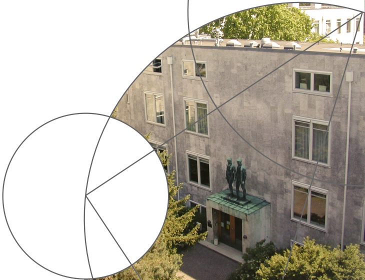
Department of Computer Science

Interval and polyhedral analysis

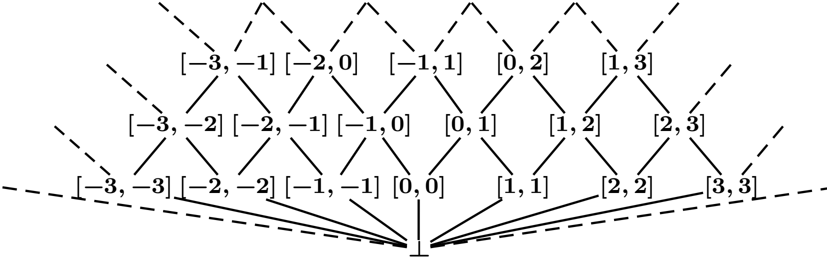
Thomas Jensen



Plan

1. Interval analysis
2. Widening and narrowing
3. Polyhedral abstract interpretation

The lattice of intervals



The lattice of intervals

Elements :

$$\mathrm{Itv} \stackrel{\mathrm{def}}{=} \{\, [a,b] \mid a,b \in \overline{\mathbb{Z}},\ a \leqslant b \,\} \cup \{\bot\} \quad \text{with } \overline{\mathbb{Z}} = \mathbb{Z} \cup \{-\infty, +\infty\}$$

Order :

$$\frac{I \in \mathrm{Itv}}{\bot \sqsubseteq_{\mathrm{Itv}} I} \qquad \frac{c \leqslant a \qquad b \leqslant d \qquad a,b,c,d \in \overline{\mathbb{Z}}}{[a,b] \sqsubseteq_{\mathrm{Itv}} [c,d]}$$

Lattice operations :

$$\begin{aligned}
I \sqcup_{\mathrm{Itv}} \bot &\stackrel{\mathrm{def}}{=} I,\ \forall I \in \mathrm{Itv} \\
\bot \sqcup_{\mathrm{Itv}} I &\stackrel{\mathrm{def}}{=} I,\ \forall I \in \mathrm{Itv} \\
[a,b] \sqcup_{\mathrm{Itv}} [c,d] &\stackrel{\mathrm{def}}{=} [\min(a,c), \max(b,d)]
\end{aligned}$$

$$\begin{aligned}
I \sqcap_{\mathrm{Itv}} \bot &\stackrel{\mathrm{def}}{=} \bot,\ \forall I \in \mathrm{Itv} \\
\bot \sqcap_{\mathrm{Itv}} I &\stackrel{\mathrm{def}}{=} \bot,\ \forall I \in \mathrm{Itv} \\
[a,b] \sqcap_{\mathrm{Itv}} [c,d] &\stackrel{\mathrm{def}}{=} \rho_{\mathrm{Itv}}([\max(a,c), \min(b,d)])
\end{aligned}$$

Normalizer : $\rho_{\mathrm{Itv}} \in (\overline{\mathbb{Z}} \times \overline{\mathbb{Z}}) \to \mathrm{Itv}$ defined by

$$\rho_{\mathrm{Itv}}(a,b) = \begin{cases} [a,b] & \text{if } a \leqslant b, \\ \perp & \text{otherwise} \end{cases}$$

Least and greatest element :

$$\begin{aligned} \perp_{\mathrm{Itv}} &\stackrel{\text{def}}{=} \perp \\ \top_{\mathrm{Itv}} &\stackrel{\text{def}}{=} [-\infty, +\infty] \end{aligned}$$

Abstraction and concretisation :

$$\begin{array}{lcll} \alpha_{\mathrm{Itv}}(S) & \stackrel{\text{def}}{=} & \perp & \text{if } S = \emptyset \\ \alpha_{\mathrm{Itv}}(S) & \stackrel{\text{def}}{=} & [\min(S), \max(S)] & \text{otherwise} \end{array}$$

$$\begin{aligned} \gamma_{\mathrm{Itv}}(\perp) &\stackrel{\text{def}}{=} \emptyset \\ \gamma_{\mathrm{Itv}}([a,b]) &\stackrel{\text{def}}{=} \{\, z \in \mathbb{Z} \mid a \leqslant z \text{ and } z \leqslant b \,\} \end{aligned}$$

Abstraction of basic functions

All operators are *strict* : they return $\perp$ if one of their arguments is $\perp$.

$$
\begin{array}{rcl}
+^\sharp([a,b],[c,d]) & = & [a+c, b+d] \\
-^\sharp([a,b],[c,d]) & = & [a-d, b-c] \\
\times^\sharp([a,b],[c,d]) & = & [\min(ac,ad,bc,bd), \max(ac,ad,bc,bd)]
\end{array}
$$

$$
\mathrm{const}(n)^\sharp = [n,n]
$$

Example :

$$
+^\sharp([2,\infty],[3,4]) = [5,\infty]
$$

Abstract comparison operators

Update knowledge about the value of variables.

Example : if $x \mapsto [2,5]$ and $y \mapsto [3,7]$ and we know that the test $x = y$ succeeds then we can refine our description of x and y :

$$[\![=]\!]\downarrow^{\sharp}_{\text{comp}}\ ([2,5],[3,7]) = ([3,5],[3,5])$$

In general :

$$\begin{array}{lcl}
[\![=]\!]\downarrow^{\sharp}_{\text{comp}}\ ([a,b],[c,d]) & = & ([a,b] \sqcap_{\text{Itv}} [c,d], [a,b] \sqcap_{\text{Itv}} [c,d]) \\
[\![<]\!]\downarrow^{\sharp}_{\text{comp}}\ ([a,b],[c,d]) & = & ([a,b] \sqcap_{\text{Itv}} [-\infty, d-1], [a+1,+\infty] \sqcap_{\text{Itv}} [c,d]) \\
[\![\leqslant]\!]\downarrow^{\sharp}_{\text{comp}}\ ([a,b],[c,d]) & = & ([a,b] \sqcap_{\text{Itv}} [-\infty, d], [a,+\infty] \sqcap_{\text{Itv}} [c,d]) \\
[\![\neq]\!]\downarrow^{\sharp}_{\text{comp}}\ ([a,b],[c,d]) & = & ?\ \textit{exercise...}
\end{array}$$

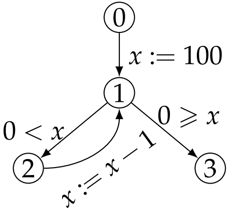
Abstract interpretation with intervals

```
x := 100;

while 0 < x {

  x := x − 1;
}
```

$$X_1 = [100, 100] \sqcup_{\mathrm{Itv}} \left(X_2 -^{\sharp} [1,1]\right)$$

$$X_2 = [1, +\infty] \sqcap_{\mathrm{Itv}} X_1$$

$$X_3 = [-\infty, 0] \sqcap_{\mathrm{Itv}} X_1$$



Example : fixpoint iteration

$$
\begin{aligned}
X_1 &= [100,100] \sqcup_{\mathrm{Itv}} \left(X_2 -^\sharp [1,1]\right)\\
X_2 &= [1,+\infty] \sqcap_{\mathrm{Itv}} X_1\\
X_3 &= [-\infty,0] \sqcap_{\mathrm{Itv}} X_1
\end{aligned}
$$

Iteration strategy : $1 \to 2 \to 3 \to 1 \to 2 \to \cdots$

$$
\begin{aligned}
X_1^0 &= \bot \quad & X_1^{n+1} &= [100,100] \sqcup_{\mathrm{Itv}} \left(X_2^n -^\sharp [1,1]\right)\\
X_2^0 &= \bot \quad & X_2^{n+1} &= [1,+\infty] \sqcap_{\mathrm{Itv}} X_1^{n+1}\\
X_3^0 &= \bot \quad & X_3^{n+1} &= [-\infty,0] \sqcap_{\mathrm{Itv}} X_1^{n+1}
\end{aligned}
$$

X_1	$\bot$	$\cdots$
X_2	$\bot$	$\cdots$
X_3	$\bot$	$\cdots$

Example : fixpoint iteration

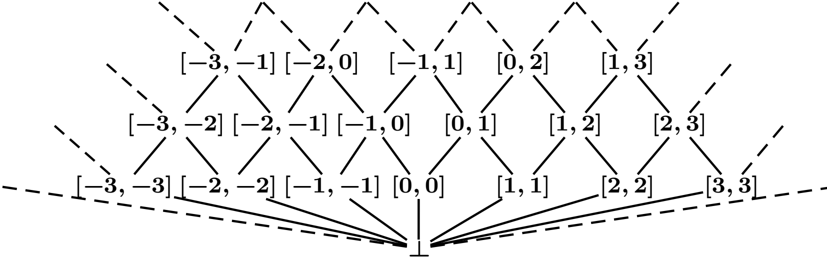
$$
\begin{aligned}
X_1 &= [100,100] \sqcup_{\mathrm{Itv}} \left(X_2 -^\sharp [1,1]\right)\\
X_2 &= [1,+\infty] \sqcap_{\mathrm{Itv}} X_1\\
X_3 &= [-\infty,0] \sqcap_{\mathrm{Itv}} X_1
\end{aligned}
$$

Iteration strategy : $1 \to 2 \to 3 \to 1 \to 2 \to \cdots$

$$
\begin{array}{ll}
X_1^0 = \bot & X_1^{n+1} = [100,100] \sqcup_{\mathrm{Itv}} \left(X_2^n -^\sharp [1,1]\right)\\
X_2^0 = \bot & X_2^{n+1} = [1,+\infty] \sqcap_{\mathrm{Itv}} X_1^{n+1}\\
X_3^0 = \bot & X_3^{n+1} = [-\infty,0] \sqcap_{\mathrm{Itv}} X_1^{n+1}
\end{array}
$$

X_1	$\bot$	$[100,100]$	$[99,100]$	$[98,100]$	$[97,100]$	$\cdots$	$[1,100]$	$[0,100]$
X_2	$\bot$	$[100,100]$	$[99,100]$	$[98,100]$	$[97,100]$	$\cdots$	$[1,100]$	$[1,100]$
X_3	$\bot$	$\bot$	$\bot$	$\bot$	$\bot$	$\cdots$	$\bot$	$[0,0]$

Convergence problem



The lattice of intervals has infinite ascending chains.

$$\perp \sqsubset [0,0] \sqsubset [0,1] \sqsubset \cdots \sqsubset [0,n] \sqsubset \cdots .$$

.

Solution : dynamic approximation

- we extrapolate the limit thanks to a widening operator ∇

$$\perp \sqsubset [0,0] \sqsubset [0,1] \sqsubset [0,2] \sqsubset [0,+\infty] = [0,2]\nabla[0,3]$$

Plan

1. Interval analysis
2. Widening and narrowing
3. Polyhedral abstract interpretation

Fixpoint approximation

Lemma

Let $(A, \sqsubseteq, \sqcup, \sqcap)$ be a complete lattice and f a monotone operator on A. If a is a post-fixpoint of f (i.e. $f(a) \sqsubseteq a$), then

$$\mathrm{lfp}(f) \sqsubseteq a$$

.

We may want to over-approximate $\mathrm{lfp}(f)$ when :

- The lattice does not satisfies the ascending chain condition, the iteration $\bot, f(\bot), \ldots, f^n(\bot), \ldots$ may never terminate.
- The ascending chain condition is satisfied but the iteration chain is too long to allow an efficient computation.
- The underlying lattice is not complete, so the limits of the ascending iterations do not necessarily belong to the abstraction domain.

Widening

Idea : the standard iteration is of the form

$$x^0 = \bot, \quad x^{n+1} = F(x^n) = x^n \sqcup F(x^n)$$

We will replace it by something of the form

$$y^0 = \bot, \quad y^{n+1} = y^n \nabla F(y^n)$$

such that

(i) (y^n) is increasing,

(ii) $x^n \sqsubseteq y^n$, for all n,

(iii) and (y^n) stabilizes after a finite number of steps.

Widening : definition

A widening is an operator $\nabla : L \times L \to L$ such that

- $\forall x, x' \in L, x \sqcup x' \sqsubseteq x \nabla x'$ (implies (i) & (ii))
- If $x^0 \sqsubseteq x^1 \sqsubseteq \ldots$ is an increasing chain, then the increasing chain $y^0 = x^0, y^{n+1} = y^n \nabla x^{n+1}$ stabilizes after a finite number of steps (implies (iii)).

Usage : we replace $x^0 = \bot, x^{n+1} = F(x^n)$
by $y^0 = \bot, y^{n+1} = y^n \nabla F(y^n)$

Widening : theorem

Theorem

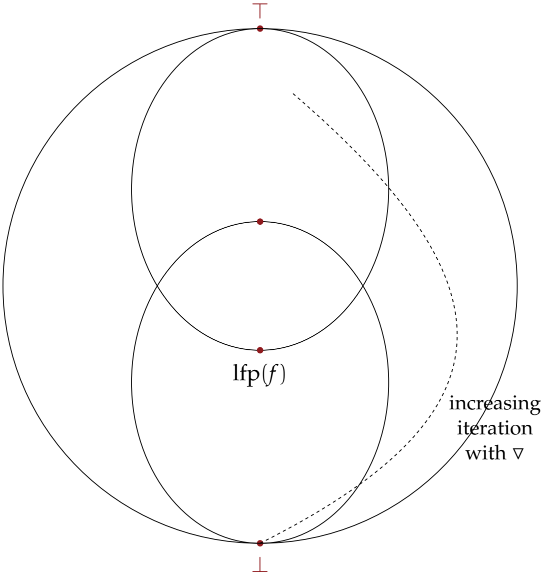
Let

- L *be a complete lattice,*
- $F : L \to L$ *be a monotone function and*
- $\nabla : L \times L \to L$ *a widening operator.*

Then, the chain $y^0 = \bot, y^{n+1} = y^n \nabla F(y^n)$ *stabilizes after a finite number of steps at a post-fixpoint* y *of* F*.*

Corollary : $\mathrm{lfp}(F) \sqsubseteq y$.

Scheme



Example : widening on intervals

Idea : as soon as a bound is not stable, we extrapolate it by $+\infty$ (or $-\infty$).
After such an extrapolation, the bound can't move any more.

Definition :

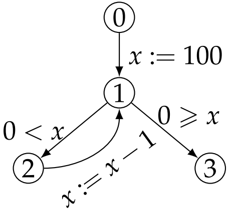
$$\begin{array}{rcl}
[a,b]\nabla_{\mathrm{Itv}}[a',b'] & = & [\quad \text{if } a' < a \text{ then } -\infty \text{ else } a, \\
 & & \quad\;\; \text{if } b' > b \text{ then } +\infty \text{ else } b\,] \\
\perp\nabla_{\mathrm{Itv}}[a',b'] & = & [a',b'] \\
I\ \nabla_{\mathrm{Itv}}\ \perp & = & I
\end{array}$$

Examples :

$[-3,4]\nabla_{\mathrm{Itv}}[-3,2] = [-3,4]$
$[-3,4]\nabla_{\mathrm{Itv}}[-3,5] = [-3,+\infty]$

Example

```
x := 100;

while 0 < x {

  x := x − 1;
}
```



$$X_1 = [100, 100] \sqcup_{\text{Itv}} \left(X_2 -^\sharp [1, 1]\right)$$

$$X_2 = [1, +\infty] \sqcap_{\text{Itv}} X_1$$

$$X_3 = [-\infty, 0] \sqcap_{\text{Itv}} X_1$$

Example : without widening

$$
\begin{aligned}
X_1 &= [100, 100] \sqcup_{\mathrm{Itv}} \left(X_2 -^\sharp [1, 1]\right) \\
X_2 &= [1, +\infty] \sqcap_{\mathrm{Itv}} X_1 \\
X_3 &= [-\infty, 0] \sqcap_{\mathrm{Itv}} X_1
\end{aligned}
$$

Iteration strategy : $1 \rightarrow 2 \rightarrow 3 \rightarrow 1 \rightarrow 2 \rightarrow \cdots$

$$
\begin{array}{ll}
X_1^0 = \perp & X_1^{n+1} = [100, 100] \sqcup_{\mathrm{Itv}} \left(X_2^n -^\sharp [1, 1]\right) \\
X_2^0 = \perp & X_2^{n+1} = [1, +\infty] \sqcap_{\mathrm{Itv}} X_1^{n+1} \\
X_3^0 = \perp & X_3^{n+1} = [-\infty, 0] \sqcap_{\mathrm{Itv}} X_1^{n+1}
\end{array}
$$

X_1	$\perp$	$\cdots$
X_2	$\perp$	$\cdots$
X_3	$\perp$	$\cdots$

Example : without widening

$$
\begin{aligned}
X_1 &= [100,100] \sqcup_{\mathrm{Itv}} \left(X_2 -^\sharp [1,1]\right) \\
X_2 &= [1,+\infty] \sqcap_{\mathrm{Itv}} X_1 \\
X_3 &= [-\infty,0] \sqcap_{\mathrm{Itv}} X_1
\end{aligned}
$$

Iteration strategy : $1 \to 2 \to 3 \to 1 \to 2 \to \cdots$

$$
\begin{array}{ll}
X_1^0 = \bot & X_1^{n+1} = [100,100] \sqcup_{\mathrm{Itv}} \left(X_2^n -^\sharp [1,1]\right) \\
X_2^0 = \bot & X_2^{n+1} = [1,+\infty] \sqcap_{\mathrm{Itv}} X_1^{n+1} \\
X_3^0 = \bot & X_3^{n+1} = [-\infty,0] \sqcap_{\mathrm{Itv}} X_1^{n+1}
\end{array}
$$

X_1	$\bot$	$[100,100]$	$[99,100]$	$[98,100]$	$[97,100]$	$\cdots$	$[1,100]$	$[0,100]$
X_2	$\bot$	$[100,100]$	$[99,100]$	$[98,100]$	$[97,100]$	$\cdots$	$[1,100]$	$[1,100]$
X_3	$\bot$	$\bot$	$\bot$	$\bot$	$\bot$	$\cdots$	$\bot$	$[0,0]$

Example : with widening at each node of the cfg

$$
\begin{aligned}
X_1 &= [100, 100] \sqcup_{\text{Itv}} \left(X_2 -^\sharp [1, 1]\right) \\
X_2 &= [1, +\infty] \sqcap_{\text{Itv}} X_1 \\
X_3 &= [-\infty, 0] \sqcap_{\text{Itv}} X_1
\end{aligned}
$$

Iteration strategy : $1 \to 2 \to 3 \to 1 \to 2 \to \cdots$

$$
\begin{array}{ll}
X_1^0 = \perp & X_1^{n+1} = X_1^n \nabla_{\text{Itv}} \left([100, 100] \sqcup_{\text{Itv}} \left(X_2^n -^\sharp [1, 1]\right)\right) \\
X_2^0 = \perp & X_2^{n+1} = X_2^n \nabla_{\text{Itv}} \left([1, +\infty] \sqcap_{\text{Itv}} X_1^{n+1}\right) \\
X_3^0 = \perp & X_3^{n+1} = X_3^n \nabla_{\text{Itv}} \left([-\infty, 0] \sqcap_{\text{Itv}} X_1^{n+1}\right)
\end{array}
$$

X_1	$\perp$
X_2	$\perp$
X_3	$\perp$

Example : with widening at each node of the cfg

$$
\begin{aligned}
X_1 &= [100, 100] \sqcup_{\mathrm{Itv}} \left(X_2 -^{\sharp} [1, 1]\right) \\
X_2 &= [1, +\infty] \sqcap_{\mathrm{Itv}} X_1 \\
X_3 &= [-\infty, 0] \sqcap_{\mathrm{Itv}} X_1
\end{aligned}
$$

Iteration strategy : $1 \to 2 \to 3 \to 1 \to 2 \to \cdots$

$$
\begin{array}{ll}
X_1^0 = \perp & X_1^{n+1} = X_1^n \nabla_{\mathrm{Itv}} \left([100, 100] \sqcup_{\mathrm{Itv}} \left(X_2^n -^{\sharp} [1, 1]\right)\right) \\
X_2^0 = \perp & X_2^{n+1} = X_2^n \nabla_{\mathrm{Itv}} \left([1, +\infty] \sqcap_{\mathrm{Itv}} X_1^{n+1}\right) \\
X_3^0 = \perp & X_3^{n+1} = X_3^n \nabla_{\mathrm{Itv}} \left([-\infty, 0] \sqcap_{\mathrm{Itv}} X_1^{n+1}\right)
\end{array}
$$

X_1	$\perp$	$[100, 100]$	$[-\infty, 100]$
X_2	$\perp$	$[100, 100]$	$[-\infty, 100]$
X_3	$\perp$	$\perp$	$[-\infty, 0]$

Improving fixpoint approximation

Idea : iterating a little more may help...

Theorem

Let $(A, \sqsubseteq, \sqcup, \sqcap)$ *be a complete lattice,* f *a monotone operator on* A *and* a *a post-fixpoint of* f.

The chain $(x_n)_n$ *defined by*

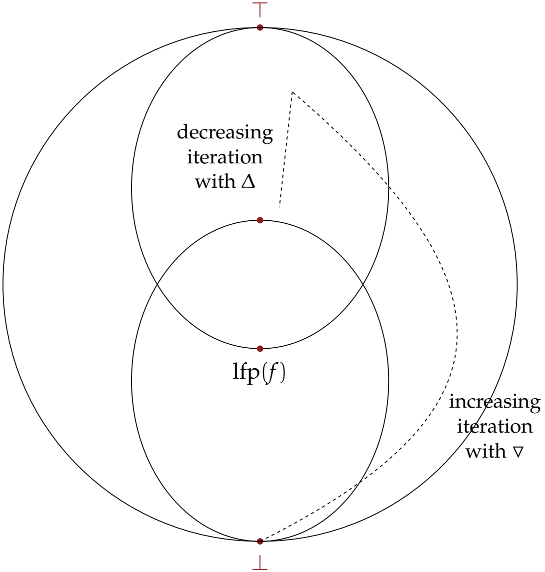
$$\begin{cases} x_0 & = & a \\ x_{k+1} & = & f(x_k) \end{cases}$$

admits for limit ($\bigsqcap\{x_n\}$) *the greatest fixpoint of* f *smaller than* a *(written* $\mathrm{gfp}_a(f)$*).*

- $\mathrm{lfp}(f) \sqsubseteq \bigsqcap\{x_n\}$.
- *Each intermediate step is a correct approximation :*

$$\forall k,\ \mathrm{lfp}(f) \sqsubseteq \mathrm{gfp}_a(f) \sqsubseteq x_k \sqsubseteq a$$

Scheme


$\top$
decreasing
iteration
with Δ
$\mathrm{lfp}(f)$
increasing
iteration
with ∇
$\bot$

Narrowing on intervals

Intuition : improve infinite bounds.

$$
\begin{array}{rcl}
[a,b]\,\Delta_{\mathrm{Itv}}\,[c,d] & = & [\text{if } a = -\infty \text{ then } c \text{ else } a \;\; ; \;\; \text{if } b = +\infty \text{ then } d \text{ else } b] \\
I \;\; \Delta_{\mathrm{Itv}} \;\; \bot & = & \bot \\
\bot \;\; \Delta_{\mathrm{Itv}} \;\; I & = & \bot
\end{array}
$$

In practice : a few standard iterations already improve a lot the result that has been obtained after widening.

- Assignments by constants and conditional guards make the decreasing iterations efficient : they *filter* the (too big) approximations computed by the widening

Example : with narrowing at each node of the cfg

$$
\begin{aligned}
X_1 &= [100,100] \sqcup_{\mathrm{Itv}} \left(X_2 -^\sharp [1,1]\right)\\
X_2 &= [1,+\infty] \sqcap_{\mathrm{Itv}} X_1\\
X_3 &= [-\infty,0] \sqcap_{\mathrm{Itv}} X_1
\end{aligned}
$$

Iteration strategy : $1 \to 2 \to 3 \to 1 \to 2 \to \cdots$

$$
\begin{array}{ll}
X_1^0 = [-\infty,100] & X_1^{n+1} = X_1^n \Delta_{\mathrm{Itv}} \left([100,100] \sqcup_{\mathrm{Itv}} \left(X_2^n -^\sharp [1,1]\right)\right)\\
X_2^0 = [-\infty,100] & X_2^{n+1} = X_2^n \Delta_{\mathrm{Itv}} \left([1,+\infty] \sqcap_{\mathrm{Itv}} X_1^{n+1}\right)\\
X_3^0 = [-\infty,0] & X_3^{n+1} = X_3^n \Delta_{\mathrm{Itv}} \left([-\infty,0] \sqcap_{\mathrm{Itv}} X_1^{n+1}\right)
\end{array}
$$

X_1	$[-\infty,100]$
X_2	$[-\infty,100]$
X_3	$[-\infty,0]$

Example : with narrowing at each node of the cfg

$$
\begin{aligned}
X_1 &= [100, 100] \sqcup_{\text{Itv}} \left(X_2 -^\sharp [1, 1]\right) \\
X_2 &= [1, +\infty] \sqcap_{\text{Itv}} X_1 \\
X_3 &= [-\infty, 0] \sqcap_{\text{Itv}} X_1
\end{aligned}
$$

Iteration strategy : $1 \to 2 \to 3 \to 1 \to 2 \to \cdots$

$$
\begin{array}{ll}
X_1^0 = [-\infty, 100] & X_1^{n+1} = X_1^n \Delta_{\text{Itv}} \left([100, 100] \sqcup_{\text{Itv}} \left(X_2^n -^\sharp [1, 1]\right)\right) \\
X_2^0 = [-\infty, 100] & X_2^{n+1} = X_2^n \Delta_{\text{Itv}} \left([1, +\infty] \sqcap_{\text{Itv}} X_1^{n+1}\right) \\
X_3^0 = [-\infty, 0] & X_3^{n+1} = X_3^n \Delta_{\text{Itv}} \left([-\infty, 0] \sqcap_{\text{Itv}} X_1^{n+1}\right)
\end{array}
$$

X_1	$[-\infty, 100]$	$[-\infty, 100]$	$[0, 100]$
X_2	$[-\infty, 100]$	$[1, 100]$	$[1, 100]$
X_3	$[-\infty, 0]$	$[-\infty, 0]$	$[0, 0]$

The particular case of an equation system

Consider a system with $f_1, \ldots, f_n$ monotone.

$$\begin{cases} x_1 & = & f_1(x_1, \ldots, x_n) \\ & \vdots & \\ x_n & = & f_n(x_1, \ldots, x_n) \end{cases}$$

Standard iteration :

$$\begin{array}{rcl} x_1^{i+1} & = & f_1(x_1^i, \ldots, x_n^i) \\ x_2^{i+1} & = & f_2(x_1^i, \ldots, x_n^i) \\ & \vdots & \\ x_n^{i+1} & = & f_n(x_1^i, \ldots, x_n^i) \end{array}$$

Standard iteration with widening :

$$\begin{array}{rcl} x_1^{i+1} & = & x_1^i \nabla f_1(x_1^i, \ldots, x_n^i) \\ x_2^{i+1} & = & x_2^i \nabla f_2(x_1^i, \ldots, x_n^i) \\ & \vdots & \\ x_n^{i+1} & = & x_n^i \nabla f_n(x_1^i, \ldots, x_n^i) \end{array}$$

The particular case of an equation system

$$\begin{cases} x_1 & = & f_1(x_1, \ldots, x_n) \\ & \vdots & \\ x_n & = & f_n(x_1, \ldots, x_n) \end{cases}$$

It is sufficient (and generally more precise) to use ∇ for a selection of index W provided that each cycle in the system has at least one point in W.

$$\forall k = 1..n,\ x_k^{i+1} = \begin{array}{ll} x_k^i \nabla f_k(x_1^i, \ldots, x_n^i) & \text{if } k \in W \\ f_k(x_1^i, \ldots, x_n^i) & \text{otherwise} \end{array}$$

Chaotic iteration : at each step, we use only one equation, without forgetting one for ever.
Beware : this time the iteration strategy may affect the precision of the obtained post-fixpoint!
Delayed widening : It is generally better to wait a few standard iterations before launching the widenings.

Plan

1. Interval analysis
2. Widening and narrowing
3. Polyhedral abstract interpretation

The need for relational program analysis

Consider the program

```
i := 1;
t := 0;
while i < 100 {
  t := t + 1;
  i := i + 2;
}
```

where `t` is inserted to argue termination of the loop.

Interval analysis with widening and narrowing will

- find precise bounds for `i` ($[1, 100]$)
- but not for `t` (only finds $[0, \infty]$).

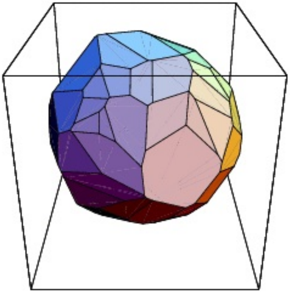
Need to know that the two variables are related : $\mathtt{i} = 2\mathtt{t} + 1$.

Polyhedral abstract interpretation

Polyhedral analysis seeks to discover invariants of **linear equality and inequality** relations (such as $x = y$ or $x \leqslant 2y + z$) among the variables of an imperative program.

A convex polyhedron can be defined

- algebraically as the set of solutions of a system of linear inequalities.
- geometrically, as a finite intersection of half-spaces.

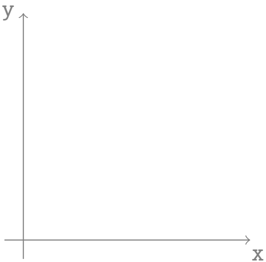


The classical reference :

Automatic discovery of linear restraints among variables of a program. P. Cousot and N. Halbwachs. POPL'78.

Polyhedral analysis

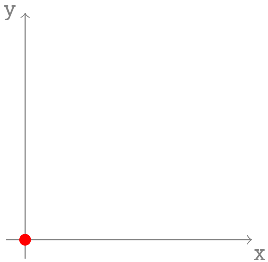
State properties are over-approximated by convex polyhedra in $\mathbb{Q}^2$.




```
x = 0; y = 0;

while (x<6) {
  if (?) {
    y = y+2;
  };
  x = x+1;
}
```

Polyhedral analysis

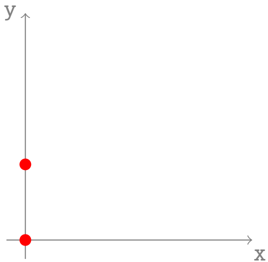
State properties are over-approximated by convex polyhedra in $\mathbb{Q}^2$.




```
x = 0; y = 0;
        {x = 0 ∧ y = 0}

while (x<6) {
  if (?) {
        {x = 0 ∧ y = 0}
    y = y+2;

  };



  x = x+1;

}
```

Polyhedral analysis

State properties are over-approximated by convex polyhedra in $\mathbb{Q}^2$.



At junction points, we over-approximates union by a convex union.

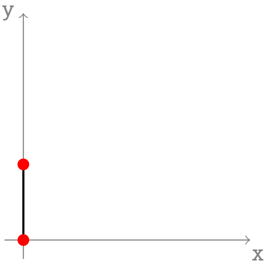
```
x = 0; y = 0;
```
$\{x = 0 \wedge y = 0\}$

```
while (x<6) {
  if (?) {
```
$\{x = 0 \wedge y = 0\}$
```
    y = y+2;
```
$\{x = 0 \wedge y = 2\}$
```
  };
```
$\{x = 0 \wedge y = 0\} \uplus \{x = 0 \wedge y = 2\}$

```
  x = x+1;

}
```

Polyhedral analysis

State properties are over-approximated by convex polyhedra in $\mathbb{Q}^2$.

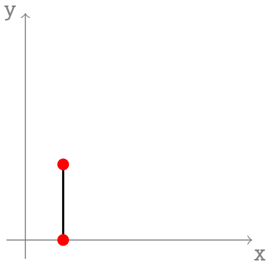


At junction points, we over-approximates union by a convex union.

```
x = 0; y = 0;
        {x = 0 ∧ y = 0}

while (x<6) {
  if (?) {
        {x = 0 ∧ y = 0}
    y = y+2;
        {x = 0 ∧ y = 2}
  };
        {x = 0 ∧ 0 ⩽ y ⩽ 2}

  x = x+1;

}
```

Polyhedral analysis

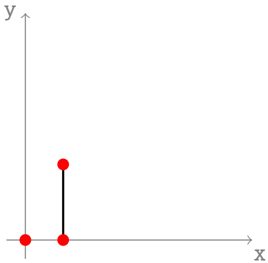
State properties are over-approximated by convex polyhedra in $\mathbb{Q}^2$.




```
x = 0; y = 0;
        {x = 0 ∧ y = 0}

while (x<6) {
  if (?) {
        {x = 0 ∧ y = 0}
    y = y+2;
        {x = 0 ∧ y = 2}
  };
        {x = 0 ∧ 0 ⩽ y ⩽ 2}

  x = x+1;
        {x = 1 ∧ 0 ⩽ y ⩽ 2}
}
```

Polyhedral analysis

State properties are over-approximated by convex polyhedra in $\mathbb{Q}^2$.



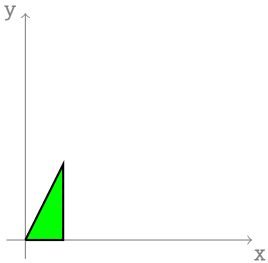

```
x = 0; y = 0;
```
$\{x = 0 \wedge y = 0\} \uplus \{x = 1 \wedge 0 \leqslant y \leqslant 2\}$

```
while (x<6) {
  if (?) {
```
$\{x = 0 \wedge y = 0\}$
```
    y = y+2;
```
$\{x = 0 \wedge y = 2\}$
```
  };
```
$\{x = 0 \wedge 0 \leqslant y \leqslant 2\}$

```
  x = x+1;
```
$\{x = 1 \wedge 0 \leqslant y \leqslant 2\}$
```
}
```

Polyhedral analysis

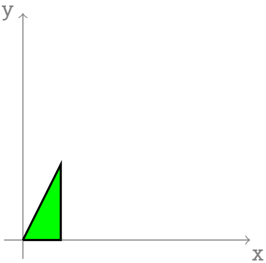
State properties are over-approximated by convex polyhedra in $\mathbb{Q}^2$.




```
x = 0; y = 0;
        {x ⩽ 1 ∧ 0 ⩽ y ⩽ 2x}

while (x<6) {
  if (?) {
        {x = 0 ∧ y = 0}
    y = y+2;
        {x = 0 ∧ y = 2}
  };
        {x = 0 ∧ 0 ⩽ y ⩽ 2}

  x = x+1;
        {x = 1 ∧ 0 ⩽ y ⩽ 2}
}
```

Polyhedral analysis

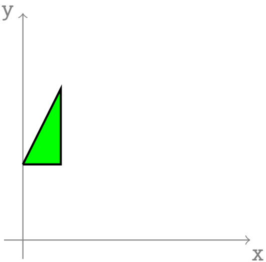
State properties are over-approximated by convex polyhedra in $\mathbb{Q}^2$.




```
x = 0; y = 0;
      {x ⩽ 1 ∧ 0 ⩽ y ⩽ 2x}

while (x<6) {
  if (?) {
      {x ⩽ 1 ∧ 0 ⩽ y ⩽ 2x}
    y = y+2;
      {x = 0 ∧ y = 2}
  };
      {x = 0 ∧ 0 ⩽ y ⩽ 2}

  x = x+1;
      {x = 1 ∧ 0 ⩽ y ⩽ 2}
}
```

Polyhedral analysis

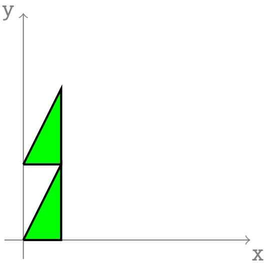
State properties are over-approximated by convex polyhedra in $\mathbb{Q}^2$.




```
x = 0; y = 0;
        {x ⩽ 1 ∧ 0 ⩽ y ⩽ 2x}

while (x<6) {
  if (?) {
        {x ⩽ 1 ∧ 0 ⩽ y ⩽ 2x}
    y = y+2;
        {x ⩽ 1 ∧ 2 ⩽ y ⩽ 2x + 2}
  };
        {x = 0 ∧ 0 ⩽ y ⩽ 2}

  x = x+1;
        {x = 1 ∧ 0 ⩽ y ⩽ 2}
}
```

Polyhedral analysis

State properties are over-approximated by convex polyhedra in $\mathbb{Q}^2$.




```
x = 0; y = 0;
```

$\{x \leqslant 1 \wedge 0 \leqslant y \leqslant 2x\}$

```
while (x<6) {
  if (?) {
```

$\{x \leqslant 1 \wedge 0 \leqslant y \leqslant 2x\}$

```
    y = y+2;
```

$\{x \leqslant 1 \wedge 2 \leqslant y \leqslant 2x+2\}$

```
  };
```

$\{x \leqslant 1 \wedge 0 \leqslant y \leqslant 2x\}$
$\uplus\{x \leqslant 1 \wedge 2 \leqslant y \leqslant 2x+2\}$

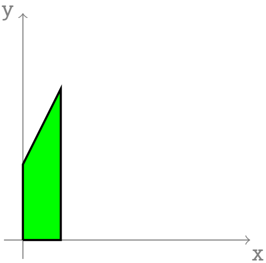
```
  x = x+1;
```

$\{x = 1 \wedge 0 \leqslant y \leqslant 2\}$

```
}
```

Polyhedral analysis

State properties are over-approximated by convex polyhedra in $\mathbb{Q}^2$.



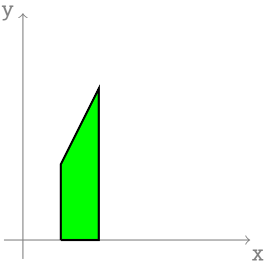

```
x = 0; y = 0;
```
$\{x \leqslant 1 \wedge 0 \leqslant y \leqslant 2x\}$

```
while (x<6) {
  if (?) {
```
$\{x \leqslant 1 \wedge 0 \leqslant y \leqslant 2x\}$
```
    y = y+2;
```
$\{x \leqslant 1 \wedge 2 \leqslant y \leqslant 2x + 2\}$
```
  };
```
$\{0 \leqslant x \leqslant 1 \wedge 0 \leqslant y \leqslant 2x + 2\}$

```
  x = x+1;
```
$\{x = 1 \wedge 0 \leqslant y \leqslant 2\}$
```
}
```

Polyhedral analysis

State properties are over-approximated by convex polyhedra in $\mathbb{Q}^2$.




```
x = 0; y = 0;
```
$\{x \leqslant 1 \wedge 0 \leqslant y \leqslant 2x\}$

```
while (x<6) {
  if (?) {
```
$\{x \leqslant 1 \wedge 0 \leqslant y \leqslant 2x\}$

```
    y = y+2;
```
$\{x \leqslant 1 \wedge 2 \leqslant y \leqslant 2x+2\}$

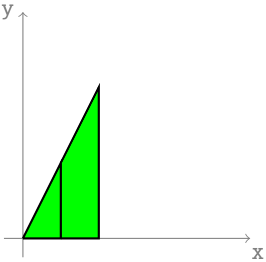
```
  };
```
$\{0 \leqslant x \leqslant 1 \wedge 0 \leqslant y \leqslant 2x+2\}$

```
  x = x+1;
```
$\{1 \leqslant x \leqslant 2 \wedge 0 \leqslant y \leqslant 2x\}$

```
}
```

Polyhedral analysis

State properties are over-approximated by convex polyhedra in $\mathbb{Q}^2$.




```
x = 0; y = 0;
```
$\{x \leqslant 1 \wedge 0 \leqslant y \leqslant 2x\}$

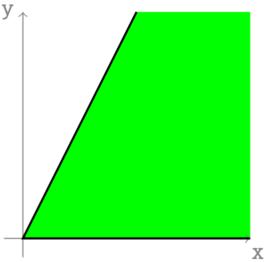
$\nabla \{x \leqslant 2 \wedge 0 \leqslant y \leqslant 2x\}$

```
while (x<6) {
  if (?) {
```
$\{x \leqslant 1 \wedge 0 \leqslant y \leqslant 2x\}$
```
    y = y+2;
```
$\{x \leqslant 1 \wedge 2 \leqslant y \leqslant 2x + 2\}$
```
  };
```
$\{0 \leqslant x \leqslant 1 \wedge 0 \leqslant y \leqslant 2x + 2\}$
```
  x = x+1;
```
$\{1 \leqslant x \leqslant 2 \wedge 0 \leqslant y \leqslant 2x\}$
```
}
```

At loop headers, we use heuristics (widening) to ensure finite convergence.

Polyhedral analysis

State properties are over-approximated by convex polyhedra in $\mathbb{Q}^2$.




```
x = 0; y = 0;
      {0 ⩽ y ⩽ 2x}

while (x<6) {
  if (?) {
      {x ⩽ 1 ∧ 0 ⩽ y ⩽ 2x}
    y = y+2;
      {x ⩽ 1 ∧ 2 ⩽ y ⩽ 2x + 2}
  };
      {0 ⩽ x ⩽ 1 ∧ 0 ⩽ y ⩽ 2x + 2}

  x = x+1;
      {1 ⩽ x ⩽ 2 ∧ 0 ⩽ y ⩽ 2x}
}
```

At loop headers, we use heuristics (widening) to ensure finite convergence.

Polyhedral analysis

State properties are over-approximated by convex polyhedra in $\mathbb{Q}^2$.

By propagation we obtain a post-fixpoint

```
x = 0; y = 0;
        {0 ⩽ y ⩽ 2x}

while (x<6) {
  if (?) {
        {0 ⩽ y ⩽ 2x ∧ x ⩽ 5}
    y = y+2;
        {2 ⩽ y ⩽ 2x + 2 ∧ x ⩽ 5}
  };
        {0 ⩽ y ⩽ 2x + 2 ∧ 0 ⩽ x ⩽ 5}

  x = x+1;
        {0 ⩽ y ⩽ 2x ∧ 1 ⩽ x ⩽ 6}
}
        {0 ⩽ y ⩽ 2x ∧ 6 ⩽ x}
```

Polyhedral analysis

State properties are over-approximated by convex polyhedra in $\mathbb{Q}^2$.

By propagation we obtain a post-fixpoint which is enhanced by downward iteration.

```
x = 0; y = 0;
```

$\{0 \leqslant y \leqslant 2x \wedge x \leqslant 6\}$

```
while (x<6) {
  if (?) {
```

$\{0 \leqslant y \leqslant 2x \wedge x \leqslant 5\}$

```
    y = y+2;
```

$\{2 \leqslant y \leqslant 2x + 2 \wedge x \leqslant 5\}$

```
  };
```

$\{0 \leqslant y \leqslant 2x + 2 \wedge 0 \leqslant x \leqslant 5\}$

```
  x = x+1;
```

$\{0 \leqslant y \leqslant 2x \wedge 1 \leqslant x \leqslant 6\}$

```
}
```

$\{0 \leqslant y \leqslant 2x \wedge 6 = x\}$

Polyhedral analysis

A more complex example.

The analysis accepts to replace some constants by parameters.

```
x = 0; y = A;
```
$\{A \leqslant \mathrm{y} \leqslant 2\mathrm{x} + A \wedge x \leqslant N\}$

```
while (x<N) {
  if (?) {
```
$\{A \leqslant \mathrm{y} \leqslant 2\mathrm{x} + A \;\wedge\; x \leqslant N - 1\}$
```
    y = y+2;
```
$\{A + 2 \leqslant \mathrm{y} \leqslant 2\mathrm{x} + A + 2 \;\wedge\; x \leqslant N - 1\}$
```
  };
```
$\{A \leqslant \mathrm{y} \leqslant 2\mathrm{x} + A + 2 \;\wedge\; 0 \leqslant x \leqslant N - 1\}$

```
  x = x+1;
```
$\{A \leqslant \mathrm{y} \leqslant 2\mathrm{x} + A \;\wedge\; 1 \leqslant x \leqslant N\}$
```
}
```
$\{A \leqslant \mathrm{y} \leqslant 2\mathrm{x} + A \;\wedge\; N = x\}$

The four polyhedra operations

- $\uplus \in \mathbb{P}_n \times \mathbb{P}_n \to \mathbb{P}_n$: convex union
  - over-approximates the concrete union at junction points
- $\cap \in \mathbb{P}_n \times \mathbb{P}_n \to \mathbb{P}_n$: intersection
  - over-approximates the concrete intersection after a conditional instruction
- $[\![\mathtt{x} := e]\!] \in \mathbb{P}_n \to \mathbb{P}_n$: affine transformation
  - over-approximates the assignment of a variable by a linear expression
- $\nabla \in \mathbb{P}_n \times \mathbb{P}_n \to \mathbb{P}_n$: widening
  - ensures (and accelerates) convergence of (post-)fixpoint iteration
  - includes heuristics to infer loop invariants

```
x = 0; y = 0;
```

$$P_0 = [\![\mathtt{y} := \mathtt{0}]\!]\,[\![\mathtt{x} := \mathtt{0}]\!]\,(\mathbb{Q}^2)\ \nabla\ P_4$$

```
while (x<6) {
  if (?) {
```

$$P_1 = P_0 \cap \{x < 6\}$$

```
    y = y+2;
```

$$P_2 = [\![\mathtt{y} := \mathtt{y} + \mathtt{2}]\!]\,(P_1)$$

```
  };
```

$$P_3 = P_1 \uplus P_2$$

```
  x = x+1;
```

$$P_4 = [\![\mathtt{x} := \mathtt{x} + \mathtt{1}]\!]\,(P_3)$$

```
}
```

$$P_5 = P_0 \cap \{x \geqslant 6\}$$

More about abstraction of assignments

Distinguish between two types of assignments `x := exp`

- invertible assignments, where `x` appears in `exp`
  Example : `x = 2x + y`
- non-invertible assignments, when the new value of `x` does not depend on its old value.
  Example : `x = y + 42`

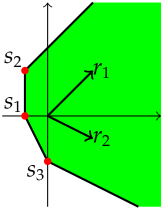
Invertible assignments can be abstracted precisely in two steps :

1. express the old value of `x` as an expression involving the new value (written `x'`)
   Example : for `x := x + 1`, we have `x'-1 = x`
2. in the constraints, replace all occurrences of `x` by the inverted expression for `x`.
   Example : $y \geqslant 2x$ becomes $y \geqslant 2(x-1)$.

For non-invertible assignments : all information about old `x` is lost, so we remove all constraints involving `x`, and add $x = exp$.

Library for manipulating polyhedra

- Parma Polyhedra Library [1] (PPL), NewPolka : complex C/C++ libraries
- They rely on the Double Description Method
  - polyhedra are managed using two representations in parallel



    - by set of inequalities

$$P = \left\{ (x,y) \in \mathbb{Q}^2 \;\middle|\; \begin{array}{l} x \geqslant -1 \\ x - y \geqslant -3 \\ 2x + y \geqslant -2 \\ x + 2y \geqslant -4 \end{array} \right\}$$

    - by set of generators

$$P = \left\{ \lambda_1 s_1 + \lambda_2 s_2 + \lambda_3 s_3 + \mu_1 r_1 + \mu_2 r_2 \in \mathbb{Q}^2 \;\middle|\; \begin{array}{l} \lambda_1, \lambda_2, \lambda_3, \mu_1, \mu_2 \in \mathbb{R}^+ \\ \lambda_1 + \lambda_2 + \lambda_3 = 1 \end{array} \right\}$$

  - operations efficiency strongly depends on the chosen representations, so they keep both

1. Previous tutorial on polyhedra partially comes from http://www.cs.unipr.it/ppl/

Other relational domains

Polyhedral analysis uses all linear relations ($\bigwedge_j \sum_i a_{ij}x_i \geqslant c_j$).
It is precise but has exponential complexity.

Other examples of (weakly) relational analyses

- linear equalities ($x = \sum_i a_i x_i$)
- zones and octagons.

Octagons : restrict constraints to be of form $\pm X_1 \pm X_2 \leqslant C$.

- At most two variables are related in one constraint.
- Only allowed coefficients are -1,1.

Efficient polynomial-time operations and good precision.

Example (Miné p. 131) : least upper bound of two boxes with intervals and with octagons :