

Numerical Abstract Domains

MPRI 2–6: Abstract Interpretation,
application to verification and static analysis

Antoine Miné

CNRS, École normale supérieure

3 February 2012

Outline

- Some **applications** of numerical domains
- Generalities, notations (reminders)
- Presentation of a few **numerical abstract domains**
 - non-relational domains: **intervals**, **congruences**
 - **linear equality** domains
 - **polyhedra** domain (double description)
 - weakly relational domains: **zones**, **octagons**
- Handling **non-linear expressions**
- **Floating-point** abstractions
- Bibliography ([**exemple**])

Shortcomings of non-relational domains

Accumulating loss of precision

Non-relation domains cannot represent variable **relationships**.

Rate limiter

```

Y:=0; while • true do
  X:=[-128,128]; D:=[0,16];
  S:=Y; Y:=X; R:=X-S;
  if R<=-D then Y:=S-D fi;
  if R>=D then Y:=S+D fi
done
  
```

X: input signal
 Y: output signal
 S: last output
 R: delta Y-S
 D: max. allowed for |R|

Iterations in the interval domain (without widening):

$x^{\#0}$	$x^{\#1}$	$x^{\#2}$...	$x^{\#n}$
$Y = 0$	$ Y \leq 144$	$ Y \leq 160$...	$ Y \leq 128 + 16n$

In fact, $Y \in [-128, 128]$ always holds.

To prove that, e.g. $Y \geq -128$, we must be able to:

- **represent** the properties $R = X - S$ and $R \leq -D$,
- **combine** them to deduce $S - X \geq D$, and then $Y = S - D \geq X$.

The need for relational loop invariants

To prove some invariant after the **end of a loop**, we often need to find a **loop invariant** of a **more complex form**.

relational loop invariant

```
X:=0; I:=1;
while • I<5000 do
  if ? then X:=X+1 else X:=X-1 fi;
  I:=I+1
done ◆
```

A non-relational analysis finds at ◆ that $I = 5000$ and $X \in \mathbb{Z}$.

The best invariant is: $(I = 5000) \wedge (X \in [-4999, 4999]) \wedge (X \equiv 0 [2])$.

To find this **non-relational** invariant, we must find a **relational** loop invariant at •: $(-I < X < I) \wedge (X + I \equiv 1 [2]) \wedge (I \in [1, 5000])$, and apply the loop exit condition $C^\sharp \llbracket I \geq 5000 \rrbracket$.

Reminders

Syntax

Fixed finite set of variables \mathbb{V} ,
with value in \mathbb{I} , $\mathbb{I} \in \{\mathbb{Z}, \mathbb{Q}, \mathbb{R}, \mathbb{M}, \mathbb{F}\}$

arithmetic expressions:

exp	::=	V	variable $V \in \mathbb{V}$
		$-exp$	negation
		$exp \diamond exp$	binary operation: $\diamond \in \{+, -, \times, /\}$
		$[c, c']$	constant range, $c, c' \in \mathbb{I} \cup \{\pm\infty\}$
			c is a shorthand for $[c, c]$

commands:

com	::=	$V := exp$	assignment into $V \in \mathbb{V}$
		$exp \bowtie 0$	test, $\bowtie \in \{=, <, >, \leq, \geq, <>\}$

Concrete semantics

Semantics of expressions: $E[e]: (\mathbb{V} \rightarrow \mathbb{I}) \rightarrow \mathcal{P}(\mathbb{I})$

$$\begin{aligned}
 E[[c, c']] \rho &\stackrel{\text{def}}{=} \{x \in \mathbb{I} \mid c \leq x \leq c'\} \\
 E[[\mathbf{v}]] \rho &\stackrel{\text{def}}{=} \{\rho(\mathbf{v})\} \\
 E[[-e]] \rho &\stackrel{\text{def}}{=} \{-v \mid v \in E[e] \rho\} \\
 E[[e_1 + e_2]] \rho &\stackrel{\text{def}}{=} \{v_1 + v_2 \mid v_1 \in E[[e_1]] \rho, v_2 \in E[[e_2]] \rho\} \\
 &\dots
 \end{aligned}$$

Forward commands: $C[c]: \mathcal{P}(\mathbb{V} \rightarrow \mathbb{I}) \rightarrow \mathcal{P}(\mathbb{V} \rightarrow \mathbb{I})$

$$\begin{aligned}
 C[[\mathbf{v} := e]] \mathcal{X} &\stackrel{\text{def}}{=} \{\rho[\mathbf{v} \mapsto v] \mid \rho \in \mathcal{X}, v \in E[e] \rho\} \\
 C[[e \bowtie 0]] \mathcal{X} &\stackrel{\text{def}}{=} \{\rho \mid \rho \in \mathcal{X}, \exists v \in E[e] \rho, v \bowtie 0\}
 \end{aligned}$$

Backward commands: $C[\overleftarrow{c}]: \mathcal{P}(\mathbb{V} \rightarrow \mathbb{I}) \rightarrow \mathcal{P}(\mathbb{V} \rightarrow \mathbb{I})$

$$\begin{aligned}
 C[\overleftarrow{[\mathbf{v} := e]}] \mathcal{X} &\stackrel{\text{def}}{=} \{\rho \mid \exists v \in E[e] \rho, \rho[\mathbf{v} \mapsto v] \in \mathcal{X}\} \\
 C[\overleftarrow{[e \bowtie 0]}] \mathcal{X} &\stackrel{\text{def}}{=} C[[e \bowtie 0]] \mathcal{X}
 \end{aligned}$$

Abstract domain

- Abstract elements:

- $\mathcal{D}^\#$, a set of computer-representable elements
- $\gamma : \mathcal{D}^\# \rightarrow \mathcal{D}$ concretization
- $\subseteq^\#$, an approximation order: $\mathcal{X}^\# \subseteq^\# \mathcal{Y}^\# \implies \gamma(\mathcal{X}^\#) \subseteq \gamma(\mathcal{Y}^\#)$

- Abstract operators:

- $C^\#[c]$ such that $C[c] \gamma(\mathcal{X}^\#) \subseteq \gamma(C^\#[c] \mathcal{X}^\#)$
- $U^\#$ such that $\gamma(\mathcal{X}^\#) \cup \gamma(\mathcal{Y}^\#) \subseteq \gamma(\mathcal{X}^\# \cup^\# \mathcal{Y}^\#)$
- $\cap^\#$ such that $\gamma(\mathcal{X}^\#) \cap \gamma(\mathcal{Y}^\#) \subseteq \gamma(\mathcal{X}^\# \cap^\# \mathcal{Y}^\#)$
- $C^\#[\overleftarrow{c}]$ such that $\gamma(\mathcal{X}^\#) \cap C[\overleftarrow{c}] \gamma(\mathcal{R}^\#) \subseteq \gamma(C^\#[\overleftarrow{c}] (\mathcal{X}^\#, \mathcal{R}^\#))$

- Fixpoint extrapolation:

- $\nabla : (\mathcal{D}^\# \times \mathcal{D}^\#) \rightarrow \mathcal{D}^\#$ widening
- $\Delta : (\mathcal{D}^\# \times \mathcal{D}^\#) \rightarrow \mathcal{D}^\#$ narrowing

Linear equality domains

The affine equality domain

Here $\mathbb{I} \in \{\mathbb{Q}, \mathbb{R}\}$.

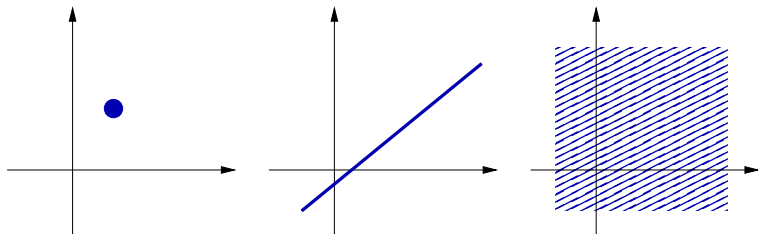
We look for invariants of the form:

$$\bigwedge_j (\sum_{i=1}^n \alpha_{ij} \mathbf{v}_i = \beta_j), \alpha_{ij}, \beta_j \in \mathbb{I}$$

where all the α_{ij} and β_j are inferred automatically.

We use a domain of affine spaces proposed by [Karr76]:

$$\mathcal{D}^\# \stackrel{\text{def}}{=} \{ \text{affine subspaces of } \mathbb{V} \rightarrow \mathbb{I} \}$$



Affine equality representation

Machine representation: an affine subspace is represented as

- either the constant \perp^\sharp ,
- or a pair $\langle \mathbf{M}, \vec{C} \rangle$ where
 - $\mathbf{M} \in \mathbb{I}^{m \times n}$ is a $m \times n$ matrix, $n = |\mathbb{V}|$ and $m \leq n$,
 - $\vec{C} \in \mathbb{I}^m$ is a row-vector with m rows.

$\langle \mathbf{M}, \vec{C} \rangle$ represents an equation system, with solutions:

$$\gamma(\langle \mathbf{M}, \vec{C} \rangle) \stackrel{\text{def}}{=} \{ \vec{V} \in \mathbb{I}^n \mid \mathbf{M} \times \vec{V} = \vec{C} \}$$

\mathbf{M} should be in **row echelon form**:

- $\forall i \leq m, \exists k_i$ such that $M_{ik_i} = 1$
and $\forall c < k_i, M_{ic} = 0, \forall l \neq i, M_{lk_i} = 0$,
- if $i < i'$ then $k_i < k_{i'}$.

Remarks:

- the representation is unique,
- as $m \leq n = |\mathbb{V}|$, the memory cost is in $\mathcal{O}(n^2)$ at worst,
- \top^\sharp is represented as the empty equation system: $m = 0$.

Normalisation and emptiness testing

Let $\mathbf{M} \times \vec{V} = \vec{C}$ be a system, not necessarily in normal form.

The **Gaussian reduction** tells in $\mathcal{O}(n^3)$ time:

- whether the system is satisfiable, and in that case
- gives an equivalent system in normal form.

i.e. returns an element in \mathcal{D}^\sharp .

Example:

$$\left\{ \begin{array}{rclcl} 2X & + & Y & + & Z & = & 19 \\ 2X & + & Y & - & Z & = & 9 \\ & & & & 3Z & = & 15 \end{array} \right.$$

\Downarrow

$$\left\{ \begin{array}{rclcl} X & + & 0.5Y & & & = & 7 \\ & & & & Z & = & 5 \end{array} \right.$$

Normalisation and emptiness testing (cont.)

Gaussian reduction algorithm: $Gauss(\langle \mathbf{M}, \vec{C} \rangle)$

```

r:=0 (rank r)
for c from 1 to n (column c)
  if  $\exists \ell > r, M_{\ell c} \neq 0$  (pivot  $\ell$ )
    r := r + 1
    swap  $\langle \vec{M}_\ell, C_\ell \rangle$  and  $\langle \vec{M}_r, C_r \rangle$ 
    divide  $\langle \vec{M}_r, C_r \rangle$  by  $M_{rc}$ 
    for j from 1 to n,  $j \neq r$ 
      replace  $\langle \vec{M}_j, C_j \rangle$  with  $\langle \vec{M}_j, C_j \rangle - M_{jc} \langle \vec{M}_r, C_r \rangle$ 
  if  $\exists \ell, \langle \vec{M}_\ell, C_\ell \rangle = \langle 0, \dots, 0, c \rangle, c \neq 0$ 
    then return unsatisfiable

remove all rows  $\langle \vec{M}_\ell, C_\ell \rangle$  that equal  $\langle 0, \dots, 0, 0 \rangle$ 

```

Affine equality operators

Applications

If $\mathcal{X}^\#, \mathcal{Y}^\# \neq \perp^\#$, we define:

$$\mathcal{X}^\# \cap^\# \mathcal{Y}^\# \stackrel{\text{def}}{=} \text{Gauss} \left(\left\langle \left[\begin{array}{c} \mathbf{M}_{\mathcal{X}^\#} \\ \mathbf{M}_{\mathcal{Y}^\#} \end{array} \right], \left[\begin{array}{c} \vec{c}_{\mathcal{X}^\#} \\ \vec{c}_{\mathcal{Y}^\#} \end{array} \right] \right\rangle \right)$$

$$\mathcal{X}^\# =^\# \mathcal{Y}^\# \stackrel{\text{def}}{\iff} \mathbf{M}_{\mathcal{X}^\#} = \mathbf{M}_{\mathcal{Y}^\#} \quad \text{and} \quad \vec{c}_{\mathcal{X}^\#} = \vec{c}_{\mathcal{Y}^\#}$$

$$\mathcal{X}^\# \subseteq^\# \mathcal{Y}^\# \stackrel{\text{def}}{\iff} \mathcal{X}^\# \cap^\# \mathcal{Y}^\# =^\# \mathcal{X}^\#$$

$$\mathbf{C}^\#[\sum_j \alpha_j \mathbf{v}_j - \beta = 0] \mathcal{X}^\# \stackrel{\text{def}}{=} \text{Gauss} \left(\left\langle \left[\begin{array}{c} \mathbf{M}_{\mathcal{X}^\#} \\ \alpha_1 \cdots \alpha_n \end{array} \right], \left[\begin{array}{c} \vec{c}_{\mathcal{X}^\#} \\ \beta \end{array} \right] \right\rangle \right)$$

$$\mathbf{C}^\#[e \bowtie 0] \mathcal{X}^\# \stackrel{\text{def}}{=} \mathcal{X}^\# \quad \text{for other tests}$$

Remark:

$\subseteq^\#, =^\#, \cap^\#, =^\#$ and $\mathbf{C}^\#[\sum_j \alpha_j \mathbf{v}_j - \beta = 0]$ are **exact**:

$$\mathcal{X}^\# \subseteq^\# \mathcal{Y}^\# \iff \gamma(\mathcal{X}^\#) \subseteq \gamma(\mathcal{Y}^\#), \quad \gamma(\mathcal{X}^\# \cap^\# \mathcal{Y}^\#) = \gamma(\mathcal{X}^\#) \cap \gamma(\mathcal{Y}^\#), \dots$$

Generator representation

Generator representation

An affine subspace can also be represented as a set of **vector generators** $\vec{G}_1, \dots, \vec{G}_m$ and an **origin point** \vec{O} , denoted as $[\mathbf{G}, \vec{O}]$.

$$\gamma([\mathbf{G}, \vec{O}]) \stackrel{\text{def}}{=} \{ \mathbf{G} \times \vec{\lambda} + \vec{O} \mid \vec{\lambda} \in \mathbb{I}^m \} \quad (\mathbf{G} \in \mathbb{I}^{n \times m}, \vec{O} \in \mathbb{I}^n)$$

We can **switch** between a generator and a constraint representation:

- From generators to constraints: $\langle \mathbf{M}, \vec{C} \rangle = \text{Cons}([\mathbf{G}, \vec{O}])$

Write the system $\vec{V} = \mathbf{G} \times \vec{\lambda} + \vec{O}$ with variables $\vec{V}, \vec{\lambda}$.

Solve it in $\vec{\lambda}$ (by row operations).

Keep the constraints involving only \vec{V} .

$$\text{e.g. } \begin{cases} X = \lambda + 2 \\ Y = 2\lambda + \mu + 3 \\ Z = \mu \end{cases} \implies \begin{cases} X - 2 = \lambda \\ -2X + Y + 1 = \mu \\ 2X - Y + Z - 1 = 0 \end{cases}$$

The result is: $2X - Y + Z = 1$.

Generator representation (cont.)

- From constraints to generators: $[\mathbf{G}, \vec{O}] \stackrel{\text{def}}{=} \text{Gen}(\langle \mathbf{M}, \vec{C} \rangle)$

Assume $\langle \mathbf{M}, \vec{C} \rangle$ is normalized.

For each non-leading variable V , assign a distinct λ_V ,
solve leading variables in terms of non-leading ones.

$$\text{e.g. } \begin{cases} X + 0.5Y = 7 \\ \quad \quad Z = 5 \end{cases} \implies \begin{bmatrix} -0.5 \\ 1 \\ 0 \end{bmatrix} \lambda_Y + \begin{bmatrix} 7 \\ 0 \\ 5 \end{bmatrix}$$

Affine equality operators (cont.)

Applications

Given $\mathcal{X}^\#, \mathcal{Y}^\# \neq \perp^\#$, we define:

$$\mathcal{X}^\# \cup^\# \mathcal{Y}^\# \stackrel{\text{def}}{=} \text{Cons} \left(\text{Gauss} \left(\left[\left[\mathbf{G}_{\mathcal{X}^\#} \quad \mathbf{G}_{\mathcal{Y}^\#} \quad (\vec{0}_{\mathcal{Y}^\#} - \vec{0}_{\mathcal{X}^\#}) \right], \vec{0}_{\mathcal{X}^\#} \right] \right) \right)$$

$$\mathbf{C}^\# \llbracket \mathbf{V}_j :=] - \infty, +\infty \rrbracket \mathcal{X}^\# \stackrel{\text{def}}{=} \text{Cons} \left(\text{Gauss} \left(\left[\left[\mathbf{G}_{\mathcal{X}^\#} \quad \vec{x}_j \right], \vec{0}_{\mathcal{X}^\#} \right] \right) \right)$$

$$\mathbf{C}^\# \llbracket \mathbf{V}_j := \sum_i \alpha_i \mathbf{V}_i + \beta \rrbracket \mathcal{X}^\# \stackrel{\text{def}}{=}$$

if $\alpha_j = 0$, $(\mathbf{C}^\# \llbracket \sum_i \alpha_i \mathbf{V}_i - \mathbf{V}_j + \beta = 0 \rrbracket \circ \mathbf{C}^\# \llbracket \mathbf{V}_j :=] - \infty, +\infty \rrbracket) \mathcal{X}^\#$

if $\alpha_j \neq 0$, $\mathcal{X}^\#$ where \mathbf{V}_j is replaced with $(\mathbf{V}_j - \sum_{i \neq j} \alpha_i \mathbf{V}_i - \beta) / \alpha_j$

$$\mathbf{C}^\# \llbracket \mathbf{V}_j := \mathbf{e} \rrbracket \mathcal{X}^\# \stackrel{\text{def}}{=} \mathbf{C}^\# \llbracket \mathbf{V}_j :=] - \infty, +\infty \rrbracket \mathcal{X}^\# \text{ for other assignments}$$

Remarks:

- $\cup^\#$ is **optimal**, but not exact.
- $\mathbf{C}^\# \llbracket \mathbf{V}_j := \sum_i \alpha_i \mathbf{V}_i + \beta \rrbracket$ and $\mathbf{C}^\# \llbracket \mathbf{V}_j :=] - \infty, +\infty \rrbracket$ are **exact**.

Affine equality operators (cont.)

Backward assignments:

$$C^\# \left[\overleftarrow{V_j :=} -\infty, +\infty \right] (\mathcal{X}^\#, \mathcal{R}^\#) \stackrel{\text{def}}{=} \mathcal{X}^\# \cap^\# (C^\# [V_j :=] -\infty, +\infty [] \mathcal{R}^\#)$$

$$C^\# \left[\overleftarrow{V_j := \sum_i \alpha_i v_i + \beta} \right] (\mathcal{X}^\#, \mathcal{R}^\#) \stackrel{\text{def}}{=} \mathcal{X}^\# \cap^\# (\mathcal{R}^\# \text{ where } V_j \text{ is replaced with } (\sum_i \alpha_i v_i + \beta))$$

$$C^\# \left[\overleftarrow{V_j := e} \right] (\mathcal{X}^\#, \mathcal{R}^\#) \stackrel{\text{def}}{=} C^\# \left[\overleftarrow{V_j :=} -\infty, +\infty [] (\mathcal{X}^\#, \mathcal{R}^\#) \right]$$

for other assignments

Remarks:

- $C^\# \left[\overleftarrow{V_j := \sum_i \alpha_i v_i + \beta} \right]$ and $C^\# \left[\overleftarrow{V_j :=} -\infty, +\infty [] \right]$ are **exact**
- a backward assignment can be seen as a substitution wrt. constraints (similar to weakest preconditions [Dijk75])

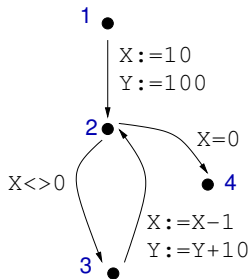
Analysis example

No infinite increasing chain: we can iterate without widening.

Forward analysis example:

```

1 X:=10; Y:=100;
  while 2 X<>0 do 3
    X:=X-1;
    Y:=Y+10
  done 4
  
```



ℓ	$\mathcal{x}_\ell^{\#0}$	$\mathcal{x}_\ell^{\#1}$	$\mathcal{x}_\ell^{\#2}$	$\mathcal{x}_\ell^{\#3}$	$\mathcal{x}_\ell^{\#4}$
1	$\top^\#$	$\top^\#$	$\top^\#$	$\top^\#$	$\top^\#$
2	$\perp^\#$	(10, 100)	(10, 100)	$10X + Y = 200$	$10X + Y = 200$
3	$\perp^\#$	$\perp^\#$	(10, 100)	(10, 100)	$10X + Y = 200$
4	$\perp^\#$	$\perp^\#$	$\perp^\#$	$\perp^\#$	(0, 200)

Note in particular:

$$\mathcal{x}_2^{\#3} = \{(10, 100)\} \cup \{(9, 110)\} = \{(X, Y) \mid 10X + Y = 200\}$$

Constraint-only equality domain

In fact [Karr76] does not use the generator representation.
 (rationale: few constraints but many generators in practice)

We need to redefine two operators: forgetting and union.

- $C^\#[\mathbb{V}_j :=] - \infty, +\infty[[]]$

Pick the row $\langle \vec{M}_i, C_i \rangle$ such that $M_{ij} \neq 0$ and i maximal.

Use it to eliminate all non-0 occurrences of \mathbb{V}_j in \mathbf{M} .

Then remove the row $\langle \vec{M}_i, C_i \rangle$.

e.g. forgetting Z:
$$\begin{cases} X + Z = 10 \\ Y + Z = 7 \end{cases} \implies \begin{cases} X - Y = 3 \end{cases}$$

The operator is exact.

Constraint-only equality domain (cont.)

- $\langle \mathbf{M}, \vec{C} \rangle \cup^\# \langle \mathbf{N}, \vec{D} \rangle$

Idea: unify columns 1 to n in $\langle \mathbf{M}, \vec{C} \rangle$ and $\langle \mathbf{N}, \vec{D} \rangle$ using row operations.

e.g. unify columns ${}^t(\vec{0} \ 1 \ \vec{0})$ and ${}^t(\vec{\beta} \ 0 \ \vec{0})$.

$$\left(\begin{array}{c} \mathbf{R} \ \vec{0} \ \mathbf{M}_1 \\ \vec{0} \ 1 \ \vec{M}_2 \\ \mathbf{0} \ \vec{0} \ \mathbf{M}_3 \end{array} \right), \left(\begin{array}{c} \mathbf{R} \ \vec{\beta} \ \mathbf{N}_1 \\ \vec{0} \ 0 \ \vec{N}_2 \\ \mathbf{0} \ \vec{0} \ \mathbf{N}_3 \end{array} \right) \implies \left(\begin{array}{c} \mathbf{R} \ \vec{\beta} \ \mathbf{M}'_1 \\ \vec{0} \ 0 \ \vec{0} \\ \mathbf{0} \ \vec{0} \ \mathbf{M}_3 \end{array} \right), \left(\begin{array}{c} \mathbf{R} \ \vec{\beta} \ \mathbf{N}_1 \\ \vec{0} \ 0 \ \vec{N}_2 \\ \mathbf{0} \ \vec{0} \ \mathbf{N}_3 \end{array} \right)$$

Use the row $(\vec{0} \ 1 \ \vec{M}_2)$ to create β in the left argument.

Then remove the row $(\vec{0} \ 1 \ \vec{M}_2)$.

The right argument is unchanged.

Unifying ${}^t(\vec{\alpha} \ 0 \ \vec{0})$ and ${}^t(\vec{\beta} \ 0 \ \vec{0})$ is a bit more complicated...

A note on integers

Suppose now that $\mathbb{I} = \mathbb{Z}$.

- \mathbb{Z} is not closed under affine operations: $(x/y) \times y \neq x$,
- Gaussian reduction implemented in \mathbb{Z} is unsound.
(e.g. unsound normalization $2X + Y = 19 \not\Rightarrow X = 9$, by truncation)

One possible solution

- keep a representation using matrices with coefficients in \mathbb{Q} ,
- keep all abstract operators as in \mathbb{Q} ,
- change the concretization into: $\gamma_{\mathbb{Z}}(\mathcal{X}^{\#}) \stackrel{\text{def}}{=} \gamma(\mathcal{X}^{\#}) \cap \mathbb{Z}^n$.

With respect to $\gamma_{\mathbb{Z}}$, **the operators are no longer best / exact.**

Example: where $\mathcal{X}^{\#}$ is the equation $Y = 2X$

- $\gamma_{\mathbb{Z}}(\mathcal{X}^{\#}) = \{ (X, Y) \mid X \in \mathbb{Z}, Y = 2X \}$
- $(C[\![X := 0]\!] \circ \gamma_{\mathbb{Z}})\mathcal{X}^{\#} = \{ (X, Y) \mid X = 0, Y \text{ is even} \}$
- $(\gamma_{\mathbb{Z}} \circ C^{\#}[\![X := 0]\!])\mathcal{X}^{\#} = \{ (X, Y) \mid X = 0, Y \in \mathbb{Z} \}$

The analysis forgets the “intergenerness” of variables.

The congruence equality domain

Now, $\mathbb{I} = \mathbb{Z}$.

We look for invariants of the form: $\bigwedge_j \left(\sum_{i=1}^n m_{ij} v_i \equiv c_j [k_j] \right)$.

Algorithms:

- there exists minimal forms (but not unique), computed using an extension of **Euclide's algorithm**,
- there is a dual representation: $\{ \mathbf{G} \times \vec{\lambda} + \vec{O} \mid \vec{\lambda} \in \mathbb{Z}^m \}$, and passage algorithms,
- see [Gran91].

Polyhedron domain

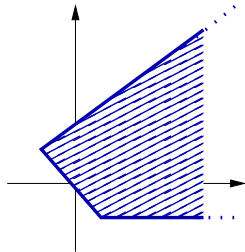
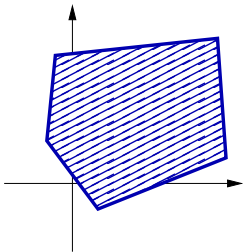
The polyhedron domain

Here again, $\mathbb{I} \in \{\mathbb{Q}, \mathbb{R}\}$.

We look for invariants of the form: $\bigwedge_j \left(\sum_{i=1}^n \alpha_{ij} v_i \geq \beta_j \right)$.

We use the polyhedron domain proposed by [Cous78]:

$$\mathcal{D}^\# \stackrel{\text{def}}{=} \{\text{closed convex polyhedra of } \mathbb{V} \rightarrow \mathbb{I}\}$$



Note: polyhedra need not be bounded (\neq polytopes).

Double description of polyhedra

Polyhedra have **dual** representations (Weyl–Minkowski Theorem).
(see [Schr86])

Constraint representation

$\langle \mathbf{M}, \vec{\mathbf{C}} \rangle$ with $\mathbf{M} \in \mathbb{I}^{m \times n}$ and $\vec{\mathbf{C}} \in \mathbb{I}^m$

represents: $\gamma(\langle \mathbf{M}, \vec{\mathbf{C}} \rangle) \stackrel{\text{def}}{=} \{ \vec{\mathbf{V}} \mid \mathbf{M} \times \vec{\mathbf{V}} \geq \vec{\mathbf{C}} \}$

We will also often use a **constraint set notation** $\{ \sum_i \alpha_{ij} \mathbf{v}_i \geq \beta_j \}$.

Generator representation

$[\mathbf{P}, \mathbf{R}]$ where

- $\mathbf{P} \in \mathbb{I}^{n \times p}$ is a set of p **points**: $\vec{P}_1, \dots, \vec{P}_p$
- $\mathbf{R} \in \mathbb{I}^{n \times r}$ is a set of r **rays**: $\vec{R}_1, \dots, \vec{R}_r$

$\gamma([\mathbf{P}, \mathbf{R}]) \stackrel{\text{def}}{=} \left\{ \left(\sum_{j=1}^p \alpha_j \vec{P}_j \right) + \left(\sum_{j=1}^r \beta_j \vec{R}_j \right) \mid \forall j, \alpha_j \geq 0, \sum_{j=1}^p \alpha_j = 1, \forall j, \beta_j \geq 0 \right\}$

Origin of duality

Dual $A^* \stackrel{\text{def}}{=} \{ \vec{x} \in \mathbb{I}^n \mid \forall \vec{a} \in A, \vec{a} \cdot \vec{x} \leq 0 \}$

- $\{\vec{a}\}^*$ and $\{\lambda \vec{r} \mid \lambda \geq 0\}^*$ are half-spaces,
- $(A \cup B)^* = A^* \cap B^*$,
- if A is convex, closed, and $\vec{0} \in A$, then $A^{**} = A$.

Duality on polyhedral cones:

Cone: $C = \{ \vec{V} \mid \mathbf{M} \times \vec{V} \geq \vec{0} \}$ or $C = \{ \sum_{j=1}^r \beta_j \vec{R}_j \mid \forall j, \beta_j \geq 0 \}$

- $C^{**} = C$,
- C^* is also a polyhedral cone,
- a ray of C corresponds to a constraint of C^* ,
- a constraint of C corresponds to a ray of C^* .

extended to polyhedra by homogenisation to polyhedral codes:

$$C(P) \stackrel{\text{def}}{=} \{ \lambda \vec{V} \mid \lambda \geq 0, (V_1, \dots, V_n) \in \gamma(P), V_{n+1} = 1 \} \subseteq \mathbb{I}^{n+1}$$

Polyhedra representation (cont.)

Minimal representations

- A constraint system is **minimal** if no constraint can be omitted without changing the concretization.
- A generator system is **minimal** if no generator can be omitted without changing the concretization.

Remarks:

- most operators are easier on one representation;
- minimal representations are **not unique**;
- there is **no memory bound** on the representations (even minimal ones);
- **equality** constraints, as well as **lines** (pairs of opposed rays) may be handled separately and more efficiently.

Chernikova's algorithm

Switch from a constraint system to an equivalent generator system.

Algorithm introduced by [Cher68].

Notes:

- By **duality**, we can use the same algorithm to switch from generators to constraints.
- The minimal generator system can be **exponential** in the original constraint system.
(e.g. a n -dimensional hyper-cube has $2n$ constraints and 2^n vertices)

Algorithm: **incrementally** add constraints one by one

Start with:
$$\begin{cases} \mathbf{P}_0 = \{ (0, \dots, 0) \} & \text{(origin)} \\ \mathbf{R}_0 = \{ \vec{x}_i, -\vec{x}_i \mid 1 \leq i \leq n \} & \text{(axes)} \end{cases}$$

Chernikova's algorithm (cont.)

Update $[\mathbf{P}_{k-1}, \mathbf{R}_{k-1}]$ to $[\mathbf{P}_k, \mathbf{R}_k]$

by adding one constraint $\vec{M}_k \cdot \vec{V} \geq C_k \in \langle \mathbf{M}, \vec{C} \rangle$:

start with $\mathbf{P}_k = \mathbf{R}_k = \emptyset$,

- for any $\vec{P} \in \mathbf{P}_{k-1}$ s.t. $\vec{M}_k \cdot \vec{P} \geq C_k$, add \vec{P} to \mathbf{P}_k ;
- for any $\vec{R} \in \mathbf{R}_{k-1}$ s.t. $\vec{M}_k \cdot \vec{R} \geq 0$, add \vec{R} to \mathbf{R}_k ;
- for any $\vec{P}, \vec{Q} \in \mathbf{P}_{k-1}$ s.t. $\vec{M}_k \cdot \vec{P} > C_k$ and $\vec{M}_k \cdot \vec{Q} < C_k$, add to \mathbf{P}_k :

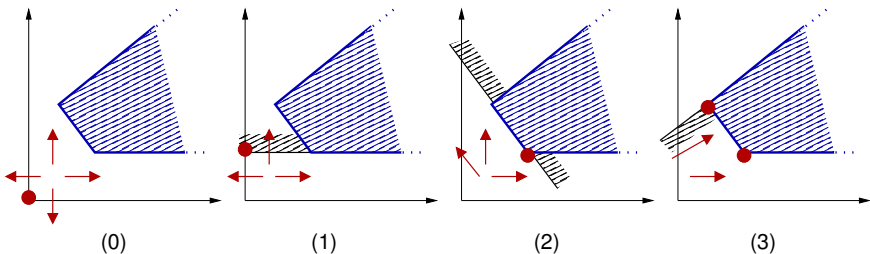
$$\frac{C_k - \vec{M}_k \cdot \vec{Q}}{\vec{M}_k \cdot \vec{P} - \vec{M}_k \cdot \vec{Q}} \vec{P} - \frac{C_k - \vec{M}_k \cdot \vec{P}}{\vec{M}_k \cdot \vec{P} - \vec{M}_k \cdot \vec{Q}} \vec{Q}$$

- for any $\vec{P} \in \mathbf{P}_{k-1}, \vec{R} \in \mathbf{R}_{k-1}$ s.t. either $\vec{M}_k \cdot \vec{P} > C_k$ and $\vec{M}_k \cdot \vec{R} < 0$, or $\vec{M}_k \cdot \vec{P} < C_k$ and $\vec{M}_k \cdot \vec{R} > 0$, add to \mathbf{P}_k :

$$\vec{P} + \frac{C_k - \vec{M}_k \cdot \vec{P}}{\vec{M}_k \cdot \vec{R}} \vec{R}$$

- for any $\vec{R}, \vec{S} \in \mathbf{R}_{k-1}$ s.t. $\vec{M}_k \cdot \vec{R} > 0$ and $\vec{M}_k \cdot \vec{S} < 0$, add to \mathbf{R}_k : $(\vec{M}_k \cdot \vec{S})\vec{R} - (\vec{M}_k \cdot \vec{R})\vec{S}$

Chernikova's algorithm (example)

Example:

	$\mathbf{P}_0 = \{(0, 0)\}$	$\mathbf{R}_0 = \{(1, 0); (-1, 0); (0, 1); (0, -1)\}$
$Y \geq 1$	$\mathbf{P}_1 = \{(0, 1)\}$	$\mathbf{R}_1 = \{(1, 0); (-1, 0); (0, 1)\}$
$X + Y \geq 3$	$\mathbf{P}_2 = \{(2, 1)\}$	$\mathbf{R}_2 = \{(1, 0); (-1, 1); (0, 1)\}$
$X - Y \leq 1$	$\mathbf{P}_3 = \{(2, 1); (1, 2)\}$	$\mathbf{R}_3 = \{(0, 1); (1, 1)\}$

Redudancy removal

Goal: only introduce non-redundant points and rays during Chernikova's algorithm.

Definitions (for rays in polyhedral cones)

Given $C = \{\vec{V} \mid \mathbf{M} \times \vec{V} \geq \vec{0}\} = \{\mathbf{R} \times \vec{\beta} \mid \vec{\beta} \geq \vec{0}\}$.

\vec{R} saturates $\vec{M}_k \cdot \vec{V} \geq 0 \stackrel{\text{def}}{\iff} \vec{M}_k \cdot \vec{R} = 0$.

$S(\vec{R}, C) \stackrel{\text{def}}{=} \{k \mid \vec{M}_k \cdot \vec{R} = 0\}$.

Theorem:

assume C has no line ($\nexists \vec{L} \neq \vec{0}$ s.t. $\forall \alpha, \alpha \vec{L} \in C$)

\vec{R} is non-redundant wrt. $\mathbf{R} \iff \nexists \vec{R}_i \in \mathbf{R}, S(\vec{R}, C) \subseteq S(\vec{R}_i, C)$

- $S(\vec{R}_i, C), \vec{R}_i \in \mathbf{R}$ is maintained during Chernikova's algorithm in a **saturation matrix**,
- extension possible to polyhedra and lines,
- various improvements exist [LeVe92].

Operators on polyhedra

Given $\mathcal{X}^\#, \mathcal{Y}^\# \neq \perp^\#$, we define:

$$\mathcal{X}^\# \subseteq^\# \mathcal{Y}^\# \stackrel{\text{def}}{\iff} \begin{cases} \forall \vec{P} \in \mathbf{P}_{\mathcal{X}^\#}, \mathbf{M}_{\mathcal{Y}^\#} \times \vec{P} \geq \vec{C}_{\mathcal{Y}^\#} \\ \forall \vec{R} \in \mathbf{R}_{\mathcal{X}^\#}, \mathbf{M}_{\mathcal{Y}^\#} \times \vec{R} \geq \vec{0} \end{cases}$$

$$\mathcal{X}^\# =^\# \mathcal{Y}^\# \stackrel{\text{def}}{\iff} \mathcal{X}^\# \subseteq^\# \mathcal{Y}^\# \text{ and } \mathcal{Y}^\# \subseteq^\# \mathcal{X}^\#$$

$$\mathcal{X}^\# \cap^\# \mathcal{Y}^\# \stackrel{\text{def}}{=} \left\langle \left[\begin{array}{c} \mathbf{M}_{\mathcal{X}^\#} \\ \mathbf{M}_{\mathcal{Y}^\#} \end{array} \right], \left[\begin{array}{c} \vec{C}_{\mathcal{X}^\#} \\ \vec{C}_{\mathcal{Y}^\#} \end{array} \right] \right\rangle \quad (\text{join constraint sets})$$

$$\mathcal{X}^\# \cup^\# \mathcal{Y}^\# \stackrel{\text{def}}{=} \left[[\mathbf{P}_{\mathcal{X}^\#} \ \mathbf{P}_{\mathcal{Y}^\#}], [\mathbf{R}_{\mathcal{X}^\#} \ \mathbf{R}_{\mathcal{Y}^\#}] \right] \quad (\text{join generator sets})$$

Remarks:

- $\subseteq^\#, =^\#$ and $\cap^\#$ are **exact**.
- $\cup^\#$ is **optimal**: we get the **topological closure of the convex hull** of $\gamma(\mathcal{X}^\#) \cup \gamma(\mathcal{Y}^\#)$.

Operators on polyhedra (cont.)

$$\mathbf{C}^\sharp[\sum_i \alpha_i \mathbf{v}_i + \beta \geq 0] \mathcal{X}^\sharp \stackrel{\text{def}}{=} \left\langle \left[\begin{array}{c} \mathbf{M}_{\mathcal{X}^\sharp} \\ \alpha_1 \cdots \alpha_n \end{array} \right], \left[\begin{array}{c} \vec{\mathbf{C}}_{\mathcal{X}^\sharp} \\ -\beta \end{array} \right] \right\rangle$$

$$\mathbf{C}^\sharp[\sum_i \alpha_i \mathbf{v}_i + \beta = 0] \mathcal{X}^\sharp \stackrel{\text{def}}{=} (\mathbf{C}^\sharp[\sum_i \alpha_i \mathbf{v}_i + \beta \geq 0] \circ \mathbf{C}^\sharp[\sum_i (-\alpha_i) \mathbf{v}_i - \beta \geq 0]) \mathcal{X}^\sharp$$

$$\mathbf{C}^\sharp[\mathbf{v}_j :=] - \infty, +\infty[] \mathcal{X}^\sharp \stackrel{\text{def}}{=} [\mathbf{P}_{\mathcal{X}^\sharp}, [\mathbf{R}_{\mathcal{X}^\sharp} \vec{x}_j \ (-\vec{x}_j)]]$$

$$\mathbf{C}^\sharp[\mathbf{v}_j := \sum_i \alpha_i \mathbf{v}_i + \beta] \mathcal{X}^\sharp \stackrel{\text{def}}{=} \begin{array}{l} \text{if } \alpha_j = 0, (\mathbf{C}^\sharp[\sum_i \alpha_i \mathbf{v}_i - \mathbf{v}_j + \beta = 0] \circ \mathbf{C}^\sharp[\mathbf{v}_j :=] - \infty, +\infty[]) \mathcal{X}^\sharp \\ \text{if } \alpha_j \neq 0, \langle \mathbf{M}, \vec{\mathbf{C}} \rangle \text{ where } \mathbf{v}_j \text{ is replaced with } \frac{1}{\alpha_j}(\mathbf{v}_j - \sum_{i \neq j} \alpha_i \mathbf{v}_i - \beta) \end{array}$$

Remarks:

- $\mathbf{C}^\sharp[\sum_i \alpha_i \mathbf{v}_i + \beta \geq 0]$, $\mathbf{C}^\sharp[\mathbf{v}_j := \sum_i \alpha_i \mathbf{v}_i + \beta] \mathcal{X}$ and $\mathbf{C}^\sharp[\mathbf{v}_j :=] - \infty, +\infty[]$ are **exact**.
- We can also define $\mathbf{C}^\sharp[\mathbf{v}_j := \sum_i \alpha_i \mathbf{v}_i + \beta]$ on a generator system.

Operators on polyhedra (cont.)

Backward assignments:

$$C^\# \left[\overleftarrow{V_j :=} -\infty, +\infty \right] (\mathcal{X}^\#, \mathcal{R}^\#) \stackrel{\text{def}}{=} \mathcal{X}^\# \cap^\# (C^\# [V_j :=] -\infty, +\infty [] \mathcal{R}^\#)$$

$$C^\# \left[\overleftarrow{V_j := \sum_i \alpha_i V_i + \beta} \right] (\mathcal{X}^\#, \mathcal{R}^\#) \stackrel{\text{def}}{=} \\ \mathcal{X}^\# \cap^\# (\mathcal{R}^\# \text{ where } V_j \text{ is replaced with } (\sum_i \alpha_i V_i + \beta))$$

$$C^\# \left[\overleftarrow{V_j := e} \right] (\mathcal{X}^\#, \mathcal{R}^\#) \stackrel{\text{def}}{=} C^\# \left[\overleftarrow{V_j :=} -\infty, +\infty \right] (\mathcal{X}^\#, \mathcal{R}^\#)$$

for other assignments

Note: identical to the case of linear equalities.

Polyhedra widening

$\mathcal{D}^\#$ has strictly increasing infinite chains \implies we need a widening.

Definition:

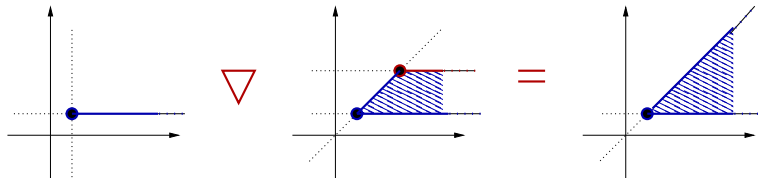
Take $\mathcal{X}^\#$ and $\mathcal{Y}^\#$ in minimal constraint-set form.

$$\mathcal{X}^\# \nabla \mathcal{Y}^\# \stackrel{\text{def}}{=} \left\{ c \in \mathcal{X}^\# \mid \mathcal{Y}^\# \subseteq^\# \{c\} \right\} \cup \left\{ c \in \mathcal{Y}^\# \mid \exists c' \in \mathcal{X}^\#, \mathcal{X}^\# =^\# (\mathcal{X}^\# \setminus c') \cup \{c\} \right\}.$$

We suppress any unstable constraint $c \in \mathcal{X}^\#$, i.e., $\mathcal{Y}^\# \not\subseteq^\# \{c\}$.

However, we keep constraints $c \in \mathcal{Y}^\#$ equivalent to those in $\mathcal{X}^\#$, i.e., when $\exists c' \in \mathcal{X}^\#, \mathcal{X}^\# =^\# (\mathcal{X}^\# \setminus c') \cup \{c\}$.

Example:



Example analysis

Example program

```

X:=2; I:=0;
while • I<10 do
  if ? then X:=X+2 else X:=X-3 fi;
  I:=I+1
done◆

```

We use a finite number (one) of intersections $\cap^\#$ as narrowing.
Iterations with widening and narrowing at \bullet give:

$$\mathcal{X}_{\bullet}^{\#1} = \{X = 2, I = 0\}$$

$$\begin{aligned} \mathcal{X}_{\bullet}^{\#2} &= \{X = 2, I = 0\} \nabla (\{X = 2, I = 0\} \cup^\# \{X \in [-1, 4], I = 1\}) \\ &= \{X = 2, I = 0\} \nabla \{I \in [0, 1], 2 - 3I \leq X \leq 2I + 2\} \\ &= \{I \geq 0, 2 - 3I \leq X \leq 2I + 2\} \end{aligned}$$

$$\begin{aligned} \mathcal{X}_{\bullet}^{\#3} &= \{I \geq 0, 2 - 3I \leq X \leq 2I + 2\} \cap^\# \\ &\quad (\{X = 2, I = 0\} \cup^\# \{I \in [1, 10], 2 - 3I \leq X \leq 2I + 2\}) \\ &= \{I \in [0, 10], 2 - 3I \leq X \leq 2I + 2\} \end{aligned}$$

At \blacklozenge we find eventually: $I = 10 \wedge X \in [-28, 22]$.

Other polyhedra widenings

Widening with thresholds:

Given a **finite** set T of **constraints**, we add to $\mathcal{X}^\# \nabla \mathcal{Y}^\#$ all the constraints from T satisfied by both $\mathcal{X}^\#$ and $\mathcal{Y}^\#$.

Delayed widening:

We replace $\mathcal{X}^\# \nabla \mathcal{Y}^\#$ with $\mathcal{X}^\# \cup^\# \mathcal{Y}^\#$ a **finite** number of times (this works for any widening and abstract domain).

See also [Bagn03].

Strict inequalities

The polyhedron domain can be extended to allow strict constraints: $\{ \vec{V} \mid \mathbf{M} \times \vec{V} \geq \vec{C} \text{ and } \mathbf{M}' \times \vec{V} > \vec{C}' \}$

Idea:

A **non-closed** polyhedron on \mathbb{V} is represented as a **closed** polyhedron P on $\mathbb{V}' \stackrel{\text{def}}{=} \mathbb{V} \cup \{V_\epsilon\}$.

$$\begin{array}{ll} \alpha_1 V_1 + \dots + \alpha_n V_n + 0V_\epsilon \geq 0 & \text{represents } \alpha_1 V_1 + \dots + \alpha_n V_n \geq 0 \\ \alpha_1 V_1 + \dots + \alpha_n V_n - cV_\epsilon \geq 0, c > 0 & \text{represents } \alpha_1 V_1 + \dots + \alpha_n V_n > 0 \end{array}$$

P represents the non necessarily closed polyhedron:

$$\gamma_\epsilon(P) \stackrel{\text{def}}{=} \{(V_1, \dots, V_n) \mid \exists V_\epsilon > 0, (V_1, \dots, V_n, V_\epsilon) \in \gamma(P)\}.$$

Notes:

- The minimal form needs some adaptation [Bagn02].
- Chernikova's algorithm, \cap^\sharp , \cup^\sharp , $C^\sharp[[c]]$, and $C^\sharp[[\overleftarrow{c}]]$ can be easily reused.

Integer polyhedra

How can we deal with $\mathbb{I} = \mathbb{Z}$?

Issue: integer linear programming is difficult.

Example: satisfiability of conjunctions of linear constraints:

- polynomial cost in \mathbb{Q} ,
- NP-complete cost in \mathbb{Z} .

Possible solutions:

- Use some complete integer algorithms.
(e.g. Presburger arithmetics)
Costly, and we do not have any abstract domain structure.
- Keep \mathbb{Q} -polyhedra as representation, and change the concretization into: $\gamma_{\mathbb{Z}}(\mathcal{X}^{\#}) \stackrel{\text{def}}{=} \gamma(\mathcal{X}^{\#}) \cap \mathbb{Z}^n$.
However, operators are no longer exact / optimal.

Weakly relational domains

Zone domain

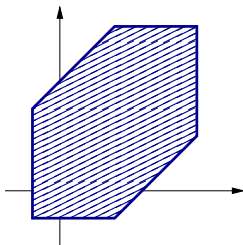
The zone domain

Here, $\mathbb{I} \in \{\mathbb{Z}, \mathbb{Q}, \mathbb{R}\}$.

We look for invariants of the form:

$$\bigwedge v_i - v_j \leq c \text{ or } \pm v_i \leq c, \quad c \in \mathbb{I}$$

A subset of \mathbb{I}^n bounded by such constraints is called a **zone**.



[Mine01a]

Machine representation

A **potential constraint** has the form: $V_j - V_i \leq c$.

Potential graph: directed, weighted graph \mathcal{G}

- nodes are labelled with variables in \mathbb{V} ,
- we add an arc with **weight** c from V_i to V_j for each constraint $V_j - V_i \leq c$.

Difference Bound Matrix (DBM)

Adjacency matrix \mathbf{m} of \mathcal{G} :

- \mathbf{m} is square, with size $n \times n$, and elements in $\mathbb{I} \cup \{+\infty\}$,
- $m_{ij} = c < +\infty$ denotes the constraint $V_j - V_i \leq c$,
- $m_{ij} = +\infty$ if there is no upper bound on $V_j - V_i$.

Concretization:

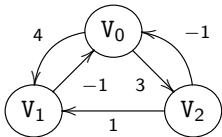
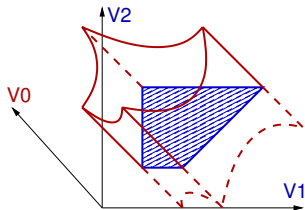
$$\gamma(\mathbf{m}) \stackrel{\text{def}}{=} \{ (v_1, \dots, v_n) \in \mathbb{I}^n \mid \forall i, j, v_j - v_i \leq m_{ij} \}.$$

Machine representation (cont.)

Unary constraints add a constant null variable V_0 .

- \mathbf{m} has size $(n + 1) \times (n + 1)$;
- $V_i \leq c$ is denoted as $V_i - V_0 \leq c$, i.e., $m_{i0} = c$;
- $V_i \geq c$ is denoted as $V_0 - V_i \leq -c$, i.e., $m_{0i} = -c$;
- γ is now: $\gamma_0(\mathbf{m}) \stackrel{\text{def}}{=} \{ (v_1, \dots, v_n) \mid (0, v_1, \dots, v_n) \in \gamma(\mathbf{m}) \}$.

Example:



	V_0	V_1	V_2
V_0	$+\infty$	4	3
V_1	-1	$+\infty$	$+\infty$
V_2	-1	1	$+\infty$

The DBM lattice

$\mathcal{D}^\#$ contains all DBMs, plus $\perp^\#$.

\leq on $\mathbb{I} \cup \{+\infty\}$ is extended **point-wisely**.

If $\mathbf{m}, \mathbf{n} \neq \perp^\#$:

$$\begin{array}{ll}
 \mathbf{m} \subseteq^\# \mathbf{n} & \stackrel{\text{def}}{\iff} \forall i, j, m_{ij} \leq n_{ij} \\
 \mathbf{m} =^\# \mathbf{n} & \stackrel{\text{def}}{\iff} \forall i, j, m_{ij} = n_{ij} \\
 [\mathbf{m} \cap^\# \mathbf{n}]_{ij} & \stackrel{\text{def}}{=} \min(m_{ij}, n_{ij}) \\
 [\mathbf{m} \cup^\# \mathbf{n}]_{ij} & \stackrel{\text{def}}{=} \max(m_{ij}, n_{ij}) \\
 [\top^\#]_{ij} & \stackrel{\text{def}}{=} +\infty
 \end{array}$$

$(\mathcal{D}^\#, \subseteq^\#, \cup^\#, \cap^\#, \perp^\#, \top^\#)$ is a **lattice**.

Remarks:

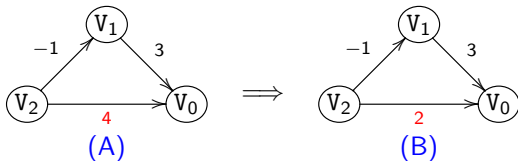
- $\mathcal{D}^\#$ is complete if \leq is ($\mathbb{I} = \mathbb{R}$ or \mathbb{Z} , but not \mathbb{Q}),
- $\mathbf{m} \subseteq^\# \mathbf{n} \implies \gamma_0(\mathbf{m}) \subseteq \gamma_0(\mathbf{n})$, but **not the converse**,
- $\mathbf{m} =^\# \mathbf{n} \implies \gamma_0(\mathbf{m}) = \gamma_0(\mathbf{n})$, but **not the converse**.

Normal form, equality and inclusion testing

Issue: how can we compare $\gamma_0(\mathbf{m})$ and $\gamma_0(\mathbf{n})$?

Idea: find a normal form by **propagating/tightening constraints**.

$$\left\{ \begin{array}{l} V_0 - V_1 \leq 3 \\ V_1 - V_2 \leq -1 \\ V_0 - V_2 \leq 4 \end{array} \right. \quad \Rightarrow \quad \left\{ \begin{array}{l} V_0 - V_1 \leq 3 \\ V_1 - V_2 \leq -1 \\ V_0 - V_2 \leq 2 \end{array} \right.$$



Definition: shortest-path closure \mathbf{m}^*

$$m_{ij}^* \stackrel{\text{def}}{=} \min_N \sum_{k=1}^{N-1} m_{i_k i_{k+1}} \\ \langle i = i_1, \dots, i_N = j \rangle$$

Exists only when \mathbf{m} has no cycle with strictly negative weight.

Floyd–Warshall algorithm

Properties:

- $\gamma_0(\mathbf{m}) = \emptyset \iff \mathcal{G}$ has a cycle with strictly negative weight.
- if $\gamma_0(\mathbf{m}) \neq \emptyset$, the shortest-path graph \mathbf{m}^* is a normal form:

$$\mathbf{m}^* = \min_{\subseteq^\#} \{ \mathbf{n} \mid \gamma_0(\mathbf{m}) = \gamma_0(\mathbf{n}) \}$$
- If $\gamma_0(\mathbf{m}), \gamma_0(\mathbf{n}) \neq \emptyset$, then
 - $\gamma_0(\mathbf{m}) = \gamma_0(\mathbf{n}) \iff \mathbf{m}^* =^\# \mathbf{n}^*$,
 - $\gamma_0(\mathbf{m}) \subseteq \gamma_0(\mathbf{n}) \iff \mathbf{m}^* \subseteq^\# \mathbf{n}$.

Floyd–Warshall algorithm

$$\begin{cases} m_{ij}^0 & \stackrel{\text{def}}{=} m_{ij} \\ m_{ij}^{k+1} & \stackrel{\text{def}}{=} \min(m_{ij}^k, m_{ik}^k + m_{kj}^k) \end{cases}$$

- If $\gamma_0(\mathbf{m}) \neq \emptyset$, then $\mathbf{m}^* = \mathbf{m}^{n+1}$, (normal form)
- $\gamma_0(\mathbf{m}) = \emptyset \iff \exists i, m_{ii}^{n+1} < 0$, (emptiness testing)
- \mathbf{m}^{n+1} can be computed in $\mathcal{O}(n^3)$ time.

Abstract operators

Abstract union $\cup^\#$

- $\gamma_0(\mathbf{m} \cup^\# \mathbf{n})$ may not be the smallest zone containing $\gamma_0(\mathbf{m})$ and $\gamma_0(\mathbf{n})$.
- however, $(\mathbf{m}^*) \cup^\# (\mathbf{n}^*)$ is **optimal**:

$$(\mathbf{m}^*) \cup^\# (\mathbf{n}^*) = \min_{\subseteq^\#} \{ \mathbf{o} \mid \gamma_0(\mathbf{o}) \supseteq \gamma_0(\mathbf{m}) \cup \gamma_0(\mathbf{n}) \}$$
 which implies

$$\gamma_0((\mathbf{m}^*) \cup^\# (\mathbf{n}^*)) = \min_{\subseteq} \{ \gamma_0(\mathbf{o}) \mid \gamma_0(\mathbf{o}) \supseteq \gamma_0(\mathbf{m}) \cup \gamma_0(\mathbf{n}) \}$$
- $(\mathbf{m}^*) \cup^\# (\mathbf{n}^*)$ is always closed.

Abstract intersection $\cap^\#$

- $\cap^\#$ is **always exact**: $\gamma_0(\mathbf{m} \cap^\# \mathbf{n}) = \gamma_0(\mathbf{m}) \cap \gamma_0(\mathbf{n})$
- $(\mathbf{m}^*) \cap^\# (\mathbf{n}^*)$ may not be closed.

Remark:

The set of closed matrices with $\perp^\#$, and the operations $\subseteq^\#$, $\cup^\#$, $\lambda \mathbf{m}, \mathbf{n}. (\mathbf{m} \cap^\# \mathbf{n})^*$ define a sub-lattice.

γ_0 is injective in this sub-lattice.

Abstract operators (cont.)

We can define:

$$[C^\sharp \llbracket V_{j_0} - V_{i_0} \leq c \rrbracket \mathbf{m}]_{ij} \stackrel{\text{def}}{=} \begin{cases} \min(m_{ij}, c) & \text{if } (i, j) = (i_0, j_0), \\ m_{ij} & \text{otherwise.} \end{cases}$$

$$C^\sharp \llbracket V_{j_0} - V_{i_0} = [a, b] \rrbracket \mathbf{m} \stackrel{\text{def}}{=} (C^\sharp \llbracket V_{j_0} - V_{i_0} \leq b \rrbracket \circ C^\sharp \llbracket V_{i_0} - V_{j_0} \leq -a \rrbracket) \mathbf{m}$$

$$[C^\sharp \llbracket V_{j_0} := \rrbracket - \infty, +\infty \rrbracket \mathbf{m}]_{ij} \stackrel{\text{def}}{=} \begin{cases} +\infty & \text{if } i = j_0 \text{ or } j = j_0, \\ m_{ij}^* & \text{otherwise.} \end{cases}$$

(not optimal on non-closed arguments)

$$C^\sharp \llbracket V_{j_0} := V_{i_0} + [a, b] \rrbracket \mathbf{m} \stackrel{\text{def}}{=} (C^\sharp \llbracket V_{j_0} - V_{i_0} = [a, b] \rrbracket \circ C^\sharp \llbracket V_{j_0} := \rrbracket - \infty, +\infty \rrbracket) \mathbf{m} \quad \text{if } i_0 \neq j_0$$

$$[C^\sharp \llbracket V_{j_0} := V_{j_0} + [a, b] \rrbracket \mathbf{m}]_{ij} \stackrel{\text{def}}{=} \begin{cases} m_{ij} - a & \text{if } i = j_0 \text{ and } j \neq j_0 \\ m_{ij} + b & \text{if } i \neq j_0 \text{ and } j = j_0 \\ m_{ij} & \text{otherwise.} \end{cases}$$

($i_0 \neq j_0$; V_{i_0} can be replaced with 0 by setting $i_0 = 0$)

These transfer functions are **exact**.

Abstract operators (cont.)

Backward assignment:

$$C^\# \llbracket \overleftarrow{V_{j_0} :=} -\infty, +\infty \rrbracket (\mathbf{m}, \mathbf{r}) \stackrel{\text{def}}{=} \mathbf{m} \cap^\# (C^\# \llbracket V_{j_0} := \rrbracket -\infty, +\infty \rrbracket \mathbf{r})$$

$$C^\# \llbracket \overleftarrow{V_{j_0} := V_{j_0} + [a, b]} \rrbracket (\mathbf{m}, \mathbf{r}) \stackrel{\text{def}}{=} \mathbf{m} \cap^\# (C^\# \llbracket V_{j_0} := V_{j_0} + [-b, -a] \rrbracket \mathbf{r})$$

$$\left[C^\# \llbracket \overleftarrow{V_{j_0} := V_{i_0} + [a, b]} \rrbracket (\mathbf{m}, \mathbf{r}) \right]_{ij} \stackrel{\text{def}}{=} \mathbf{m} \cap^\# \begin{cases} \min(\mathbf{r}_{ij}^*, \mathbf{r}_{j_0j}^* + b) & \text{if } i = i_0 \text{ and } j \neq i_0, j_0 \\ \min(\mathbf{r}_{ij}^*, \mathbf{r}_{ij_0}^* - a) & \text{if } j = i_0 \text{ and } i \neq i_0, j_0 \\ +\infty & \text{if } i = j_0 \text{ or } j = j_0 \\ \mathbf{r}_{ij}^* & \text{otherwise.} \end{cases}$$

Abstract operators (cont.)

Issue: given an arbitrary linear assignment $V_{j_0} := a_0 + \sum_k a_k \times V_k$

- there is no exact abstraction, in general;
- the best abstraction $\alpha \circ C[[c]] \circ \gamma$ is costly to compute.
(e.g. convert to a polyhedron and back, with exponential cost)

Possible solution:

Given a (more general) assignment $e = [a_0, b_0] + \sum_k [a_k, b_k] \times V_k$

we define an **approximate** operator as follows:

$$[C^\sharp[[V_{j_0} := e]] \mathbf{m}]_{ij} \stackrel{\text{def}}{=} \begin{cases} \max(E^\sharp[[e]] \mathbf{m}) & \text{if } i = 0 \text{ and } j = j_0 \\ -\min(E^\sharp[[e]] \mathbf{m}) & \text{if } i = j_0 \text{ and } j = 0 \\ \max(E^\sharp[[e - V_i]] \mathbf{m}) & \text{if } i \neq 0, j_0 \text{ and } j = j_0 \\ -\min(E^\sharp[[e + V_j]] \mathbf{m}) & \text{if } i = j_0 \text{ and } j \neq 0, j_0 \\ m_{ij} & \text{otherwise} \end{cases}$$

where $E^\sharp[[e]] \mathbf{m}$ evaluates e using interval arithmetics with $V_k \in [-m_{k0}^*, m_{0k}^*]$.

Quadratic total cost (plus the cost of closure).

Abstract operators (cont.)

Example:

Argument

$$\left\{ \begin{array}{l} 0 \leq Y \leq 10 \\ 0 \leq Z \leq 10 \\ 0 \leq Y - Z \leq 10 \end{array} \right.$$

$\Downarrow X := Y - Z$

$$\left\{ \begin{array}{l} -10 \leq X \leq 10 \\ -20 \leq X - Y \leq 10 \\ -20 \leq X - Z \leq 10 \end{array} \right.$$

Intervals

$$\left\{ \begin{array}{l} -10 \leq X \leq 10 \\ -10 \leq X - Y \leq 0 \\ -10 \leq X - Z \leq 10 \end{array} \right.$$

Approximate
solution

$$\left\{ \begin{array}{l} 0 \leq X \leq 10 \\ -10 \leq X - Y \leq 0 \\ -10 \leq X - Z \leq 10 \end{array} \right.$$

Best
(polyhedra)

We have a good trade-off between cost and precision.

The same idea can be used for tests and backward assignments.

Widening and narrowing

The zone domain has both strictly increasing and decreasing infinite chains.

Widening ∇

$$[\mathbf{m} \nabla \mathbf{n}]_{ij} \stackrel{\text{def}}{=} \begin{cases} m_{ij} & \text{if } n_{ij} \leq m_{ij} \\ +\infty & \text{otherwise} \end{cases}$$

Unstable constraints are deleted.

Narrowing Δ

$$[\mathbf{m} \Delta \mathbf{n}]_{ij} \stackrel{\text{def}}{=} \begin{cases} n_{ij} & \text{if } m_{ij} = +\infty \\ m_{ij} & \text{otherwise} \end{cases}$$

Only $+\infty$ bounds are refined.

Remarks:

- We can construct widenings with thresholds.
- ∇ (resp. Δ) can be seen as a **point-wise extension** of an interval widening (resp. narrowing).

Interaction between closure and widening

Widening ∇ and closure $*$ cannot always be mixed safely:

- $\mathbf{m}_{i+1} \stackrel{\text{def}}{=} \mathbf{m}_i \nabla (\mathbf{n}_i^*)$ OK
- $\mathbf{m}_{i+1} \stackrel{\text{def}}{=} (\mathbf{m}_i^*) \nabla \mathbf{n}_i$ wrong!
- $\mathbf{m}_{i+1} \stackrel{\text{def}}{=} (\mathbf{m}_i \nabla \mathbf{n}_i)^*$ wrong

otherwise the sequence (\mathbf{m}_i) may be infinite!

Example:

```

X:=0; Y:=[-1,1];
while • true do
  R:=[-1,1];
  if X=Y then Y:=X+R
  else X:=Y+R fi
done
  
```

$\mathcal{X}^{\#2j}$	$\mathcal{X}^{\#2j+1}$
$X \in [-2j, 2j]$	$X \in [-2j - 2, 2j + 2]$
$Y \in [-2j - 1, 2j + 1]$	$Y \in [-2j - 1, 2j + 1]$
$X - Y \in [-1, 1]$	$X - Y \in [-1, 1]$

Applying the closure after the widening at \bullet prevents convergence. Without the closure, we would find in finite time $X - Y \in [-1, 1]$.

Note: this situation also occurs in **reduced products**

(here, $\mathcal{D}^{\#} \simeq$ reduced product of $n \times n$ intervals, $*$ \simeq reduction)

Octagon domain

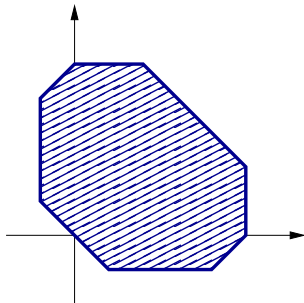
The octagon domain

Now, $\mathbb{I} \in \{\mathbb{Q}, \mathbb{R}\}$.

We look for invariants of the form: $\bigwedge \pm v_i \pm v_j \leq c, \quad c \in \mathbb{I}$

A subset of \mathbb{I}^n defined by such constraints is called an **octagon**.

It is a generalisation of zones (more symmetric).



[Mine01b]

Machine representation

Idea: use a **variable change** to get back to potential constraints.

Let $\mathbb{V}' \stackrel{\text{def}}{=} \{V'_1, \dots, V'_{2n}\}$.

the constraint:	is encoded as:
$V_i - V_j \leq c \quad (i \neq j)$	$V'_{2i-1} - V'_{2j-1} \leq c$ and $V'_{2j} - V'_{2i} \leq c$
$V_i + V_j \leq c \quad (i \neq j)$	$V'_{2i-1} - V'_{2j} \leq c$ and $V'_{2j-1} - V'_{2i} \leq c$
$-V_i - V_j \leq c \quad (i \neq j)$	$V'_{2j} - V'_{2i-1} \leq c$ and $V'_{2i} - V'_{2j-1} \leq c$
$V_i \leq c$	$V'_{2i-1} - V'_{2i} \leq 2c$
$V_i \geq c$	$V'_{2i} - V'_{2i-1} \leq -2c$

We use a matrix \mathbf{m} of size $(2n) \times (2n)$ with elements in $\mathbb{I} \cup \{+\infty\}$ and $\gamma_{\pm}(\mathbf{m}) \stackrel{\text{def}}{=} \{(v_1, \dots, v_n) \mid (v_1, -v_1, \dots, v_n, -v_n) \in \gamma(\mathbf{m})\}$.

Note:

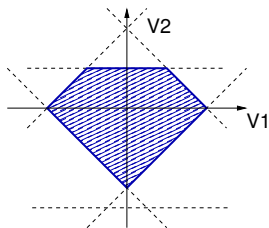
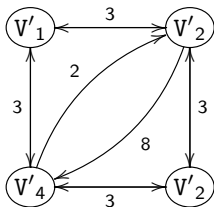
Two distinct \mathbf{m} elements can represent the same constraint on \mathbb{V} .

To avoid this, we impose that $\forall i, j, m_{ij} = m_{j\bar{i}}$ where $\bar{i} = i \oplus 1$.

Machine representation (cont.)

Example:

$$\left\{ \begin{array}{l} V_1 + V_2 \leq 3 \\ V_2 - V_1 \leq 3 \\ V_1 - V_2 \leq 3 \\ -V_1 - V_2 \leq -3 \\ 2V_2 \leq 2 \\ -2V_2 \leq 8 \end{array} \right.$$



Lattice

Constructed by point-wise extension of \leq on $\mathbb{I} \cup \{+\infty\}$.

Algorithms

\mathbf{m}^* is not a normal form for γ_{\pm} .

Idea use **two** local transformations instead of one:

$$\begin{cases} V'_i - V'_k \leq c \\ V'_k - V'_j \leq d \end{cases} \implies V'_i - V'_j \leq c + d$$

and

$$\begin{cases} V'_i - V'_{\bar{i}} \leq c \\ V'_{\bar{j}} - V'_j \leq d \end{cases} \implies V'_i - V'_j \leq (c + d)/2$$

Modified Floyd–Warshall algorithm

$$\mathbf{m}^{\bullet} \stackrel{\text{def}}{=} S(\mathbf{m}^{2n+1})$$

where:

$$(A) \quad \begin{cases} \mathbf{m}^1 \stackrel{\text{def}}{=} \mathbf{m} \\ \mathbf{m}^{k+1} \stackrel{\text{def}}{=} \min(n_{ij}, n_{ik} + n_{kj}), 1 \leq k \leq 2n \end{cases}$$

$$(B) \quad [S(\mathbf{n})]_{ij} \stackrel{\text{def}}{=} \min(n_{ij}, (n_{i\bar{i}} + n_{\bar{j}j})/2)$$

Algorithms (cont.)

Applications

- $\gamma_{\pm}(\mathbf{m}) = \emptyset \iff \exists i, \mathbf{m}_{ii}^{\bullet} < 0$,
- if $\gamma_{\pm}(\mathbf{m}) \neq \emptyset$, \mathbf{m}^{\bullet} is a normal form:

$$\mathbf{m}^{\bullet} = \min_{\subseteq^{\#}} \{ \mathbf{n} \mid \gamma_{\pm}(\mathbf{n}) = \gamma_{\pm}(\mathbf{m}) \},$$
- $(\mathbf{m}^{\bullet}) \cup^{\#} (\mathbf{n}^{\bullet})$ is the best abstraction for the set-union
 $\gamma_{\pm}(\mathbf{m}) \cup \gamma_{\pm}(\mathbf{n})$.

Widening and narrowing

- The zone widening and narrowing can be used on octagons.
- The widened iterates should not be closed.
 (prevents convergence)

Abstract transfer functions are similar to the case of the zone domain.

Analysis example

Rate limiter

```

Y:=0; while • true do
  X:=[-128,128]; D:=[0,16];
  S:=Y; Y:=X; R:=X-S;
  if R<=-D then Y:=S-D fi;
  if R>=D then Y:=S+D fi
done

```

X: input signal
Y: output signal
S: last output
R: delta Y-S
D: max. allowed for |R|

Analysis using:

- the octagon domain,
- an abstract operator for $V_{j_0} := [a_0, b_0] + \sum_k [a_k, b_k] \times V_k$ similar to the one we defined on zones,
- a widening with thresholds T .

Result: we prove that $|Y|$ is bounded by: $\min \{ t \in T \mid t \geq 144 \}$.

Note: the polyhedron domain would find $|Y| \leq 128$ and does not require thresholds, but it is more costly.

Integer octagons

Recall that zones work equally well on \mathbb{Q} , \mathbb{R} and \mathbb{Z} .

Issue:

The octagon domain we have presented is not complete on \mathbb{Z} :

- the algorithm for \mathbf{m}^\bullet uses divisions by 2,
- when replacing $x \mapsto x/2$ with $\mapsto \lfloor x/2 \rfloor$, we get:
 $\mathbf{m}^\bullet \neq \min_{\subseteq\#} \{ \mathbf{o} \mid \gamma_{\pm}(\mathbf{o}) = \gamma_{\pm}(\mathbf{m}) \}$.

Possible solutions:

- Use \mathbf{m}^\bullet with $\lfloor x/2 \rfloor$ instead of $/2$.
 All computations remain **sound** on integers.
 The best-precision results are no longer valid.
- See [Bagn08] for a $\mathcal{O}(n^3)$ time “tight closure” for integer octagons.

Linearization

Abstraction framework

Issue:

Most relational domains can only deal with linear expressions.
How can we abstract non-linear assignments such as $X := Y \times Z$?

Idea: replace $Y \times Z$ with a **sound linear** approximation.

Framework:

We define an **approximation preorder** \preceq on expressions:

$$R \models e_1 \preceq e_2 \stackrel{\text{def}}{\iff} \forall \rho \in R, E[e_1] \rho \subseteq E[e_2] \rho.$$

Soundness properties if $\gamma(\mathcal{X}^\#) \models e \preceq e'$ then:

- $C[V := e] \gamma(\mathcal{X}^\#) \subseteq \gamma(C^\#[V := e'] \mathcal{X}^\#)$
- $C[e \bowtie 0] \gamma(\mathcal{X}^\#) \subseteq \gamma(C^\#[e' \bowtie 0] \mathcal{X}^\#)$
- $\gamma(\mathcal{X}^\#) \cap (C[\overleftarrow{V} := e] \gamma(\mathcal{R}^\#)) \subseteq \gamma(C^\#[\overleftarrow{V} := e']^\#(\mathcal{X}^\#, \mathcal{R}^\#))$

\implies we can now use e' in the abstract instead of e .

Linearization

In practice, we put expressions into **affine interval form**:

$$\text{exp}_\ell : [a_0, b_0] + \sum_k [a_k, b_k] \times v_k$$

Advantages:

- **affine** expressions are easy to manipulate,
- **interval coefficients** allow non-determinism in expressions, hence, the opportunity for **abstraction**,
- we can easily construct abstract transfer functions for affine interval expressions.

Linearization (cont.)

Operations on affine interval forms

- adding \boxplus and subtracting \boxminus two forms,
- multiplying \boxtimes and dividing \boxdiv a form by an interval.

Noting i_k the interval $[a_k, b_k]$ and using interval operations $+_b^\#, -_b^\#, \times_b^\#, /_b^\#$ (e.g., $[a, b] +_b^\# [c, d] = [a + c, b + d]$):

- $(i_0 + \sum_k i_k \times v_k) \boxplus (i'_0 + \sum_k i'_k \times v_k) \stackrel{\text{def}}{=} (i_0 +_b^\# i'_0) + \sum_k (i_k +_b^\# i'_k) \times v_k$
- $i \boxtimes (i_0 + \sum_k i_k \times v_k) \stackrel{\text{def}}{=} (i \times_b^\# i_0) + \sum_k (i \times_b^\# i_k) \times v_k$
- ...

Projection $\pi_k : \mathcal{D}^\# \rightarrow \text{exp}_\ell$

We suppose we are given an **abstract interval projection** operator π_k such that:

$$\pi_k(\mathcal{X}^\#) = [a, b] \text{ such that } [a, b] \supseteq \{ \rho(v_k) \mid \rho \in \gamma(\mathcal{X}^\#) \}.$$

Linearization (cont.)

Intervalization $\iota : (\text{exp}_\ell \times \mathcal{D}^\#) \rightarrow \text{exp}_\ell$

Flattens the expression into a single interval:

$$\iota(i_0 + \sum_k (i_k \times v_k), \mathcal{X}^\#) \stackrel{\text{def}}{=} i_0 + \# \sum_{b,k} (i_k \times \# \pi_k(\mathcal{X}^\#)).$$

Linearization $\ell : (\text{exp} \times \mathcal{D}^\#) \rightarrow \text{exp}_\ell$

Defined by induction on the syntax of expressions:

- $\ell(v, \mathcal{X}^\#) \stackrel{\text{def}}{=} [1, 1] \times v,$
- $\ell([a, b], \mathcal{X}^\#) \stackrel{\text{def}}{=} [a, b],$
- $\ell(e_1 + e_2, \mathcal{X}^\#) \stackrel{\text{def}}{=} \ell(e_1, \mathcal{X}^\#) \boxplus \ell(e_2, \mathcal{X}^\#),$
- $\ell(e_1 - e_2, \mathcal{X}^\#) \stackrel{\text{def}}{=} \ell(e_1, \mathcal{X}^\#) \boxminus \ell(e_2, \mathcal{X}^\#),$
- $\ell(e_1 / e_2, \mathcal{X}^\#) \stackrel{\text{def}}{=} \ell(e_1, \mathcal{X}^\#) \boxtimes \iota(\ell(e_2, \mathcal{X}^\#), \mathcal{X}^\#),$
- $\ell(e_1 \times e_2, \mathcal{X}^\#) \stackrel{\text{def}}{=} \text{can be } \begin{cases} \text{either } & \iota(\ell(e_1, \mathcal{X}^\#), \mathcal{X}^\#) \boxtimes \ell(e_2, \mathcal{X}^\#), \\ \text{or} & \iota(\ell(e_2, \mathcal{X}^\#), \mathcal{X}^\#) \boxtimes \ell(e_1, \mathcal{X}^\#). \end{cases}$

Linearization application

Property soundness of the linearization:

For any abstract domain \mathcal{D}^\sharp , any $\mathcal{X}^\sharp \in \mathcal{D}^\sharp$ and $e \in \text{exp}$, we have:

$$\gamma(\mathcal{X}^\sharp) \models e \preceq \ell(e, \mathcal{X}^\sharp)$$

Remarks:

- ℓ results in a loss of precision,
- ℓ is not monotonic for \preceq .
(e.g., $\ell(\mathbb{V}/\mathbb{V}, \mathbb{V} \mapsto [1, +\infty]) = [0, 1] \times \mathbb{V} \not\preceq 1$)

Application to the octagon domain

$\begin{aligned} Y &:= [0, +\infty]; \\ T &:= [-1, 1]; \\ X &:= T \times Y \end{aligned}$

- $T \times Y$ is linearized as $[-1, 1] \times Y$,
- we can prove that $|X| \leq Y$.

Linearization application (cont.)

Application to the interval domain

$C^\sharp \llbracket V := \ell(e, \mathcal{X}^\sharp) \rrbracket \mathcal{X}^\sharp$ is always more precise than $C^\sharp \llbracket V := e \rrbracket \mathcal{X}^\sharp$
 ℓ **simplifies** symbolically variables occurring several times.

Example: $X := 2 \times V - V$, where $V \in [a, b]$:

- using vanilla intervals:

$$E^\sharp \llbracket 2 \times V - V \rrbracket (\mathcal{X}^\sharp) = 2 \times_b^\sharp [a, b] -_b^\sharp [a, b] = [2a - b, 2b - a],$$

- after linearization $\ell(2 \times V - V, \mathcal{X}^\sharp) = V$, so

$$E^\sharp \llbracket \ell(2 \times V - V, \mathcal{X}^\sharp) \rrbracket \mathcal{X}^\sharp = [a, b]$$

strictly more precise than $[2a - b, 2b - a]$ when $a \neq b$.

Bibliography

Bibliography

[Bagn02] **R. Bagnara, E. Ricci, E. Zaffanella & P. M. Hill.** *Possibly not closed convex polyhedra and the Parma Polyhedra Library.* In Proc. SAS'02, LNCS 2477, 213–229, Springer, 2002.

[Bagn03] **R. Bagnara, P. Hill, E. Ricci, E. Zaffanella.** *Precise widening operators for convex polyhedra.* In Proc. SAS'03, LNCS 2694, 337–354, Springer, 2003.

[Bagn08] **R. Bagnara, P. M. Hill & E. Zaffanella.** *An improved tight closure algorithm for integer octagonal constraints.* In Proc. VMCAI'08, LNCS 4905, 8–21, Springer, 2008.

[Berd07] **J. Berdine, A. Chawdhary, B. Cook, D. Distefano & P. O'Hearn.** *Variance analyses from invariances analyses.* In Proc. POPL'07 211–224, ACM, 2007.

[Blan03] **B. Blanchet, P. Cousot, R. Cousot, J. Feret, L. Mauborgne, A. Miné, D. Monniaux & X. Rival.** *A static analyzer for large safety-critical software.* In Proc. PLDI'03, 196–207, ACM, 2003.

Bibliography (cont.)

[Bour93] **F. Bourdoncle**. *Assertion-based debugging of imperative programs by abstract interpretation*. In Proc. ESEC'93, 501–516, Springer, 1993.

[Chen08] **L. Chen, A. Miné & P. Cousot**. *A sound floating-point polyhedra abstract domain*. In Proc. APLAS'08, LNCS 5356, 3–18, Springer, 2008.

[Cher68] **N. V. Chernikova**. *Algorithm for discovering the set of all the solutions of a linear programming problem*. In U.S.S.R. Comput. Math. and Math. Phys., 8(6):282–293, 1968.

[Cous76] **P. Cousot & R. Cousot**. *Static determination of dynamic properties of programs*. In Proc. ISP'76, Dunod, 1976.

[Cous78] **P. Cousot & N. Halbwachs**. *Automatic discovery of linear restraints among variables of a program*. In Proc. POPL'78, 84–96, ACM, 1978.

Bibliography (cont.)

[Dijk75] **E. W. Dijkstra**. *Guarded commands, nondeterminacy and formal derivation of programs*. In *Com. ACM*, 18(8):453–457, 1975.

[Dor01] **N. Dor, M. Rodeh & M. Sagiv**. *Cleanness checking of string manipulations in C programs via integer analysis*. In *Proc. SAS'01, LNCS 2126*, 194–212, Springer, 2001.

[Girb06] **S. Girbal, N. Vasilache, C. Bastoul, A. Cohen, D. Parello, M. Sigler & O. Temam**. *Semi-automatic composition of loop transformations for deep parallelism and memory hierarchies*. In *J. of Parallel Prog.*, 34(3):261–317, 2006.

[Gran89] **P. Granger**. *Static analysis of arithmetical congruences*. In *JCM*, 3(4–5):165–190, 1989.

[Gran91] **P. Granger**. *Static analysis of linear congruence equalities among variables of a program*. In *Proc. TAPSOFT'91, LNCS 49*, 169–192. Springer, 1991.

Bibliography (cont.)

- [Gran92] **P. Granger**. *Improving the results of static analyses of programs by local decreasing iterations*. In Proc. FSTTSC'92, LNCS 652, 68–79, Springer, 1992.
- [Gran97] **P. Granger**. *Static analyses of congruence properties on rational numbers*. In Proc. SAS'97, LNCS 1302, 278–292, Springer, 1997.
- [Jean09] **B. Jeannet & A. Miné**. *Apron: A library of numerical abstract domains for static analysis*. In Proc. CAV'09, LNCS 5643, 661–667, Springer, 2009, <http://apron.cri.ensmp.fr/library>.
- [Karr76] **M. Karr**. *Affine relationships among variables of a program*. In Acta Informatica, 6:133–151, 1976.
- [LeVe92] **H. Le Verge**. *A note on Chernikova's algorithm*. In Research Report 1662, INRIA Rocquencourt, 1992.

Bibliography (cont.)

[Mine06] **A. Miné.** *Field-sensitive value analysis of embedded C programs with union types and pointer arithmetics.* In Proc. LCTES'06, 54–63, ACM, 2006.

[Mine01a] **A. Miné.** *A new numerical abstract domain based on difference-bound matrices.* In Proc. PADO II, LNCS 2053, 155–172, Springer, 2001.

[Mine01b] **A. Miné.** *The octagon abstract domain.* In Proc. AST'01, 310–319, IEEE, 2001.

[Mine04] **A. Miné.** *Relational abstract domains for the detection of floating-point run-time errors.* In Proc. ESOP'04, LNCS 2986, 3–17, Springer, 2004.

[Neum04] **A. Neumaier & O. Shcherbina.** *Safe bounds in linear and mixed-integer linear programming.* In Math. Program., 99(2):283–296, 2004.

Bibliography (cont.)

[Riva05] **X. Rival**. *Understanding the origin of alarms in Astrée*. In Proc. SAS'05, LNCS 3672, 303–319, Springer, 2005.

[Schr86] **A. Schrijver**. *Theory of linear and integer programming*. In John Wiley & Sons, Inc., 1986.

[Vene02] **A. Venet**. *Nonuniform alias analysis of recursive data structures and arrays*. In Proc. SAS'02, LNCS 2477, 36–51, Springer, 2002.