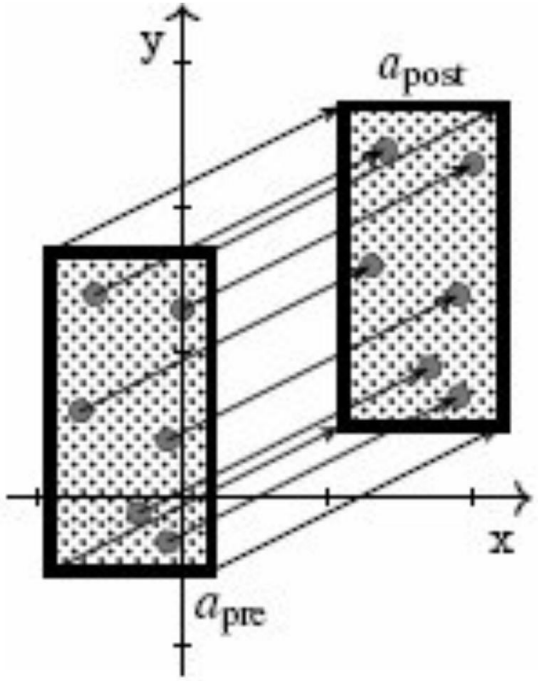
(c) Convex polyhedra abstraction

Abstraction of Post-conditions

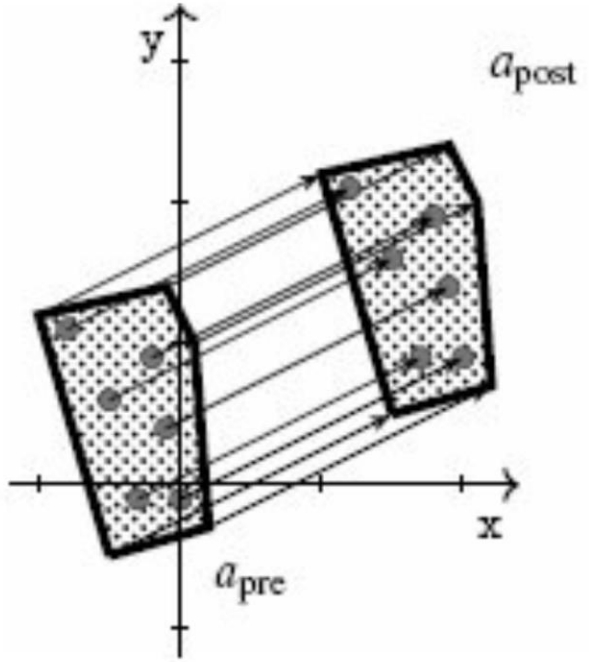
- `translation(u,v)`



(a) Concrete semantics



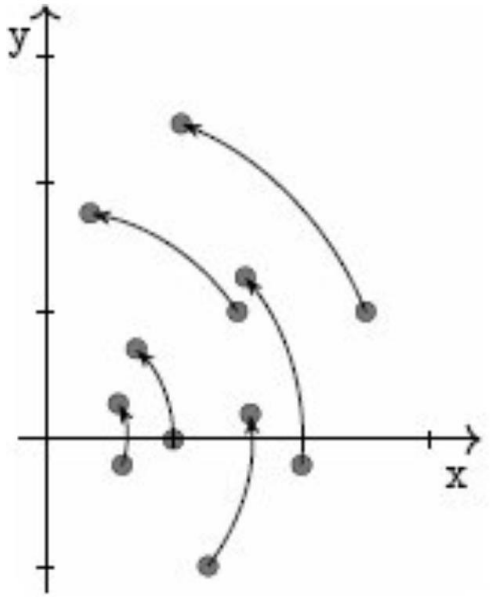
(b) Intervals



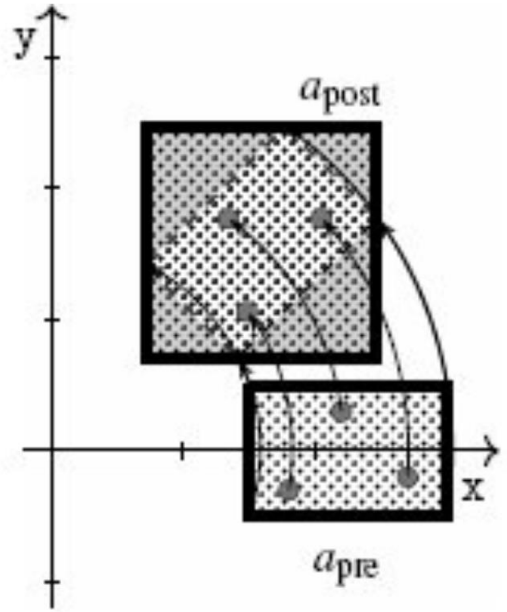
(c) Convex polyhedra

Abstraction of Post-conditions

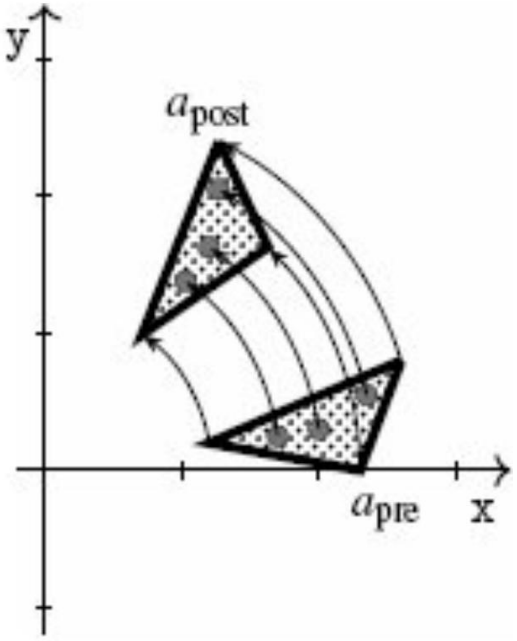
- `rotation`(u,v,θ)



(a) Concrete semantics



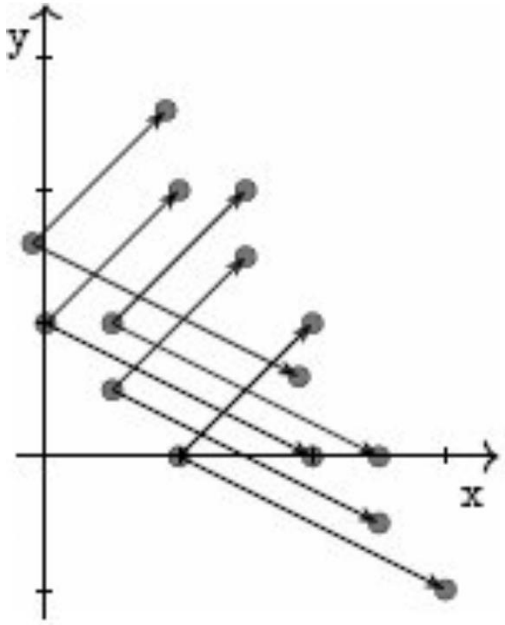
(b) Intervals



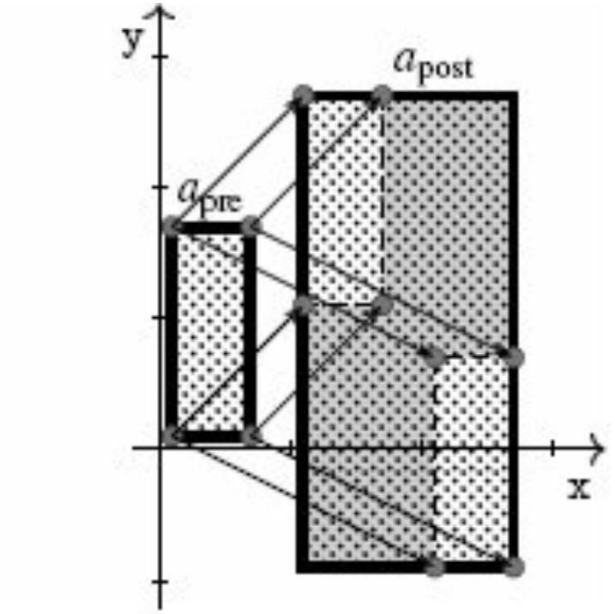
(c) Convex polyhedra

Non-Deterministic Choice

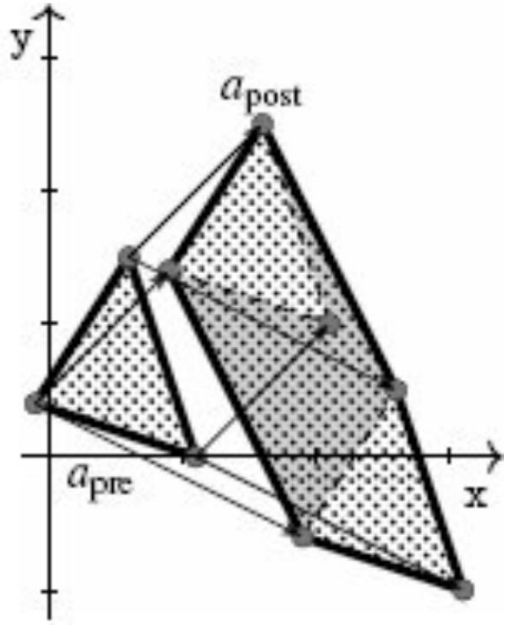
- `translation`(2,1) or `translation`(-2,-1)



(a) Concrete semantics



(b) Intervals



(c) Convex polyhedra

Iterations

$$p ::= \begin{cases} \textbf{iter}\{ \\ \quad b \\ \} \end{cases}$$

$\{\}$

$\textbf{or}\{b\}$

$\textbf{or}\{b;b\}$

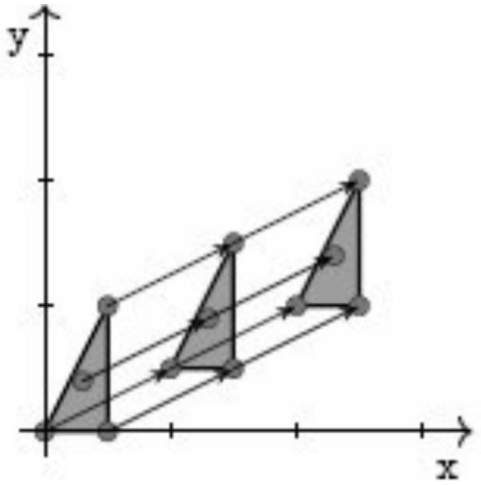
$\textbf{or}\{b;b;b\}$

$\textbf{or}\{b;b;b;b\}$

$\vdots$

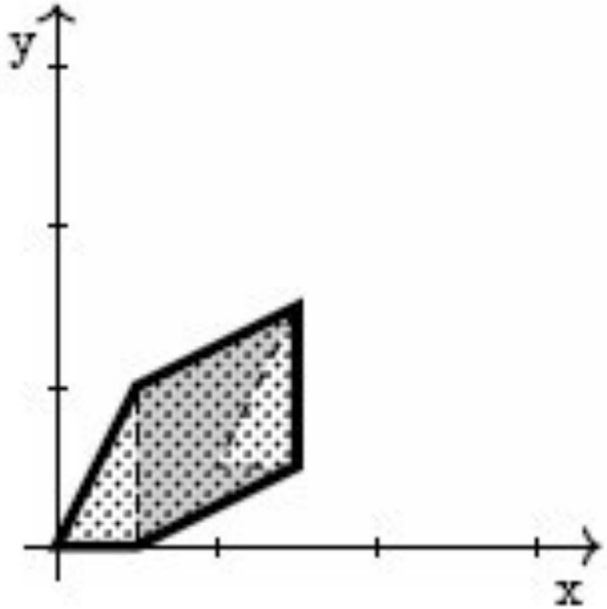
Example with loop

```
init({(x,y) | 0 ≤ y ≤ 2x and x ≤ 0.5});
iter{
    translation(1,0.5)
}
```

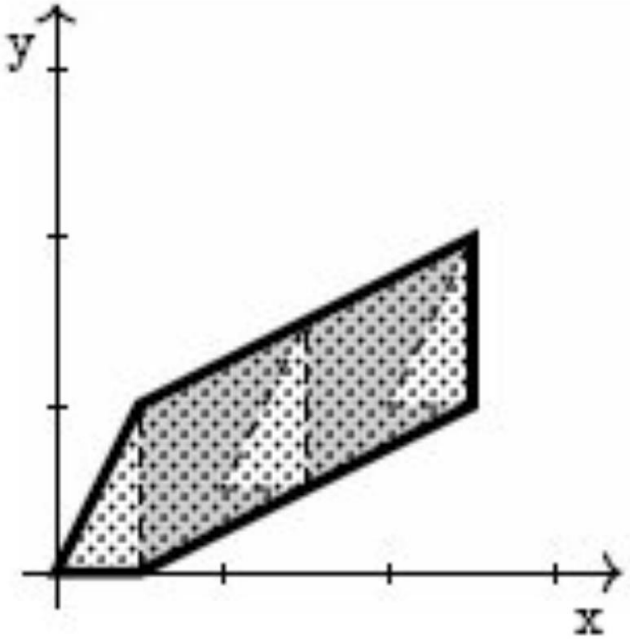


(a) Concrete semantics

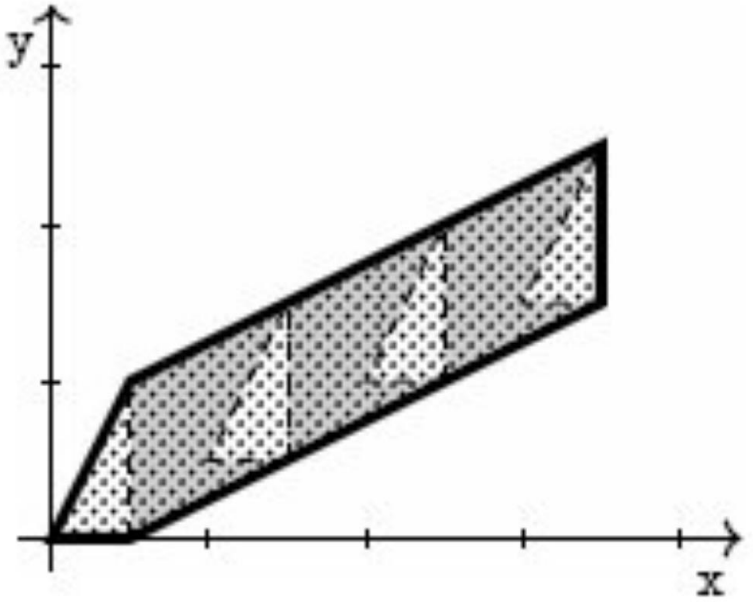
Infinite Iterations



Iteration 1

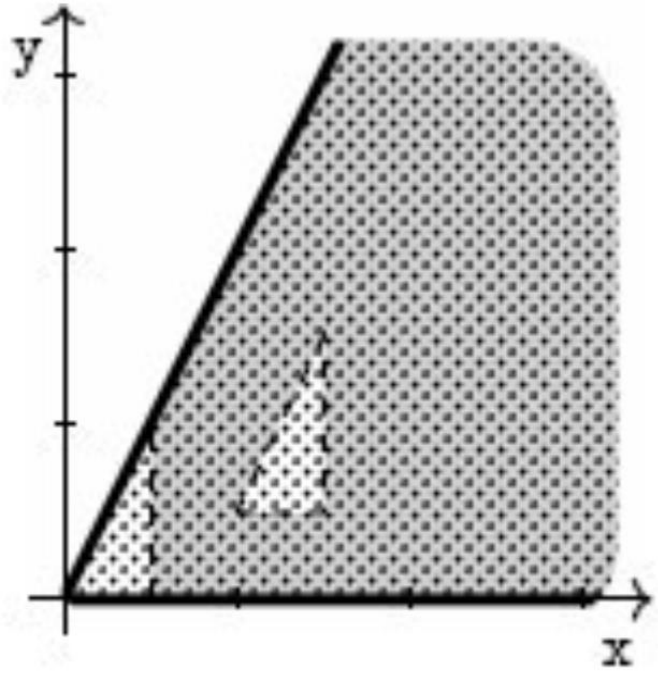


Iteration 2

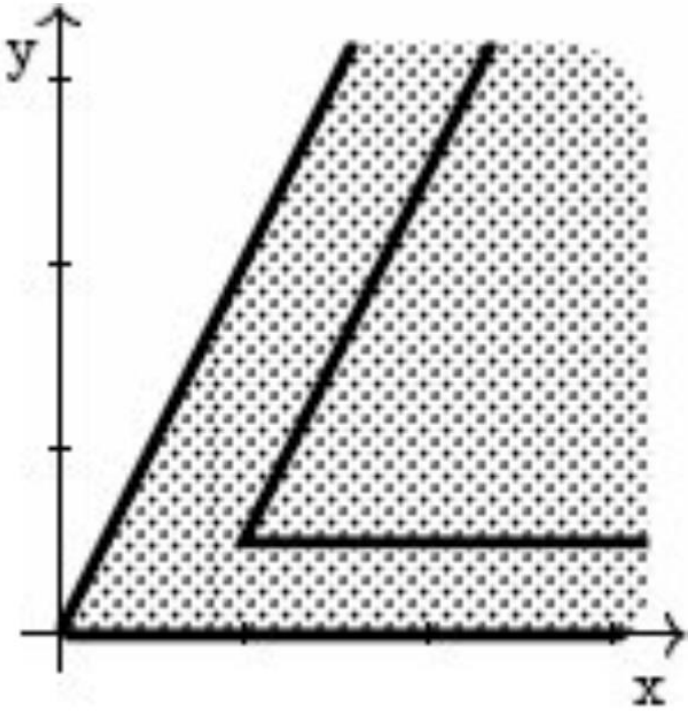


Iteration 3

Widening To Make Analysis Converge



Iteration 1



Iteration 2 (reach fixed point)