Non-Relational Numerical Abstract Domains

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

Antoine Miné

Year 2023–2024

Course 3
28 September 2023

Outline

- Concrete semantics
- Abstract domains and abstract solving
- Non-relational numerical abstract domains
  - generic Cartesian abstraction
  - the sign domain(s)
  - the constant domain
  - the interval domain
  - widenings $\triangledown$ and narrowings $\vartriangle$
  - the congruence domain
- Reduced products
- Bibliography

Next week: relational abstract domains

Concrete semantics

Syntax of a toy-language

Simple numeric programs:

- fixed, finite set of variables $\mathbb{V}$
- with value in some numeric set $\mathbb{I} \stackrel{\text{def}}{=} \{\mathbb{Z}, \mathbb{Q}, \mathbb{R}\}$
- programs as control flow graphs (CFG): $(\mathcal{L}, e, x, A)$
  with nodes $\mathcal{L}$, entry $e \in \mathcal{L}$, exit $x \in \mathcal{L}$, and arcs $A \subseteq \mathcal{L} \times \texttt{com} \times \mathcal{L}$

Atomic commands:

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

Arithmetic expressions:

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

Expression semantics (reminder)

Expression semantics: $\mathsf{E}[\![\, e \,]\!] : \mathcal{E} \to \mathcal{P}(\mathbb{I})$

where $\mathcal{E} \stackrel{\text{def}}{=} \mathbb{V} \to \mathbb{I}$.

The evaluation of e in $\rho \in \mathcal{E}$ gives a set of values:

$$
\begin{array}{lcl}
\mathsf{E}[\![\, [c, c'] \,]\!]\, \rho & \stackrel{\text{def}}{=} & \{\, x \in \mathbb{I} \mid c \leq x \leq c' \,\} \\
\mathsf{E}[\![\, V \,]\!]\, \rho & \stackrel{\text{def}}{=} & \{\, \rho(V) \,\} \\
\mathsf{E}[\![\, - e \,]\!]\, \rho & \stackrel{\text{def}}{=} & \{\, -v \mid v \in \mathsf{E}[\![\, e \,]\!]\, \rho \,\} \\
\mathsf{E}[\![\, e_1 + e_2 \,]\!]\, \rho & \stackrel{\text{def}}{=} & \{\, v_1 + v_2 \mid v_1 \in \mathsf{E}[\![\, e_1 \,]\!]\, \rho, v_2 \in \mathsf{E}[\![\, e_2 \,]\!]\, \rho \,\} \\
\mathsf{E}[\![\, e_1 - e_2 \,]\!]\, \rho & \stackrel{\text{def}}{=} & \{\, v_1 - v_2 \mid v_1 \in \mathsf{E}[\![\, e_1 \,]\!]\, \rho, v_2 \in \mathsf{E}[\![\, e_2 \,]\!]\, \rho \,\} \\
\mathsf{E}[\![\, e_1 \times e_2 \,]\!]\, \rho & \stackrel{\text{def}}{=} & \{\, v_1 \times v_2 \mid v_1 \in \mathsf{E}[\![\, e_1 \,]\!]\, \rho, v_2 \in \mathsf{E}[\![\, e_2 \,]\!]\, \rho \,\} \\
\mathsf{E}[\![\, e_1 / e_2 \,]\!]\, \rho & \stackrel{\text{def}}{=} & \{\, v_1 / v_2 \mid v_1 \in \mathsf{E}[\![\, e_1 \,]\!]\, \rho, v_2 \in \mathsf{E}[\![\, e_2 \,]\!]\, \rho,\ v_2 \neq 0 \,\}
\end{array}
$$

Forward semantics: state reachability

Transfer functions: $\mathsf{C}[\![\, \texttt{com} \,]\!] : \mathcal{P}(\mathcal{E}) \to \mathcal{P}(\mathcal{E})$

- $\mathsf{C}[\![\, V \leftarrow e \,]\!]\, \mathcal{X} \stackrel{\text{def}}{=} \{\, \rho[\, V \mapsto v\,] \,|\, \rho \in \mathcal{X},\ v \in \mathsf{E}[\![\, e \,]\!]\, \rho \,\}$
- $\mathsf{C}[\![\, e \bowtie 0 \,]\!]\, \mathcal{X} \stackrel{\text{def}}{=} \{\, \rho \,|\, \rho \in \mathcal{X},\ \exists v \in \mathsf{E}[\![\, e \,]\!]\, \rho \colon v \bowtie 0 \,\}$

Fixpoint semantics: $(\mathcal{X}_\ell)_{\ell \in \mathcal{L}} : \mathcal{P}(\mathcal{E})$

$$\begin{cases} \mathcal{X}_e = \mathcal{E} & \text{(entry)} \\ \mathcal{X}_\ell = \displaystyle\bigcup_{(\ell', c, \ell) \in A} \mathsf{C}[\![\, c \,]\!]\, \mathcal{X}_{\ell'} & \text{if } \ell \neq e \end{cases}$$

Tarski's Theorem: this smallest solution exists and is unique.

$\mathcal{D} \stackrel{\text{def}}{=} (\mathcal{P}(\mathcal{E}), \subseteq, \cup, \cap, \emptyset, \mathcal{E})$ is a complete lattice,

each $M_\ell : \mathcal{X}_\ell \mapsto \displaystyle\bigcup_{(\ell', c, \ell) \in A} \mathsf{C}[\![\, c \,]\!]\, \mathcal{X}_{\ell'}$ is monotonic in $\mathcal{D}$.

$\Rightarrow$ the solution is the least fixpoint of $(M_\ell)_{\ell \in \mathcal{L}}$.

Resolution

Resolution by increasing iterations:

$$\begin{cases} \mathcal{X}_e^0 & \stackrel{\text{def}}{=} & \mathcal{E} \\ \mathcal{X}_{\ell \neq e}^0 & \stackrel{\text{def}}{=} & \emptyset \end{cases} \qquad \begin{cases} \mathcal{X}_e^{n+1} & \stackrel{\text{def}}{=} & \mathcal{E} \\ \mathcal{X}_{\ell \neq e}^{n+1} & \stackrel{\text{def}}{=} & \displaystyle\bigcup_{(\ell',c,\ell)\in A} \mathsf{C}[\![\, c \,]\!]\, \mathcal{X}_{\ell'}^n \end{cases}$$

Kleene theorem:

Iteration converges in ω iterations to a least solution, because each $\mathsf{C}[\![\, c \,]\!]$ is continuous in the CPO $\mathcal{D}$.

Backward refinement: state co-reachability

Semantics of commands: $\overleftarrow{C}[\![c]\!]: \mathcal{P}(\mathcal{E}) \to \mathcal{P}(\mathcal{E})$

- $\overleftarrow{C}[\![V \leftarrow e]\!]\mathcal{X} \stackrel{\text{def}}{=} \{ \rho \mid \exists v \in \mathrm{E}[\![e]\!]\rho: \rho[V \mapsto v] \in \mathcal{X} \}$
- $\overleftarrow{C}[\![e \bowtie 0]\!]\mathcal{X} \stackrel{\text{def}}{=} \mathrm{C}[\![e \bowtie 0]\!]\mathcal{X}$

(necessary conditions on ρ to have a successor in $\mathcal{X}$ by c)

Refinement: given:

- a solution $(\mathcal{X}_\ell)_{\ell\in\mathcal{L}}$ of the forward system
- an output criterion $\mathcal{Y}$ at exit node x

compute a least fixpoint by decreasing iterations [Bour93b]

$$\begin{cases} \mathcal{Y}_x^0 & \stackrel{\text{def}}{=} & \mathcal{X}_x \cap \mathcal{Y} \\ \mathcal{Y}_{\ell\neq x}^0 & \stackrel{\text{def}}{=} & \mathcal{X}_\ell \end{cases}$$

$$\begin{cases} \mathcal{Y}_x^{n+1} & \stackrel{\text{def}}{=} & \mathcal{X}_x \cap \mathcal{Y} \\ \mathcal{Y}_{\ell\neq x}^{n+1} & \stackrel{\text{def}}{=} & \mathcal{X}_\ell \cap \left(\bigcup_{(\ell,c,\ell')\in A} \overleftarrow{C}[\![c]\!]\mathcal{Y}_{\ell'}^n \right) \end{cases}$$

Limit to automation

We wish to perform automatic numerical invariant discovery.

Theoretical problems

- the elements of $\mathcal{P}(\mathbb{V} \to \mathbb{I})$ are not computer representable
- the transfer functions $\mathsf{C}[\![\, c \,]\!]$, $\overleftarrow{\mathsf{C}}[\![\, c \,]\!]$ are not computable
- the lattice iterations in $\mathcal{P}(\mathcal{E})$ are transfinite

Finding the best invariant is an undecidable problem

Note:

Even when $\mathbb{I}$ is finite, a concrete analysis is not tractable:

- representing elements in $\mathcal{P}(\mathbb{V} \to \mathbb{I})$ in extension is expensive
- computing $\mathsf{C}[\![\, c \,]\!]$, $\overleftarrow{\mathsf{C}}[\![\, c \,]\!]$ explicitly is expensive
- the lattice $\mathcal{P}(\mathbb{V} \to \mathbb{I})$ has a large height ($\Rightarrow$ many iterations)

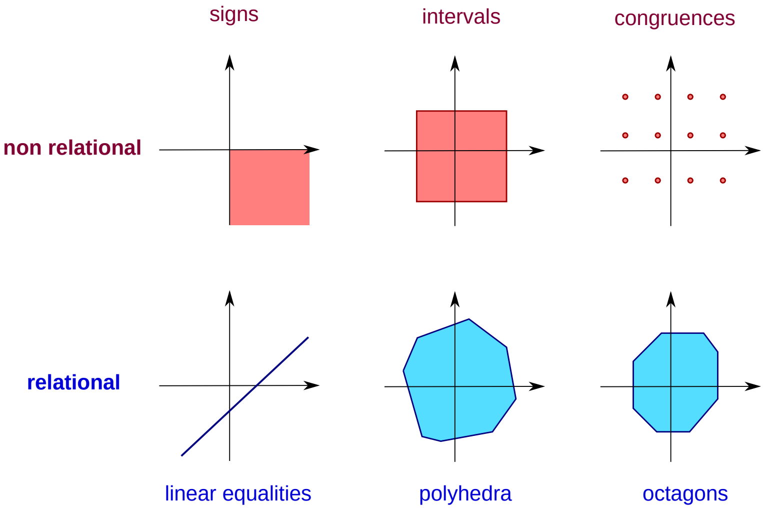
Abstractions

Numerical abstract domains

A numerical abstract domain is given by:

- a subset of $\mathcal{P}(\mathcal{E})$
  (a set of environment sets)
  together with a machine encoding,
- effective and sound abstract operators,
- an iteration strategy
  ensuring convergence in finite time.

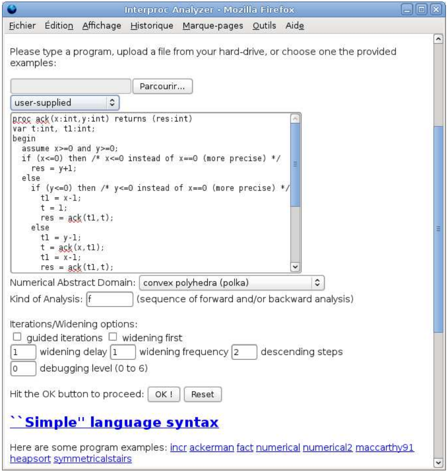
Numerical abstract domain examples



Academic implementation: Apron and Interproc

Apron: library of numerical abstractions [Jean09]

Interproc: on-line analyzer for a toy language, based on Apron



http://pop-art.inrialpes.fr/interproc/interprocweb.cgi

Numerical abstract domains (cont.)

Representation: given by

- a set $\mathcal{D}^\sharp$ of machine-representable abstract environments,
- a partial order $(\mathcal{D}^\sharp,\sqsubseteq,\bot^\sharp,\top^\sharp)$ relating the amount of information given by abstract elements,
- a concretization function γ: $\mathcal{D}^\sharp \to \mathcal{P}(\mathcal{E})$ giving a concrete meaning to each abstract element,
- an abstraction function α forming a Galois connection (α, γ) is optional.

Required algebraic properties:

- γ should be monotonic: $\mathcal{X}^\sharp \sqsubseteq \mathcal{Y}^\sharp \Longrightarrow \gamma(\mathcal{X}^\sharp) \subseteq \gamma(\mathcal{Y}^\sharp)$,
- $\gamma(\bot^\sharp) = \emptyset$,
- $\gamma(\top^\sharp) = \mathcal{E}$.

Note: γ need not be one-to-one.

Numerical abstract domains (cont.)

Abstract operators: we require:

- sound, effective, abstract transfer functions $C^\sharp[\![c]\!]$, $\overleftarrow{C}^\sharp[\![c]\!]$ for all commands c ($V \leftarrow e$, $e \bowtie 0$),
- sound, effective, abstract set operators $\cup^\sharp$, $\cap^\sharp$,
- an algorithm to decide the ordering $\sqsubseteq$.

Soundness criterion:

$F^\sharp$ is a sound abstraction of a $n-$ary operator F if:

$$\forall \mathcal{X}_1^\sharp, \ldots, \mathcal{X}_n^\sharp \in \mathcal{D}^\sharp\colon\ F(\gamma(\mathcal{X}_1^\sharp), \ldots, \gamma(\mathcal{X}_n^\sharp)) \subseteq \gamma(F^\sharp(\mathcal{X}_1^\sharp, \ldots, \mathcal{X}_n^\sharp))$$

$F^\sharp(\mathcal{X}_1^\sharp, \ldots, \mathcal{X}_n^\sharp) = \alpha(F(\gamma(\mathcal{X}_1^\sharp), \ldots, \gamma(\mathcal{X}_n^\sharp)))$ is optional.

Both semantic and algorithmic aspects.

Abstract semantics

Abstract semantic inequation system

$\mathcal{X}^\sharp : \mathcal{L} \to \mathcal{D}^\sharp$

$$\mathcal{X}^\sharp_\ell \sqsupseteq \begin{cases} \top^\sharp & \text{if } \ell = e & \text{(entry)} \\ \displaystyle\bigcup^\sharp_{(\ell',c,\ell)\in A} \mathsf{C}^\sharp[\![\, c \,]\!]\, \mathcal{X}^\sharp_{\ell'} & \text{if } \ell \neq e & \text{(abstract transfer function)} \end{cases}$$

for soundness, a post-fixpoint $\sqsupseteq$ is sufficient; a fixpoint $=$ could be too restrictive

Soundness Theorem

Any solution $(\mathcal{X}^\sharp_\ell)_{\ell\in\mathcal{L}}$ is a **sound over-approximation** of the concrete collecting semantics:

$$\forall \ell \in \mathcal{L} \colon \gamma(\mathcal{X}^\sharp_\ell) \supseteq \mathcal{X}_\ell$$

where $\mathcal{X}_\ell$ is the smallest solution of

$$\begin{cases} \mathcal{E} & \text{entry} \\ \mathcal{X}_\ell = \displaystyle\bigcup_{(\ell',c,\ell)\in A} \mathsf{C}[\![\, c \,]\!]\, \mathcal{X}_{\ell'} & \text{if } \ell \neq e \end{cases}$$

A first abstract analysis

Resolution by iteration in $\mathcal{D}^\sharp$:

$$\mathcal{X}_e^{\sharp 0} \stackrel{\text{def}}{=} \top^\sharp$$
$$\mathcal{X}_{\ell \neq e}^{\sharp 0} \stackrel{\text{def}}{=} \perp^\sharp$$
$$\mathcal{X}_\ell^{\sharp n+1} \stackrel{\text{def}}{=} \begin{cases} \top^\sharp & \text{if } \ell = e \\ \bigcup^\sharp_{(\ell',c,\ell)\in A} \mathsf{C}^\sharp[\![\, c \,]\!]\, \mathcal{X}_{\ell'}^{\sharp n} & \text{if } \ell \neq e \end{cases}$$

Iteration until stabilisation: $\forall \ell \in \mathcal{L}$: $\mathcal{X}_\ell^{\sharp \delta+1} \sqsubseteq \mathcal{X}_\ell^{\sharp \delta}$

Soundness: $\forall \ell \in \mathcal{L}$: $\mathcal{X}_\ell \subseteq \gamma(\mathcal{X}_\ell^{\sharp \delta})$

Termination: for monotonic operators on finite height lattices.

Quite restrictive !

Some improvements we will see later:

- widening operators $\triangledown$ to ensure termination in all cases
- decreasing iterations to improve precision

Also, other iteration schemes (worklist, chaotic iterations, see [Bour93a])

Backward abstract analysis

Backward refinement:

Given a forward analysis result $(\mathcal{X}^\sharp_\ell)_{\ell\in\mathcal{L}}$ and an abstract output $\mathcal{Y}^\sharp$ at x, we compute $(\mathcal{Y}^\sharp_\ell)_{\ell\in\mathcal{L}}$:

$$\mathcal{Y}^{\sharp 0}_x \stackrel{\text{def}}{=} \mathcal{X}^\sharp_x \cap^\sharp \mathcal{Y}^\sharp$$

$$\mathcal{Y}^{\sharp 0}_{\ell\neq x} \stackrel{\text{def}}{=} \mathcal{X}^\sharp_\ell$$

$$\mathcal{Y}^{\sharp n+1}_\ell \stackrel{\text{def}}{=} \begin{cases} \mathcal{X}^\sharp_x \cap^\sharp \mathcal{Y}^\sharp & \text{if } \ell = x \\ \mathcal{X}^\sharp_\ell \cap^\sharp \bigcup^\sharp_{(\ell,c,\ell')\in A} \overleftarrow{C}^\sharp[\![\, c \,]\!]\, \mathcal{Y}^{\sharp n}_{\ell'} & \text{if } \ell \neq x \end{cases}$$

Forward–backward analyses can be iterated [Bour93b].

Non-relational domains

Value abstract domains

Idea: start from an abstraction $\mathcal{B}^\sharp$ of values $\mathcal{P}(\mathbb{I})$ (representing a single variable)

Numerical value abstract domain:

$\mathcal{B}^\sharp$	abstract values, machine-representable
γ_b: $\mathcal{B}^\sharp \to \mathcal{P}(\mathbb{I})$	concretization
$\sqsubseteq_b$	partial order
$\bot_b^\sharp$, $\top_b^\sharp$	represent $\emptyset$ and $\mathbb{I}$
$\cup_b^\sharp$, $\cap_b^\sharp$	abstractions of $\cup$ and $\cap$
∇_b	extrapolation operator (introduced later, with intervals)
α_b: $\mathcal{P}(\mathbb{I}) \to \mathcal{B}^\sharp$	abstraction (optional)

Abstract arithmetic operators

Require sound abstract versions in $\mathcal{B}^\sharp$ of arithmetic operators $+$, $-$, $\times$, $/$.

Soundness conditions:

$$\begin{array}{rcl}
\{\, x \mid c \leq x \leq c' \,\} & \subseteq & \gamma_b([c, c']^\sharp_b) \\
\{\, -x \mid x \in \gamma_b(\mathcal{X}^\sharp_b) \,\} & \subseteq & \gamma_b(-^\sharp_b\, \mathcal{X}^\sharp_b) \\
\{\, x+y \mid x \in \gamma_b(\mathcal{X}^\sharp_b), y \in \gamma_b(\mathcal{Y}^\sharp_b) \,\} & \subseteq & \gamma_b(\mathcal{X}^\sharp_b +^\sharp_b \mathcal{Y}^\sharp_b) \\
& \vdots &
\end{array}$$

Using a Galois connection (α_b, γ_b):

We can define best abstract arithmetic operators:

$$\begin{array}{rcl}
[c, c']^\sharp_b & \stackrel{\text{def}}{=} & \alpha_b(\{\, x \mid c \leq x \leq c' \,\}) \\
-^\sharp_b\, \mathcal{X}^\sharp_b & \stackrel{\text{def}}{=} & \alpha_b(\{\, -x \mid x \in \gamma(\mathcal{X}^\sharp_b) \,\}) \\
\mathcal{X}^\sharp_b +^\sharp_b \mathcal{Y}^\sharp_b & \stackrel{\text{def}}{=} & \alpha_b(\{\, x+y \mid x \in \gamma(\mathcal{X}^\sharp_b),\ y \in \gamma(\mathcal{Y}^\sharp_b) \,\}) \\
\vdots & &
\end{array}$$

Derived abstract domain

Idea: associate an abstract value to each variable

$$\mathcal{D}^\sharp \stackrel{\text{def}}{=} (\mathbb{V} \to (\mathcal{B}^\sharp \setminus \{\, \bot_b^\sharp \,\})) \cup \{\, \bot^\sharp \,\}$$

- point-wise extension: $\mathcal{X}^\sharp \in \mathcal{D}^\sharp$ is a vector of elements in $\mathcal{B}^\sharp$
  (e.g. using arrays of size $|\mathbb{V}|$, or functional maps)
- smashed $\bot^\sharp$ (avoids redundant representations of $\emptyset$)

Definitions on $\mathcal{D}^\sharp$ derived from $\mathcal{B}^\sharp$:

$$\gamma(\mathcal{X}^\sharp) \stackrel{\text{def}}{=} \begin{cases} \emptyset & \text{if } \mathcal{X}^\sharp = \bot^\sharp \\ \{\, \rho \mid \forall V\colon \rho(V) \in \gamma_b(\mathcal{X}^\sharp(V)) \,\} & \text{otherwise} \end{cases}$$

$$\alpha(\mathcal{X}) \stackrel{\text{def}}{=} \begin{cases} \bot^\sharp & \text{if } \mathcal{X} = \emptyset \\ \lambda V.\alpha_b(\{\, \rho(V) \mid \rho \in \mathcal{X} \,\}) & \text{otherwise} \end{cases}$$

$$\top^\sharp \stackrel{\text{def}}{=} \lambda V.\top_b^\sharp$$

Derived abstract domain (cont.)

$$\mathcal{X}^\sharp \sqsubseteq \mathcal{Y}^\sharp \stackrel{\text{def}}{\iff} \mathcal{X}^\sharp = \bot^\sharp \vee (\mathcal{X}^\sharp, \mathcal{Y}^\sharp \neq \bot^\sharp \wedge \forall V: \mathcal{X}^\sharp(V) \sqsubseteq_b \mathcal{Y}^\sharp(V))$$

$$\mathcal{X}^\sharp \cup^\sharp \mathcal{Y}^\sharp \stackrel{\text{def}}{=} \begin{cases} \mathcal{Y}^\sharp & \text{if } \mathcal{X}^\sharp = \bot^\sharp \\ \mathcal{X}^\sharp & \text{if } \mathcal{Y}^\sharp = \bot^\sharp \\ \lambda V.\mathcal{X}^\sharp(V) \cup_b^\sharp \mathcal{Y}^\sharp(V) & \text{otherwise} \end{cases}$$

$$\mathcal{X}^\sharp \cap^\sharp \mathcal{Y}^\sharp \stackrel{\text{def}}{=} \begin{cases} \bot^\sharp & \text{if } \mathcal{X}^\sharp = \bot^\sharp \text{ or } \mathcal{Y}^\sharp = \bot^\sharp \\ \bot^\sharp & \text{if } \exists V: \mathcal{X}^\sharp(V) \cap_b^\sharp \mathcal{Y}^\sharp(V) = \bot_b^\sharp \\ \lambda V.\mathcal{X}^\sharp(V) \cap_b^\sharp \mathcal{Y}^\sharp(V) & \text{otherwise} \end{cases}$$

We will see later how to derive $\mathsf{C}^\sharp[\![\, c \,]\!]$, $\overleftarrow{\mathsf{C}}^\sharp[\![\, c \,]\!]$
from abstract arithmetic operators $+_b^\sharp$, ...

On the loss of precision: Cartesian abstraction

Non-relational domains "forget" all relationships between variables.

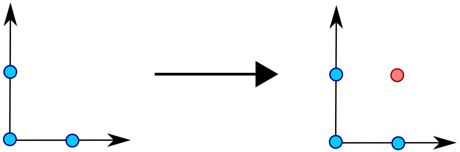
Cartesian abstraction:

Upper closure operator $\rho_c : \mathcal{P}(\mathcal{E}) \to \mathcal{P}(\mathcal{E})$

$$\rho_c(\mathcal{X}) \stackrel{\text{def}}{=} \{\, \rho \in \mathcal{E} \mid \forall V \in \mathbb{V} \colon \exists \rho' \in \mathcal{X} \colon \rho(V) = \rho'(V) \,\}$$

A domain is non-relational if $\rho \circ \gamma = \gamma$,
i.e. it cannot distinguish between $\mathcal{X}$ and $\mathcal{X}'$ if $\rho_c(\mathcal{X}) = \rho_c(\mathcal{X}')$.

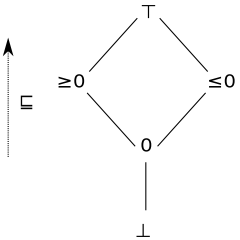
Example: $\rho_c(\{(X, Y) \mid X \in \{0, 2\}, Y \in \{0, 2\},\ X + Y \leq 2\}) = \{0, 2\} \times \{0, 2\}$.



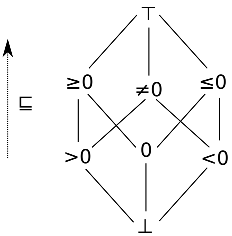
The sign domains

The sign lattices

Hasse diagram: for the lattice $(\mathcal{B}^\sharp, \sqsubseteq_b, \bot_b^\sharp, \top_b^\sharp)$



simple signs



extended signs

The extended sign domain is a refinement of the simple sign domain.

The diagram implicitly defines $\cup_b^\sharp$ and $\cap_b^\sharp$ as the least upper bound and greatest lower bound for $\sqsubseteq_b$.

Abstract operators for simple signs

Abstraction α: there is a Galois connection between $\mathcal{B}^\sharp$ and $\mathcal{P}(\mathbb{I})$:

$$\alpha_b(S) \stackrel{\text{def}}{=} \begin{cases} \bot_b^\sharp & \text{if } S = \emptyset \\ 0 & \text{if } S = \{0\} \\ \geq 0 & \text{else if } \forall s \in S\colon s \geq 0 \\ \leq 0 & \text{else if } \forall s \in S\colon s \leq 0 \\ \top_b^\sharp & \text{otherwise} \end{cases}$$

Derived abstract arithmetic operators:

$$c_b^\sharp \stackrel{\text{def}}{=} \alpha_b(\{c\}) = \begin{cases} 0 & \text{if } c = 0 \\ \leq 0 & \text{if } c < 0 \\ \geq 0 & \text{if } c > 0 \end{cases}$$

$$\begin{aligned} X^\sharp +_b^\sharp Y^\sharp &\stackrel{\text{def}}{=} \alpha_b(\{\, x + y \mid x \in \gamma_b(X^\sharp),\ y \in \gamma_b(Y^\sharp) \,\}) \\ &= \begin{cases} \bot_b^\sharp & \text{if } X \text{ or } Y^\sharp = \bot_b^\sharp \\ 0 & \text{if } X^\sharp = Y^\sharp = 0 \\ \leq 0 & \text{else if } X^\sharp \text{ and } Y^\sharp \in \{0, \leq 0\} \\ \geq 0 & \text{else if } X^\sharp \text{ and } Y^\sharp \in \{0, \geq 0\} \\ \top_b^\sharp & \text{otherwise} \end{cases} \end{aligned}$$

Generic non-relational abstract assignments

We can then define for all non-relational domains:

- an abstract semantics of expressions: $\mathsf{E}^\sharp[\![\, e \,]\!] : \mathcal{D}^\sharp \to \mathcal{B}^\sharp$

$$\mathsf{E}^\sharp[\![\, e \,]\!]\,\bot^\sharp \stackrel{\text{def}}{=} \bot_b^\sharp$$

if $\mathcal{X}^\sharp \neq \bot^\sharp$:

$$\begin{array}{lcl}
\mathsf{E}^\sharp[\![\, [c,c'] \,]\!]\,\mathcal{X}^\sharp & \stackrel{\text{def}}{=} & [c,c']_b^\sharp \\
\mathsf{E}^\sharp[\![\, V \,]\!]\,\mathcal{X}^\sharp & \stackrel{\text{def}}{=} & \mathcal{X}^\sharp(V) \\
\mathsf{E}^\sharp[\![\, -e \,]\!]\,\mathcal{X}^\sharp & \stackrel{\text{def}}{=} & -_b^\sharp\, \mathsf{E}^\sharp[\![\, e \,]\!]\,\mathcal{X}^\sharp \\
\mathsf{E}^\sharp[\![\, e_1 + e_2 \,]\!]\,\mathcal{X}^\sharp & \stackrel{\text{def}}{=} & \mathsf{E}^\sharp[\![\, e_1 \,]\!]\,\mathcal{X}^\sharp +_b^\sharp \mathsf{E}^\sharp[\![\, e_2 \,]\!]\,\mathcal{X}^\sharp \\
\vdots & &
\end{array}$$

- an abstract assignment:

$$\mathsf{C}^\sharp[\![\, V \leftarrow e \,]\!]\,\mathcal{X}^\sharp \stackrel{\text{def}}{=} \begin{cases} \bot^\sharp & \text{if } \mathcal{V}_b^\sharp = \bot_b^\sharp \\ \mathcal{X}^\sharp[V \mapsto \mathcal{V}_b^\sharp] & \text{otherwise} \end{cases}$$

where $\mathcal{V}_b^\sharp = \mathsf{E}^\sharp[\![\, e \,]\!]\,\mathcal{X}^\sharp$.

Note: in general, $\mathsf{E}^\sharp[\![\, e \,]\!]$ is less precise than $\alpha_b \circ \mathsf{E}[\![\, e \,]\!] \circ \gamma$

e.g, on intervals: $e = V - V$ and $\gamma_b(\mathcal{X}^\sharp(V)) = [0,1]$
then we get $[-1,1]$ instead of 0

Abstract tests on simple signs

Abstract test examples:

$$C^\sharp[\![\, X \leq 0 \,]\!]\,\mathcal{X}^\sharp \stackrel{\text{def}}{=} \left(\begin{cases} \mathcal{X}^\sharp[X \mapsto 0] & \text{if } \mathcal{X}^\sharp(X) \in \{0, \geq 0\} \\ \mathcal{X}^\sharp[X \mapsto \leq 0] & \text{if } \mathcal{X}^\sharp(X) \in \{\top_b^\sharp, \leq 0\} \\ \bot^\sharp & \text{otherwise} \end{cases} \right)$$

$$C^\sharp[\![\, X \leq c \,]\!]\,\mathcal{X}^\sharp \stackrel{\text{def}}{=} \left(\begin{cases} C^\sharp[\![\, X \leq 0 \,]\!]\,\mathcal{X}^\sharp & \text{if } c \leq 0 \\ \mathcal{X}^\sharp & \text{otherwise} \end{cases} \right)$$

$$C^\sharp[\![\, X \leq Y \,]\!]\,\mathcal{X}^\sharp \stackrel{\text{def}}{=}$$
$$\begin{cases} C^\sharp[\![\, X \leq 0 \,]\!]\,\mathcal{X}^\sharp & \text{if } \mathcal{X}^\sharp(Y) \in \{0, \leq 0\} \\ \mathcal{X}^\sharp & \text{otherwise} \end{cases} \quad \cap^\sharp$$
$$\begin{cases} C^\sharp[\![\, Y \geq 0 \,]\!]\,\mathcal{X}^\sharp & \text{if } \mathcal{X}^\sharp(X) \in \{0, \geq 0\} \\ \mathcal{X}^\sharp & \text{otherwise} \end{cases}$$

Other cases: $C^\sharp[\![\, expr \bowtie 0 \,]\!]\,\mathcal{X}^\sharp \stackrel{\text{def}}{=} \mathcal{X}^\sharp$ is always a sound abstraction.

We will see later a systematic way to build tests, as we did for assignments. . .

Simple sign analysis example

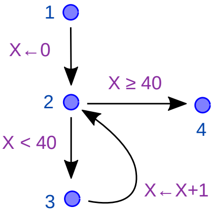
Example analysis using the simple sign domain:

```
X ← 0;
while X < 40 do
  X ← X + 1
done
```

Program

$$\begin{cases} \mathcal{X}_2^{\sharp i+1} & = & C^\sharp[\![\, X \leftarrow 0 \,]\!]\, \mathcal{X}_1^{\sharp i} \cup \\ & & C^\sharp[\![\, X \leftarrow X+1 \,]\!]\, \mathcal{X}_3^{\sharp i} \\ \mathcal{X}_3^{\sharp i+1} & = & C^\sharp[\![\, X < 40 \,]\!]\, \mathcal{X}_2^{\sharp i} \\ \mathcal{X}_4^{\sharp i+1} & = & C^\sharp[\![\, X \geq 40 \,]\!]\, \mathcal{X}_2^{\sharp i} \end{cases}$$

Iteration system



CFG

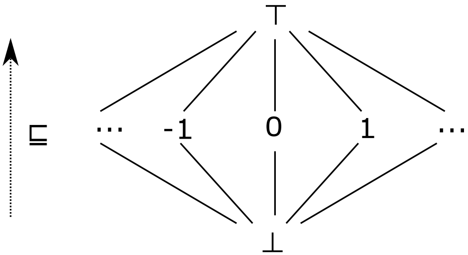
ℓ	$\mathcal{X}_\ell^{\sharp 0}$	$\mathcal{X}_\ell^{\sharp 1}$	$\mathcal{X}_\ell^{\sharp 2}$	$\mathcal{X}_\ell^{\sharp 3}$	$\mathcal{X}_\ell^{\sharp 4}$	$\mathcal{X}_\ell^{\sharp 5}$
1	$\top^\sharp$	$\top^\sharp$	$\top^\sharp$	$\top^\sharp$	$\top^\sharp$	$\top^\sharp$
2	$\perp^\sharp$	$X = 0$	$X = 0$	$X \geq 0$	$X \geq 0$	$X \geq 0$
3	$\perp^\sharp$	$\perp^\sharp$	$X = 0$	$X = 0$	$X \geq 0$	$X \geq 0$
4	$\perp^\sharp$	$\perp^\sharp$	$X = 0$	$X = 0$	$X \geq 0$	$X \geq 0$

Iterations

The constant domain

The constant lattice

Hasse diagram:



$\mathcal{B}^\sharp = \mathbb{I} \cup \{\top_b^\sharp, \bot_b^\sharp\}$

The lattice is flat but infinite.

Operations on constants

Abstraction α: there is a Galois connection:

$$\alpha_b(S) \stackrel{\text{def}}{=} \begin{cases} \perp_b^\sharp & \text{if } S = \emptyset \\ c & \text{if } S = \{c\} \\ \top_b^\sharp & \text{otherwise} \end{cases}$$

Derived abstract arithmetic operators:

$$c_b^\sharp \stackrel{\text{def}}{=} c$$

$$(X^\sharp) +_b^\sharp (Y^\sharp) \stackrel{\text{def}}{=} \begin{cases} \perp_b^\sharp & \text{if } X^\sharp \text{ or } Y^\sharp = \perp_b^\sharp \\ \top_b^\sharp & \text{else if } X^\sharp \text{ or } Y^\sharp = \top_b^\sharp \\ X^\sharp + Y^\sharp & \text{otherwise} \end{cases}$$

$$(X^\sharp) \times_b^\sharp (Y^\sharp) \stackrel{\text{def}}{=} \begin{cases} \perp_b^\sharp & \text{if } X^\sharp \text{ or } Y^\sharp = \perp_b^\sharp \\ 0 & \text{else if } X^\sharp \text{ or } Y^\sharp = 0 \\ \top_b^\sharp & \text{else if } X^\sharp \text{ or } Y^\sharp = \top_b^\sharp \\ X^\sharp \times Y^\sharp & \text{otherwise} \end{cases}$$

Operations on constants (cont.)

Abstract test examples:

$$\mathsf{C}^\sharp[\![\, X = c \,]\!]\, \mathcal{X}^\sharp \stackrel{\text{def}}{=} \begin{cases} \bot^\sharp & \text{if } \mathcal{X}^\sharp(X) \notin \{c, \top_b^\sharp\} \\ \mathcal{X}^\sharp[X \mapsto c] & \text{otherwise} \end{cases}$$

$$\mathsf{C}^\sharp[\![\, X = Y + c \,]\!]\, \mathcal{X}^\sharp \stackrel{\text{def}}{=}$$

$$\left(\begin{cases} \mathsf{C}^\sharp[\![\, X = \mathcal{X}^\sharp(Y) + c \,]\!]\, \mathcal{X}^\sharp & \text{if } \mathcal{X}^\sharp(Y) \notin \{\bot_b^\sharp, \top_b^\sharp\} \\ \mathcal{X}^\sharp & \text{otherwise} \end{cases} \right) \cap^\sharp$$

$$\left(\begin{cases} \mathsf{C}^\sharp[\![\, Y = \mathcal{X}^\sharp(X) - c \,]\!]\, \mathcal{X}^\sharp & \text{if } \mathcal{X}^\sharp(X) \notin \{\bot_b^\sharp, \top_b^\sharp\} \\ \mathcal{X}^\sharp & \text{otherwise} \end{cases} \right)$$

Constant analysis example

$\mathcal{B}^\sharp$ has finite height, the $(\mathcal{X}_\ell^{\sharp i})$ converge in finite time.

(even though $\mathcal{B}^\sharp$ is infinite. . .)

Analysis example:

```
X ← 0; Y ← 10;
while X < 100 do
  Y ← Y - 3;
  X ← X + Y; ●
  Y ← Y + 3
done
```

The constant analysis finds, at •, the invariant: $\left\{ \begin{array}{l} X = \top_b^\sharp \\ Y = 7 \end{array} \right.$

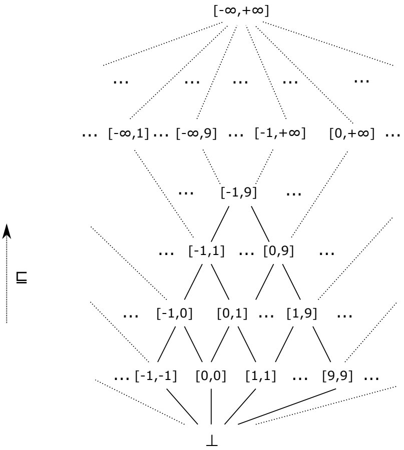
Note: the analysis can find constants that do not appear syntactically in the program.

The interval domain

The interval lattice

Introduced by [Cous76].

$$\mathcal{B}^\sharp \stackrel{\text{def}}{=} \{\, [a,b] \mid a \in \mathbb{I} \cup \{\, -\infty \,\},\ b \in \mathbb{I} \cup \{\, +\infty \,\},\ a \leq b \,\} \ \cup\ \{\, \bot^\sharp_b \,\}$$



<u>Note:</u> intervals are open at infinite bounds $+\infty$, $-\infty$.

The interval lattice (cont.)

Galois connection (α_b, γ_b):

$$\gamma_b([a,b]) \stackrel{\text{def}}{=} \{\, x \in \mathbb{I} \mid a \leq x \leq b \,\}$$

$$\alpha_b(\mathcal{X}) \stackrel{\text{def}}{=} \begin{cases} \bot_b^\sharp & \text{if } \mathcal{X} = \emptyset \\ [\min \mathcal{X}, \max \mathcal{X}] & \text{otherwise} \end{cases}$$

If $\mathbb{I} = \mathbb{Q}$, α_b is not always defined...

Partial order:

$$[a,b] \sqsubseteq_b [c,d] \stackrel{\text{def}}{\iff} a \geq c \text{ and } b \leq d$$

$$\top_b^\sharp \stackrel{\text{def}}{=} [-\infty, +\infty]$$

$$[a,b] \cup_b^\sharp [c,d] \stackrel{\text{def}}{=} [\min(a,c), \max(b,d)]$$

$$[a,b] \cap_b^\sharp [c,d] \stackrel{\text{def}}{=} \begin{cases} [\max(a,c), \min(b,d)] & \text{if } \max \leq \min \\ \bot_b^\sharp & \text{otherwise} \end{cases}$$

If $\mathbb{I} \neq \mathbb{Q}$, it is a complete lattice.

Interval abstract arithmetic operators

$$
\begin{array}{rcl}
[c,c']^\sharp_b & \stackrel{\text{def}}{=} & [c,c'] \\
-^\sharp_b\,[a,b] & \stackrel{\text{def}}{=} & [-b,-a] \\
[a,b]\ +^\sharp_b\ [c,d] & \stackrel{\text{def}}{=} & [a+c,b+d] \\
[a,b]\ -^\sharp_b\ [c,d] & \stackrel{\text{def}}{=} & [a-d,b-c] \\
[a,b]\ \times^\sharp_b\ [c,d] & \stackrel{\text{def}}{=} & [\min(ac,ad,bc,bd),\max(ac,ad,bc,bd)] \\
[a,b]\ /^\sharp_b\ [c,d] & \stackrel{\text{def}}{=} & \left\{ \begin{array}{ll}
\bot^\sharp_b & \text{if } c=d=0 \\
[\min(a/c,a/d,b/c,b/d), & \text{else if } 0 \le c \\
\ \max(a/c,a/d,b/c,b/d)] & \\
[-b,-a]/^\sharp_b[-d,-c] & \text{else if } d \le 0 \\
([a,b]/^\sharp_b[c,0]) \cup^\sharp_b ([a,b]/^\sharp_b[0,d]) & \text{otherwise}
\end{array} \right.
\end{array}
$$

where $\left| \begin{array}{l} \pm\infty \times 0 = 0, \quad 0/0 = 0, \quad \forall x\colon x/\pm\infty = 0 \\ \forall x > 0\colon x/0 = +\infty, \quad \forall x < 0\colon x/0 = -\infty \end{array} \right.$

Operators are strict: $-^\sharp_b\bot^\sharp_b = \bot^\sharp_b$, $[a,b] +^\sharp_b \bot^\sharp_b = \bot^\sharp_b$, etc.

Exactness and optimality: Example proofs

Proof: exactness of $+^\sharp_b$

$$
\begin{array}{rl}
& \{\, x+y \mid x \in \gamma_b([a,b]),\ y \in \gamma_b([c,d]) \,\} \\
= & \{\, x+y \mid a \leq x \leq b \wedge c \leq y \leq d \,\} \\
= & \{\, z \mid a+c \leq z \leq b+d \,\} \\
= & \gamma_b([a+c, b+d]) \\
= & \gamma_b([a,b]\ +^\sharp_b\ [c,d])
\end{array}
$$

Proof optimality of $\cup^\sharp_b$

$$
\begin{array}{rl}
& \alpha_b(\gamma_b([a,b]) \cup \gamma_b([c,d])) \\
= & \alpha_b(\{\, x \mid a \leq x \leq b \,\} \cup \{\, x \mid c \leq x \leq d \,\}) \\
= & \alpha_b(\{\, x \mid a \leq x \leq b \vee c \leq x \leq d \,\}) \\
= & [\min \{\, x \mid a \leq x \leq b \vee c \leq x \leq d \,\}, \max \{\, x \mid a \leq x \leq b \vee c \leq x \leq d \,\}] \\
= & [\min(a,c), \max(b,d)] \\
= & [a,b] \cup^\sharp_b [c,d]
\end{array}
$$

but $\cup^\sharp_b$ is not exact

...

Interval abstract tests (non-generic)

If $\mathcal{X}^\sharp(X) = [a, b]$ and $\mathcal{X}^\sharp(Y) = [c, d]$, we can define:

$$\mathsf{C}^\sharp[\![\, X \le c \,]\!]\, \mathcal{X}^\sharp \stackrel{\text{def}}{=} \begin{cases} \bot^\sharp & \text{if } a > c \\ \mathcal{X}^\sharp[\, X \mapsto [a, \min(b, c)]\,] & \text{otherwise} \end{cases}$$

$$\mathsf{C}^\sharp[\![\, X \le Y \,]\!]\, \mathcal{X}^\sharp \stackrel{\text{def}}{=} \begin{cases} \bot^\sharp & \text{if } a > d \\ \mathcal{X}^\sharp[\; X \mapsto [a, \min(b, d)], & \text{otherwise} \\ \qquad Y \mapsto [\max(c, a), d]\,] & \end{cases}$$

$$\mathsf{C}^\sharp[\![\, e \bowtie 0 \,]\!]\, \mathcal{X}^\sharp \stackrel{\text{def}}{=} \mathcal{X}^\sharp \quad \text{otherwise}$$

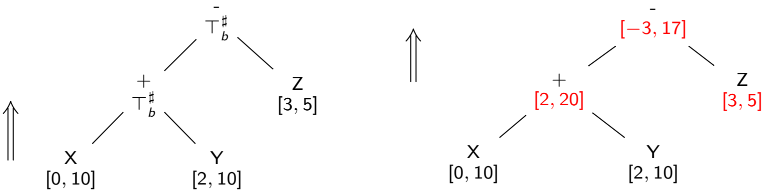
Note: fall-back operators

- $\mathsf{C}^\sharp[\![\, e \bowtie 0 \,]\!]\, \mathcal{X}^\sharp = \mathcal{X}^\sharp$ is always sound.
- $\mathsf{C}^\sharp[\![\, X \leftarrow e \,]\!]\, \mathcal{X}^\sharp = \mathcal{X}^\sharp[X \mapsto \top_b^\sharp]$ is always sound.

Generic abstract tests, step 1

Example: $C^\sharp[\![\, X + Y - Z \leq 0 \,]\!]\, \mathcal{X}^\sharp$
with $\mathcal{X}^\sharp = \{\, X \mapsto [0, 10], Y \mapsto [2, 10], Z \mapsto [3, 5] \,\}$

First step: annotate the expression tree with abstract values in $\mathcal{B}^\sharp$



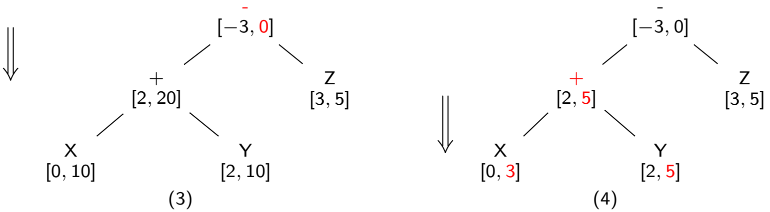
(1) (2)

Bottom-up evaluation similar to abstract expression evaluation using $+_b^\sharp$, $-_b^\sharp$, etc. but storing abstract value at each node.

Generic abstract tests, step 2

Example: $C^\sharp[\![X + Y - Z \leq 0]\!] \mathcal{X}^\sharp$
with $\mathcal{X}^\sharp = \{ X \mapsto [0, 10], Y \mapsto [2, 10], Z \mapsto [3, 5] \}$

Second step: top-down expression refinement.



- refine the root abstract value, knowing it should be negative;
- propagate refined abstract values downwards;
- values at leaf variables provide new information to store into $\mathcal{X}^\sharp$.
  $\{ X \mapsto [0, 3], Y \mapsto [2, 5], Z \mapsto [3, 5] \}$

Backward arithmetic and comparison operators

In general, we need sound backward arithmetic and comparison operators that refine their arguments given a result.

Soundness condition: for $\overleftarrow{\leq 0}^\sharp_b$, $\overleftarrow{+}^\sharp_b$, $\overleftarrow{-}^\sharp_b$, ...

$$\begin{array}{l}
\mathcal{X}^{\sharp\prime}_b = \overleftarrow{\leq 0}^\sharp_b(\mathcal{X}^\sharp_b) \implies \\
\quad \{\, x \in \gamma_b(\mathcal{X}^\sharp_b) \mid x \leq 0 \,\} \subseteq \gamma_b(\mathcal{X}^{\sharp\prime}_b) \subseteq \gamma_b(\mathcal{X}^\sharp_b)
\end{array}$$

$$\begin{array}{l}
\mathcal{X}^{\sharp\prime}_b = \overleftarrow{-}^\sharp_b(\mathcal{X}^\sharp_b, \mathcal{R}^\sharp_b) \implies \\
\quad \{\, x \mid x \in \gamma_b(\mathcal{X}^\sharp_b),\ -x \in \gamma_b(\mathcal{R}^\sharp_b) \,\} \subseteq \gamma_b(\mathcal{X}^{\sharp\prime}_b) \subseteq \gamma_b(\mathcal{X}^\sharp_b)
\end{array}$$

$$\begin{array}{l}
(\mathcal{X}^{\sharp\prime}_b, \mathcal{Y}^{\sharp\prime}_b) = \overleftarrow{+}^\sharp_b(\mathcal{X}^\sharp_b, \mathcal{Y}^\sharp_b, \mathcal{R}^\sharp_b) \implies \\
\quad \{\, x \in \gamma_b(\mathcal{X}^\sharp_b) \mid \exists y \in \gamma_b(\mathcal{Y}^\sharp_b) \colon x + y \in \gamma_b(\mathcal{R}^\sharp_b) \,\} \subseteq \gamma_b(\mathcal{X}^{\sharp\prime}_b) \subseteq \gamma_b(\mathcal{X}^\sharp_b) \\
\quad \{\, y \in \gamma_b(\mathcal{Y}^\sharp_b) \mid \exists x \in \gamma_b(\mathcal{X}^\sharp_b) \colon x + y \in \gamma_b(\mathcal{R}^\sharp_b) \,\} \subseteq \gamma_b(\mathcal{Y}^{\sharp\prime}_b) \subseteq \gamma_b(\mathcal{Y}^\sharp_b) \\
\quad \vdots
\end{array}$$

Note: best backward operators can be designed with α_b:

e.g. for $\overleftarrow{+}^\sharp_b$: $\mathcal{X}^{\sharp\prime}_b = \alpha_b(\{\, x \in \gamma_b(\mathcal{X}^\sharp_b) \mid \exists y \in \gamma_b(\mathcal{Y}^\sharp_b) \colon x + y \in \gamma_b(\mathcal{R}^\sharp_b) \,\})$

Generic backward operator construction

Synthesizing non necessarily optimal) backward arithmetic operators from forward arithmetic operators.

$$\overleftarrow{\leq 0}^\sharp_b(\mathcal{X}^\sharp_b) \stackrel{\text{def}}{=} \mathcal{X}^\sharp_b \cap^\sharp_b [-\infty, 0]^\sharp_b$$

$$\overleftarrow{-}^\sharp_b(\mathcal{X}^\sharp_b, \mathcal{R}^\sharp_b) \stackrel{\text{def}}{=} \mathcal{X}^\sharp_b \cap^\sharp_b (-^\sharp_b\, \mathcal{R}^\sharp_b)$$

(as $R = -X \implies X = -R$)

$$\overleftarrow{+}^\sharp_b(\mathcal{X}^\sharp_b, \mathcal{Y}^\sharp_b, \mathcal{R}^\sharp_b) \stackrel{\text{def}}{=} (\mathcal{X}^\sharp_b \cap^\sharp_b (\mathcal{R}^\sharp_b -^\sharp_b \mathcal{Y}^\sharp_b),\ \mathcal{Y}^\sharp_b \cap^\sharp_b (\mathcal{R}^\sharp_b -^\sharp_b \mathcal{X}^\sharp_b))$$

(as $R = X + Y \implies X = R - Y$ and $Y = R - X$)

$$\overleftarrow{-}^\sharp_b(\mathcal{X}^\sharp_b, \mathcal{Y}^\sharp_b, \mathcal{R}^\sharp_b) \stackrel{\text{def}}{=} (\mathcal{X}^\sharp_b \cap^\sharp_b (\mathcal{R}^\sharp_b +^\sharp_b \mathcal{Y}^\sharp_b),\ \mathcal{Y}^\sharp_b \cap^\sharp_b (\mathcal{X}^\sharp_b -^\sharp_b \mathcal{R}^\sharp_b))$$

$$\overleftarrow{\times}^\sharp_b(\mathcal{X}^\sharp_b, \mathcal{Y}^\sharp_b, \mathcal{R}^\sharp_b) \stackrel{\text{def}}{=} (\mathcal{X}^\sharp_b \cap^\sharp_b (\mathcal{R}^\sharp_b /^\sharp_b \mathcal{Y}^\sharp_b),\ \mathcal{Y}^\sharp_b \cap^\sharp_b (\mathcal{R}^\sharp_b /^\sharp_b \mathcal{X}^\sharp_b))$$

$$\overleftarrow{/}^\sharp_b(\mathcal{X}^\sharp_b, \mathcal{Y}^\sharp_b, \mathcal{R}^\sharp_b) \stackrel{\text{def}}{=} (\mathcal{X}^\sharp_b \cap^\sharp_b (\mathcal{S}^\sharp_b \times^\sharp_b \mathcal{Y}^\sharp_b),\ \mathcal{Y}^\sharp_b \cap^\sharp_b ((\mathcal{X}^\sharp_b /^\sharp_b \mathcal{S}^\sharp_b) \cup^\sharp_b [0,0]^\sharp_b))$$

$$\text{where } \mathcal{S}^\sharp_b = \begin{cases} \mathcal{R}^\sharp_b & \text{if } \mathbb{I} \neq \mathbb{Z} \\ \mathcal{R}^\sharp_b +^\sharp_b [-1,1]^\sharp_b & \text{if } \mathbb{I} = \mathbb{Z} \text{ (as / rounds)} \end{cases}$$

Note: $\overleftarrow{\diamond}^\sharp_b(\mathcal{X}^\sharp_b, \mathcal{Y}^\sharp_b, \mathcal{R}^\sharp_b) = (\mathcal{X}^\sharp_b, \mathcal{Y}^\sharp_b)$ is always sound (no refinement).

Application to the interval domain

Applying the generic construction to the interval domain:

$$\overleftarrow{\leq 0}^\sharp_b([a,b]) \stackrel{\text{def}}{=} \begin{cases} [a, \min(b,0)] & \text{if } a \geq 0 \\ \perp^\sharp_b & \text{otherwise} \end{cases}$$

$$\overleftarrow{-}^\sharp_b([a,b],[r,s]) \stackrel{\text{def}}{=} [a,b] \cap^\sharp_b [-s,-r]$$

$$\overleftarrow{+}^\sharp_b([a,b],[c,d],[r,s]) \stackrel{\text{def}}{=} \begin{array}{l}([a,b] \cap^\sharp_b [r-d, s-c], \\ \;[c,d] \cap^\sharp_b [r-b, s-a])\end{array}$$

...

Generic non-relational backward assignment

Abstract function: $\overleftarrow{C}^\sharp[\![V \leftarrow e]\!](\mathcal{X}^\sharp, \mathcal{R}^\sharp)$

over-approximates $\gamma(\mathcal{X}^\sharp) \cap \overleftarrow{C}[\![V \leftarrow e]\!]\gamma(\mathcal{R}^\sharp)$ given:

- an abstract pre-condition $\mathcal{X}^\sharp$ to refine,
- according to a given abstract post-condition $\mathcal{R}^\sharp$.

Algorithm: similar to the abstract test

- annotate variable leaves based on $\mathcal{X}^\sharp \cap^\sharp (\mathcal{R}^\sharp[V \mapsto \top_b^\sharp])$;
- evaluate bottom-up using forward operators $\diamond_b^\sharp$;
- intersect the root with $\mathcal{R}^\sharp(V)$;
- refine top-down using backward operators $\overleftarrow{\diamond}_b^\sharp$;
- return $\mathcal{X}^\sharp$ intersected with values at variable leaves.

Note:

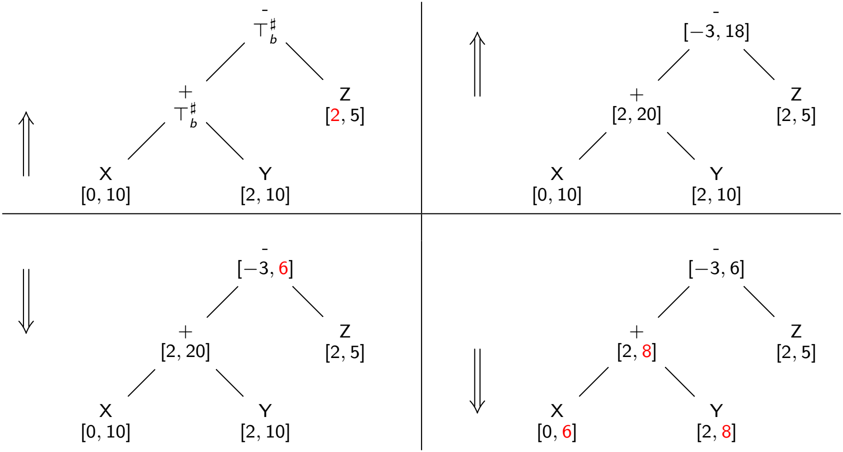
- local iterations can also be used
- fallback: $\overleftarrow{C}^\sharp[\![V \leftarrow e]\!](\mathcal{X}^\sharp, \mathcal{R}^\sharp) = \mathcal{X}^\sharp \cap^\sharp (\mathcal{R}^\sharp[V \mapsto \top_b^\sharp])$

Interval backward assignment example

Example: $\overleftarrow{C}^\sharp[\![\, X \leftarrow X + Y - Z \,]\!]\,(\mathcal{X}^\sharp, \mathcal{R}^\sharp)$

with $\mathcal{X}^\sharp = \{\, X \mapsto [0,10], Y \mapsto [2,10], Z \mapsto [1,5] \,\}$

and $\mathcal{R}^\sharp = \{\, X \mapsto [-6,6], Y \mapsto [2,10], Z \mapsto [2,6] \,\}$



Widening

$\mathcal{B}^\sharp$ has an infinite height, so does $\mathcal{D}^\sharp$.

Naive iterations $(\mathcal{X}_\ell^{\sharp i})$ may not converge in finite time.

We will use a widening operator $\triangledown$.

Definition: widening $\triangledown$

Binary operator $\mathcal{D}^\sharp \times \mathcal{D}^\sharp \to \mathcal{D}^\sharp$ ensuring

- soundness: $\gamma(\mathcal{X}^\sharp) \cup \gamma(\mathcal{Y}^\sharp) \subseteq \gamma(\mathcal{X}^\sharp \triangledown \mathcal{Y}^\sharp)$,
- termination:

  for all sequences $(\mathcal{X}_i^\sharp)$, the increasing sequence $(\mathcal{Y}_i^\sharp)$ defined by
  $$\begin{cases} \mathcal{Y}_0^\sharp & \stackrel{\text{def}}{=} \mathcal{X}_0^\sharp \\ \mathcal{Y}_{i+1}^\sharp & \stackrel{\text{def}}{=} \mathcal{Y}_i^\sharp \triangledown \mathcal{X}_{i+1}^\sharp \end{cases}$$
  is stationary, *i.e.*, $\exists i\colon \mathcal{Y}_{i+1}^\sharp = \mathcal{Y}_i^\sharp$.

Interval widening

Widening on non-relational domains:

Given a value widening $\triangledown_b \colon \mathcal{B}^\sharp \times \mathcal{B}^\sharp \to \mathcal{B}^\sharp$,
we extend it point-wise into a widening $\triangledown \colon \mathcal{D}^\sharp \times \mathcal{D}^\sharp \to \mathcal{D}^\sharp$:

$$\mathcal{X}^\sharp \mathbin{\triangledown} \mathcal{Y}^\sharp \stackrel{\text{def}}{=} \lambda V.(\mathcal{X}^\sharp(V) \mathbin{\triangledown_b} \mathcal{Y}^\sharp(V))$$

Interval widening example:

$$\begin{array}{lclcl}
\bot^\sharp & \triangledown_b & \mathcal{X}^\sharp & \stackrel{\text{def}}{=} & \mathcal{X}^\sharp \\
[a, b] & \triangledown_b & [c, d] & \stackrel{\text{def}}{=} & \left[\begin{cases} a & \text{if } a \leq c \\ -\infty & \text{otherwise} \end{cases}, \begin{cases} b & \text{if } b \geq d \\ +\infty & \text{otherwise} \end{cases} \right]
\end{array}$$

Unstable bounds are set to $\pm\infty$.

Abstract analysis with widening

Take a set $\mathcal{W} \subseteq L$ of widening points such that every CFG cycle has a point in $\mathcal{W}$.

Iteration with widening:

$$\mathcal{X}_e^{\sharp 0} \stackrel{\text{def}}{=} \top^\sharp$$
$$\mathcal{X}_{\ell \neq e}^{\sharp 0} \stackrel{\text{def}}{=} \bot^\sharp$$
$$\mathcal{X}_\ell^{\sharp n+1} \stackrel{\text{def}}{=} \begin{cases} \top^\sharp & \text{if } \ell = e \\ \bigcup^\sharp_{(\ell', c, \ell) \in A} \mathsf{C}^\sharp[\![c]\!] \mathcal{X}_{\ell'}^{\sharp n} & \text{if } \ell \notin \mathcal{W}, \ell \neq e \\ \mathcal{X}_\ell^{\sharp n} \triangledown \bigcup^\sharp_{(\ell', c, \ell) \in A} \mathsf{C}^\sharp[\![c]\!] \mathcal{X}_{\ell'}^{\sharp n} & \text{if } \ell \in \mathcal{W}, \ell \neq e \end{cases}$$

Theorem: we have:

- termination: for some δ, $\forall \ell \in \mathcal{L}$: $\mathcal{X}_\ell^{\sharp \delta+1} = \mathcal{X}_\ell^{\sharp \delta}$
- soundness: $\forall \ell \in \mathcal{L}$: $\mathcal{X}_\ell \subseteq \gamma(\mathcal{X}_\ell^{\sharp \delta})$

Note: the abstract operators $\mathsf{C}^\sharp[\![\ ]\!]$ do not have to be monotonic!

Abstract analysis with widening (proof 1/2)

Proof of soundness:

Suppose that $\forall \ell$: $\mathcal{X}_\ell^{\sharp\delta+1} = \mathcal{X}_\ell^{\sharp\delta}$.

If $\ell = e$, by definition: $\mathcal{X}_e^{\sharp\delta} = \top^\sharp$ and $\gamma(\top^\sharp) = \mathcal{E}$.

If $\ell \neq e$, $\ell \notin \mathcal{W}$, then $\mathcal{X}_\ell^{\sharp\delta} = \mathcal{X}_\ell^{\sharp\delta+1} = \cup^\sharp_{(\ell',c,\ell)\in A} \mathsf{C}^\sharp[\![c]\!] \mathcal{X}_{\ell'}^{\sharp\,\delta}$.

By soundness of $\cup^\sharp$ and $\mathsf{C}^\sharp[\![c]\!]$, $\gamma(\mathcal{X}_\ell^{\sharp\delta}) \supseteq \cup_{(\ell',c,\ell)\in A} \mathsf{C}[\![c]\!] \gamma(\mathcal{X}_{\ell'}^{\sharp\,\delta})$.

If $\ell \neq e$, $\ell \in \mathcal{W}$, then $\mathcal{X}_\ell^{\sharp\delta} = \mathcal{X}_\ell^{\sharp\delta+1} = \mathcal{X}_\ell^{\sharp\delta} \triangledown \cup^\sharp_{(\ell',c,\ell)\in A} \mathsf{C}^\sharp[\![c]\!] \mathcal{X}_{\ell'}^{\sharp\,\delta}$.

By soundness of $\triangledown$, $\gamma(\mathcal{X}_\ell^{\sharp\delta}) \supseteq \gamma(\cup^\sharp_{(\ell',c,\ell)\in A} \mathsf{C}^\sharp[\![c]\!] \mathcal{X}_{\ell'}^{\sharp\,\delta})$,

and so we also have $\gamma(\mathcal{X}_\ell^{\sharp\delta}) \supseteq \cup_{(\ell',c,\ell)\in A} \mathsf{C}[\![c]\!] \gamma(\mathcal{X}_{\ell'}^{\sharp\,\delta})$.

We have proved that $\lambda\ell.\gamma(\mathcal{X}_\ell^{\sharp\delta})$ is a postfixpoint of the concrete equation system.
Hence, it is greater than its least solution.

Abstract analysis with widening (proof 2/2)

Proof of termination:

Suppose that the iteration does not terminate in finite time.

Given a label $\ell \in \mathcal{L}$, we denote by $i_\ell^1, \ldots, i_\ell^k, \ldots$ the increasing sequence of unstable indices, i.e., such that $\forall k: \mathcal{X}^{\sharp\, i_\ell^k+1}_\ell \neq \mathcal{X}^{\sharp\, i_\ell^k}_\ell$.

As the iteration is not stable, $\forall n: \exists \ell: \mathcal{X}^{\sharp\, n}_\ell \neq \mathcal{X}^{\sharp\, n+1}_\ell$.

Hence, the sequence $(i_\ell^k)_k$ is infinite for at least one $\ell \in \mathcal{L}$.

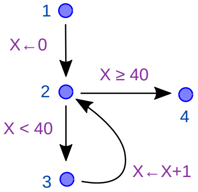
We argue that $\exists \ell \in \mathcal{W}$ such that $(i_\ell^k)_k$ is infinite as, otherwise, $N = \max \{\, i_\ell^k \mid \ell \in \mathcal{W} \,\} + |\mathcal{L}|$ is finite and satisfies: $\forall n \geq N: \forall \ell \in \mathcal{L}: \mathcal{X}^{\sharp\, n}_\ell = \mathcal{X}^{\sharp\, n+1}_\ell$, contradicting our assumption.

For such a $\ell \in \mathcal{W}$, consider the subsequence $\mathcal{Y}^\sharp_k = \mathcal{X}^{\sharp\, i_\ell^k}_\ell$ comprised of the unstable iterates of $\mathcal{X}^\sharp_\ell$.
Then $\mathcal{Y}^{\sharp\, k+1} = \mathcal{Y}^{\sharp\, k} \triangledown \mathcal{Z}^{\sharp\, k}$ for some sequence $\mathcal{Z}^{\sharp\, k}$.
The subsequence is infinite and $\forall k: \mathcal{Y}^{\sharp\, k+1} \neq \mathcal{Y}^{\sharp\, k}$, which contradicts the definition of $\triangledown$.

Hence, the iteration must terminate in finite time.

Interval analysis with widening example

Analysis example with $\mathcal{W} = \{2\}$



ℓ	$\mathcal{X}^{\sharp 0}_\ell$	$\mathcal{X}^{\sharp 1}_\ell$	$\mathcal{X}^{\sharp 2}_\ell$	$\mathcal{X}^{\sharp 3}_\ell$	$\mathcal{X}^{\sharp 4}_\ell$	$\mathcal{X}^{\sharp 5}_\ell$
1	$\top^\sharp$	$\top^\sharp$	$\top^\sharp$	$\top^\sharp$	$\top^\sharp$	$\top^\sharp$
2 $\triangledown$	$\bot^\sharp$	$= 0$	$= 0$	≥ 0	≥ 0	≥ 0
3	$\bot^\sharp$	$\bot^\sharp$	$= 0$	$= 0$	$\in [0, 39]$	$\in [0, 39]$
4	$\bot^\sharp$	$\bot^\sharp$	$\bot^\sharp$	$\bot^\sharp$	≥ 40	≥ 40

More precisely, at the widening point:

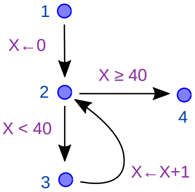
$$\begin{array}{lllllll}
\mathcal{X}^{\sharp 1}_2 & = & \bot^\sharp & \triangledown_b\,([0,0] \cup^\sharp_b \bot^\sharp) & = & \bot^\sharp & \triangledown_b\,[0,0] & = & [0,0] \\
\mathcal{X}^{\sharp 2}_2 & = & [0,0] & \triangledown_b\,([0,0] \cup^\sharp_b \bot^\sharp) & = & [0,0] & \triangledown_b\,[0,0] & = & [0,0] \\
\mathcal{X}^{\sharp 3}_2 & = & [0,0] & \triangledown_b\,([0,0] \cup^\sharp_b [1,1]) & = & [0,0] & \triangledown_b\,[0,1] & = & [0,+\infty[\\
\mathcal{X}^{\sharp 4}_2 & = & [0,+\infty] & \triangledown_b\,([0,0] \cup^\sharp_b [1,40]) & = & [0,+\infty] & \triangledown_b\,[0,40] & = & [0,+\infty]
\end{array}$$

Note that the most precise interval abstraction would be $X \in [0,40]$ at 2, and $X = 40$ at 4.

Influence of the widening point and iteration strategy

Changing $\mathcal{W}$ changes the analysis result

Example: The analysis is less precise for $\mathcal{W} = \{3\}$.



ℓ	$\mathcal{X}_\ell^{\sharp 1}$	$\mathcal{X}_\ell^{\sharp 2}$	$\mathcal{X}_\ell^{\sharp 3}$	$\mathcal{X}_\ell^{\sharp 4}$	$\mathcal{X}_\ell^{\sharp 5}$	$\mathcal{X}_\ell^{\sharp 6}$
1	$\top^\sharp$	$\top^\sharp$	$\top^\sharp$	$\top^\sharp$	$\top^\sharp$	$\top^\sharp$
2	$= 0$	$= 0$	$\in [0,1]$	$\in [0,1]$	≥ 0	≥ 0
3 $\triangledown$	$\bot^\sharp$	$= 0$	$= 0$	≥ 0	≥ 0	≥ 0
4	$\bot^\sharp$	$\bot^\sharp$	$\bot^\sharp$	$\bot^\sharp$	$\bot^\sharp$	≥ 40

Intuition: extrapolation to $+\infty$ is no longer contained by the tests.

Chaotic iterations

Changing the iteration order changes the analysis result in the presence of a widening [Bour93b].

A simple technique: Widening delay

```
V ← 0;
while 0 = [0,1] do
  if V = 0 then V ← 1 fi
done
```

V is only incremented once, from 0 to 1.

Problem:

$\triangledown$ considers V unstable and sets it to $[0, +\infty] \implies$ precision loss
$([0,0] \triangledown [0,1] = [0,+\infty])$

Solution: **delay** widening application for one or more iterations:

$$\mathcal{X}_\ell^{\sharp n+1} \stackrel{\text{def}}{=} \begin{cases} F^\sharp(\mathcal{X}_\ell^{\sharp n}) & \text{if } n < N \\ \mathcal{X}_\ell^{\sharp n} \triangledown F^\sharp(\mathcal{X}_\ell^{\sharp n}) & \text{if } n \geq N \end{cases}$$

with $N = 1$, $X_1^\sharp = [0,0] \cup^\sharp [1,1] = [0,1]$, $X_2^\sharp = [0,1] \triangledown [0,1] = [0,1] = X_1^\sharp$

(after some point, the widening must be applied continuously)

Narrowing

Using a widening makes the analysis less precise.

Some precision can be retrieved by using a narrowing $\triangle$.

Definition: narrowing $\triangle$

Binary operator $\mathcal{D}^\sharp \times \mathcal{D}^\sharp \to \mathcal{D}^\sharp$ such that:

- $\gamma(\mathcal{X}^\sharp) \cap \gamma(\mathcal{Y}^\sharp) \subseteq \gamma(\mathcal{X}^\sharp \triangle \mathcal{Y}^\sharp) \subseteq \gamma(\mathcal{X}^\sharp)$,
- for all sequences $(\mathcal{X}_i^\sharp)$, the decreasing sequence $(\mathcal{Y}_i^\sharp)$ defined by $\begin{cases} \mathcal{Y}_0^\sharp & \stackrel{\text{def}}{=} & \mathcal{X}_0^\sharp \\ \mathcal{Y}_{i+1}^\sharp & \stackrel{\text{def}}{=} & \mathcal{Y}_i^\sharp \triangle \mathcal{X}_{i+1}^\sharp \end{cases}$ is stationary.

This is not the dual of a widening!

The widening must ultimately jump above the least fixpoint (to any post-fixpoint).
The narrowing must always stay above the least fixpoint (or any fixpoint actually).

Narrowing examples

Trivial narrowing:

$\mathcal{X}^\sharp \,\triangle\, \mathcal{Y}^\sharp \stackrel{\text{def}}{=} \mathcal{X}^\sharp$ is a correct narrowing.

Finite-time intersection narrowing:

$$\mathcal{X}^{\sharp i} \,\triangle\, \mathcal{Y}^\sharp \stackrel{\text{def}}{=} \begin{cases} \mathcal{X}^{\sharp i} \cap^\sharp \mathcal{Y}^\sharp & \text{if } i \leq N \\ \mathcal{X}^{\sharp i} & \text{if } i > N \end{cases}$$

(indexed by an iteration counter i)

Interval narrowing:

$$[a,b] \,\triangle_b\, [c,d] \stackrel{\text{def}}{=} \left[\begin{cases} c & \text{if } a = -\infty \\ a & \text{otherwise} \end{cases}, \begin{cases} d & \text{if } b = +\infty \\ b & \text{otherwise} \end{cases} \right]$$

(refine only infinite bounds)

Point-wise extension to $\mathcal{D}^\sharp$: $\mathcal{X}^\sharp \,\triangle\, \mathcal{Y}^\sharp \stackrel{\text{def}}{=} \lambda V.(\mathcal{X}^\sharp(V) \,\triangle_b\, \mathcal{Y}^\sharp(V))$

Iterations with narrowing

Let $\mathcal{X}_\ell^{\sharp\delta}$ be the result after widening stabilisation, *i.e.*:

$$\mathcal{X}_\ell^{\sharp\delta} \sqsupseteq \begin{cases} \top^\sharp & \text{if } \ell = e \\ \bigcup^\sharp_{(\ell',c,\ell)\in A} \mathsf{C}^\sharp[\![\, c \,]\!]\, \mathcal{X}_{\ell'}^{\sharp\delta} & \text{if } \ell \neq e \end{cases}$$

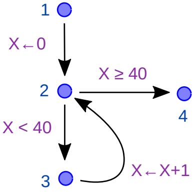
The following sequence is computed:

$$\mathcal{Y}_\ell^{\sharp 0} \stackrel{\text{def}}{=} \mathcal{X}_\ell^{\sharp\delta} \qquad \mathcal{Y}_\ell^{\sharp i+1} \stackrel{\text{def}}{=} \begin{cases} \top^\sharp & \text{if } \ell = e \\ \bigcup^\sharp_{(\ell',c,\ell)\in A} \mathsf{C}^\sharp[\![\, c \,]\!]\, \mathcal{Y}_{\ell'}^{\sharp i} & \text{if } \ell \notin \mathcal{W} \\ \mathcal{Y}_\ell^{\sharp i} \mathbin{\triangle} \bigcup^\sharp_{(\ell',c,\ell)\in A} \mathsf{C}^\sharp[\![\, c \,]\!]\, \mathcal{Y}_{\ell'}^{\sharp i} & \text{if } \ell \in \mathcal{W} \end{cases}$$

- the sequence $(\mathcal{Y}_\ell^{\sharp i})$ is decreasing and converges in finite time,
- all the $(\mathcal{Y}_\ell^{\sharp i})$ are sound abstractions of the concrete system.

Interval analysis with narrowing example

Example with $\mathcal{W} = \{2\}$



ℓ	$\mathcal{Y}_\ell^{\sharp 0}$	$\mathcal{Y}_\ell^{\sharp 1}$	$\mathcal{Y}_\ell^{\sharp 2}$	$\mathcal{Y}_\ell^{\sharp 3}$
1	$\top^\sharp$	$\top^\sharp$	$\top^\sharp$	$\top^\sharp$
2 $\triangle$	≥ 0	$\in [0,40]$	$\in [0,40]$	$\in [0,40]$
3	$\in [0,39]$	$\in [0,39]$	$\in [0,39]$	$\in [0,39]$
4	≥ 40	≥ 40	$= 40$	$= 40$

Narrowing at 2 gives:

$$\begin{array}{lclclcl}
\mathcal{Y}_2^{\sharp 1} & = & [0,+\infty] \triangle_b ([0,0] \cup_b^\sharp [1,40]) & = & [0,+\infty[\triangle_b [0,40] & = & [0,40] \\
\mathcal{Y}_2^{\sharp 2} & = & [0,40] \quad \triangle_b ([0,0] \cup_b^\sharp [1,40]) & = & [0,40] \quad \triangle_b [0,40] & = & [0,40]
\end{array}$$

Then $\mathcal{Y}_2^{\sharp 2} : X \in [0,40]$ gives $\mathcal{Y}_4^{\sharp 3} : X = 40$.

We found the most precise invariants!

Another use of narrowing: Backward analysis

Backward refinement:

Given a forward analysis result $(\mathcal{X}^\sharp_\ell)_{\ell\in\mathcal{L}}$ and an abstract output $\mathcal{Y}^\sharp$ at x, we compute $(\mathcal{Y}^\sharp_\ell)_{\ell\in\mathcal{L}}$.

$$\mathcal{Y}^{\sharp 0}_x \stackrel{\text{def}}{=} \mathcal{X}^\sharp_x \cap^\sharp \mathcal{Y}^\sharp$$

$$\mathcal{Y}^{\sharp 0}_{\ell\neq x} \stackrel{\text{def}}{=} \mathcal{X}^\sharp_\ell$$

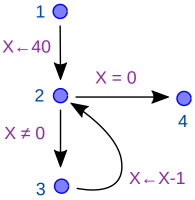
$$\mathcal{Y}^{\sharp\, n+1}_\ell \stackrel{\text{def}}{=} \begin{cases} \mathcal{X}^\sharp_x \cap^\sharp \mathcal{Y}^\sharp & \text{if } \ell = x \\ \mathcal{X}^\sharp_\ell \cap^\sharp \bigcup^\sharp_{(\ell,c,\ell')\in A} \overleftarrow{C}^\sharp[\![\, c \,]\!]\, \mathcal{Y}^{\sharp n}_{\ell'} & \text{if } \ell \notin \mathcal{W},\ \ell \neq x \\ \mathcal{Y}^{\sharp n}_\ell \mathbin{\triangle} (\mathcal{X}^\sharp_\ell \cap^\sharp \bigcup^\sharp_{(\ell,c,\ell')\in A} \overleftarrow{C}^\sharp[\![\, c \,]\!]\, \mathcal{Y}^{\sharp n}_{\ell'}) & \text{if } \ell \in \mathcal{W},\ \ell \neq x \end{cases}$$

$\triangle$ overapproximates $\cap$ while enforcing the convergence of decreasing iterations

Forward–backward analyses can be iterated [Bour93b].

Improving the interval widening

Example of imprecise analysis



ℓ	intervals with $\triangledown_b$	extended signs	intervals with $\triangledown'_b$
1	$\top^\sharp$	$\top^\sharp$	$\top^\sharp$
2 $\triangledown$	$X \leq 40$	$X \geq 0$	$X \in [0, 40]$
3	$X \leq 40$	$X > 0$	$X \in [0, 40]$
4	$X = 0$	$X = 0$	$X = 0$

The interval domain cannot prove that $X \geq 0$ at 2,
while the (less powerful) sign domain can!

(narrowing does not help)

Solution: improve the interval widening

$$[a, b] \triangledown'_b [c, d] \stackrel{\text{def}}{=} \left[\begin{cases} a & \text{if } a \leq c \\ 0 & \text{if } 0 \leq c < a \\ -\infty & \text{otherwise} \end{cases}, \begin{cases} b & \text{if } b \geq d \\ 0 & \text{if } 0 \geq b > d \\ +\infty & \text{otherwise} \end{cases} \right]$$

($\triangledown'_b$ checks the stability of 0)

Widening with thresholds

Analysis problem:

```
X ← 0;
while • 1 = 1 do
  if [0,1] = 0 then
    X ← X + 1;
    if X > 40 then X ← 0 fi
  fi
done
```

We wish to prove that $X \in [0,40]$ at $\bullet$.

- Widening at $\bullet$ finds the loop invariant $X \in [0,+\infty]$.
  $\mathcal{X}^\sharp_\bullet = [0,0] \mathbin{\triangledown_b} ([0,0] \cup^\sharp [0,1]) = [0,0] \mathbin{\triangledown_b} [0,1] = [0,+\infty[$
- Narrowing is unable to refine the invariant:
  $\mathcal{Y}^\sharp_\bullet = [0,+\infty] \mathbin{\vartriangle_b} ([0,0] \cup^\sharp [0,+\infty[) = [0,+\infty[$

  (the code that limits X is not executed at every loop iteration)

Widening with thresholds (cont.)

Solution:

Choose a finite set T of thresholds containing $+\infty$ and $-\infty$.

Definition: widening with thresholds $\triangledown_b^T$

$$[a,b] \triangledown_b^T [c,d] \stackrel{\text{def}}{=} \left[\begin{cases} a & \text{if } a \leq c \\ \max \{x \in T \mid x \leq c\} & \text{otherwise} \end{cases}, \begin{cases} b & \text{if } b \geq d \\ \min \{x \in T \mid x \geq d\} & \text{otherwise} \end{cases} \right]$$

The widening tests and stops at the first stable bound in T.

Widening with thresholds (cont.)

Applications:

- On the previous example, we find: $X \in [\,0,\ \min\{x \in T \mid x \geq 40\}\,]$.
- Useful when it is easy to find a 'good' set T.

  Example: array bound-checking
- Useful if an over-approximation of the bound is sufficient.

  Example: arithmetic overflow checking

Limitations: only works if some non-∞ bound in T is stable.

Example: with $T = \{\,5, 15\,\}$

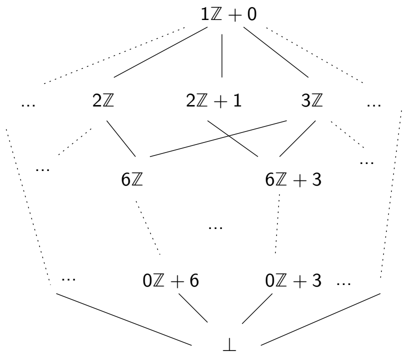
`while 1 = 1 do`
`  X ← X + 1;`
`  if X > 10 then X ← 0 fi`
`done`
	`while 1 = 1 do`
	`  X ← X + 1;`
	`  if X ≠ 10 then X ← 0 fi`
	`done`

15 is stable	no stable bound

The congruence domain

The congruence lattice

$$\mathcal{B}^\sharp \stackrel{\text{def}}{=} \{ (a\mathbb{Z} + b) \mid a \in \mathbb{N},\ b \in \mathbb{Z} \} \cup \{ \perp_b^\sharp \}$$



Introduced by Granger [Gran89].
We take $\mathbb{I} = \mathbb{Z}$.

The congruence lattice (cont.)

Concretization:

$$\gamma_b(\mathcal{X}_b^\sharp) \stackrel{\text{def}}{=} \begin{cases} \{\, ak+b \mid k \in \mathbb{Z} \,\} & \text{if } \mathcal{X}_b^\sharp = (a\mathbb{Z}+b) \\ \emptyset & \text{if } \mathcal{X}_b^\sharp = \bot_b^\sharp \end{cases}$$

Note that $\gamma(0\mathbb{Z}+b) = \{b\}$.
γ_b is not injective: $\gamma_b(2\mathbb{Z}+1) = \gamma_b(2\mathbb{Z}+3)$.

Definitions:

Given $x, x' \in \mathbb{Z}$, $y, y' \in \mathbb{N}$, we define:

- $y / y' \stackrel{\text{def}}{\iff} y$ divides y' $(\exists k \in \mathbb{N}\colon y' = ky)$ (note that $\forall y\colon y/0$)
- $x \equiv x' \,[y] \stackrel{\text{def}}{\iff} y / |x - x'|$ (in particular, $x \equiv x' \,[0] \iff x = x'$)
- $\vee$ is the LCM, extended with $y \vee 0 \stackrel{\text{def}}{=} 0 \vee y \stackrel{\text{def}}{=} 0$
- $\wedge$ is the GCD, extended with $y \wedge 0 \stackrel{\text{def}}{=} 0 \wedge y \stackrel{\text{def}}{=} y$

$(\mathbb{N}, /, \vee, \wedge, 1, 0)$ is a complete distributive lattice.

Abstract congruence operators

Complete lattice structure on $\mathcal{B}^\sharp$:

- $(a\mathbb{Z}+b) \sqsubseteq_b (a'\mathbb{Z}+b') \stackrel{\text{def}}{\iff} a'/a$ and $b \equiv b'\ [a']$
- $\top_b^\sharp \stackrel{\text{def}}{=} (1\mathbb{Z}+0)$
- $(a\mathbb{Z}+b) \cup_b^\sharp (a'\mathbb{Z}+b') \stackrel{\text{def}}{=} (a \wedge a' \wedge |b-b'|)\mathbb{Z}+b$
- $(a\mathbb{Z}+b) \cap_b^\sharp (a'\mathbb{Z}+b') \stackrel{\text{def}}{=} \begin{cases} (a \vee a')\mathbb{Z}+b'' & \text{if } b \equiv b'\ [a \wedge a'] \\ \perp_b^\sharp & \text{otherwise} \end{cases}$

  b'' such that $b'' \equiv b\ [a \vee a'] \equiv b'\ [a \vee a']$ is given by Bezout's Theorem.

Galois connection: $\alpha_b(\mathcal{X}) = \bigcup_{c \in \mathcal{X}}{}_b^\sharp\ (0\mathbb{Z}+c)$

(up to equivalence $a\mathbb{Z}+b \equiv a'\mathbb{Z}+b' \stackrel{\text{def}}{\iff} a=a' \wedge b \equiv b'\ [a]$)

Abstract congruence operators (cont.)

Arithmetic operators:

$$[c, c']^\sharp_b \stackrel{\text{def}}{=} \begin{cases} 0\mathbb{Z} + c & \text{if } c = c' \\ \top^\sharp_b & \text{otherwise} \end{cases}$$

$$-^\sharp_b (a\mathbb{Z} + b) \stackrel{\text{def}}{=} a\mathbb{Z} + (-b)$$

$$(a\mathbb{Z} + b) +^\sharp_b (a'\mathbb{Z} + b') \stackrel{\text{def}}{=} (a \wedge a')\mathbb{Z} + (b + b')$$

$$(a\mathbb{Z} + b) -^\sharp_b (a'\mathbb{Z} + b') \stackrel{\text{def}}{=} (a \wedge a')\mathbb{Z} + (b - b')$$

$$(a\mathbb{Z} + b) \times^\sharp_b (a'\mathbb{Z} + b') \stackrel{\text{def}}{=} (aa' \wedge ab' \wedge a'b)\mathbb{Z} + bb'$$

$$(a\mathbb{Z} + b) /^\sharp_b (a'\mathbb{Z} + b') \stackrel{\text{def}}{=}$$

$$\begin{cases} \perp^\sharp_b & \text{if } a'\mathbb{Z} + b' = 0\mathbb{Z} + 0 \\ (a/|b'|)\mathbb{Z} + (b/b') & \text{if } a' = 0,\ b' \neq 0,\ b'|a,\ \text{and } b'|b \\ \top^\sharp_b & \text{otherwise (not optimal)} \end{cases}$$

Abstract congruence operators (cont.)

Test operators:

$$\overleftarrow{\leq 0}^{\sharp}_{b}\,(a\mathbb{Z}+b) \quad \stackrel{\text{def}}{=} \quad \begin{cases} \bot^{\sharp}_{b} & \text{if } a=0,\ b>0 \\ a\mathbb{Z}+b & \text{otherwise} \end{cases}$$

$$\vdots$$

Note: better than the generic $\overleftarrow{\leq 0}^{\sharp}_{b}\,(\mathcal{X}^{\sharp}_{b}) \stackrel{\text{def}}{=} \mathcal{X}^{\sharp}_{b} \cap^{\sharp}_{b} [-\infty, 0]^{\sharp}_{b} = \mathcal{X}^{\sharp}_{b}$

Extrapolation operators:

- no infinite increasing chain $\implies$ no need for $\triangledown$
- infinite decreasing chains $\implies$ $\triangle$ needed

$$(a\mathbb{Z}+b)\ \triangle_b\ (a'\mathbb{Z}+b') \stackrel{\text{def}}{=} \begin{cases} a'\mathbb{Z}+b' & \text{if } a=1 \\ a\mathbb{Z}+b & \text{otherwise} \end{cases}$$

Note: $\mathcal{X}^{\sharp} \triangle \mathcal{Y}^{\sharp} \stackrel{\text{def}}{=} \mathcal{X}^{\sharp}$ is always a narrowing.

Congruence analysis example

```
X ← 0; Y ← 2;
while • X < 40 do
  X ← X + 2;
  if X < 5 then Y ← Y+18 fi;
  if X > 8 then Y ← Y-30 fi
done
```

We find, at •, the loop invariant $\begin{cases} X \in 2\mathbb{Z} \\ Y \in 6\mathbb{Z} + 2 \end{cases}$

Reduced products

Non-reduced product of domains

Product representation:

Cartesian product $\mathcal{D}^\sharp_{1\times 2}$ of $\mathcal{D}^\sharp_1$ and $\mathcal{D}^\sharp_2$:

- $\mathcal{D}^\sharp_{1\times 2} \stackrel{\text{def}}{=} \mathcal{D}^\sharp_1 \times \mathcal{D}^\sharp_2$
- $\gamma_{1\times 2}(\mathcal{X}^\sharp_1, \mathcal{X}^\sharp_2) \stackrel{\text{def}}{=} \gamma_1(\mathcal{X}^\sharp_1) \cap \gamma_2(\mathcal{X}^\sharp_2)$
- $\alpha_{1\times 2}(\mathcal{X}) \stackrel{\text{def}}{=} (\alpha_1(\mathcal{X}), \alpha_2(\mathcal{X}))$
- $(\mathcal{X}^\sharp_1, \mathcal{X}^\sharp_2) \sqsubseteq_{1\times 2} (\mathcal{Y}^\sharp_1, \mathcal{Y}^\sharp_2) \stackrel{\text{def}}{\iff} \mathcal{X}^\sharp_1 \sqsubseteq_1 \mathcal{Y}^\sharp_1$ and $\mathcal{X}^\sharp_2 \sqsubseteq_2 \mathcal{Y}^\sharp_2$

Abstract operators: performed in parallel on both components:

- $(\mathcal{X}^\sharp_1, \mathcal{X}^\sharp_2) \cup^\sharp_{1\times 2} (\mathcal{Y}^\sharp_1, \mathcal{Y}^\sharp_2) \stackrel{\text{def}}{=} (\mathcal{X}^\sharp_1 \cup^\sharp_1 \mathcal{Y}^\sharp_1, \mathcal{X}^\sharp_2 \cup^\sharp_2 \mathcal{Y}^\sharp_2)$
- $(\mathcal{X}^\sharp_1, \mathcal{X}^\sharp_2) \triangledown_{1\times 2} (\mathcal{Y}^\sharp_1, \mathcal{Y}^\sharp_2) \stackrel{\text{def}}{=} (\mathcal{X}^\sharp_1 \triangledown_1 \mathcal{Y}^\sharp_1, \mathcal{X}^\sharp_2 \triangledown_2 \mathcal{Y}^\sharp_2)$
- $\mathsf{C}^\sharp[\![c]\!]_{1\times 2}(\mathcal{X}^\sharp_1, \mathcal{X}^\sharp_2) \stackrel{\text{def}}{=} (\mathsf{C}^\sharp[\![c]\!]_1(\mathcal{X}^\sharp_1), \mathsf{C}^\sharp[\![c]\!]_2(\mathcal{X}^\sharp_2))$

Non-reduced product example

The product analysis is no more precise than two separate analyses.

Example: interval-congruence product:

```
 X ← 1;
 while X ≤ 10 do
   X ← X + 2
 done;
•if X ≥ 12 then♦ X ← 0★ fi
```

	interval	congruence	product
•	$X \in [11, 12]$	$X \equiv 1\ [2]$	$X = 11$
♦	$X = 12$	$X \equiv 1\ [2]$	$\emptyset$
★	$X = 0$	$X = 0$	$X = 0$

We cannot prove that the `if` branch is never taken!

Fully-reduced product

Definition:

Given the Galois connections (α_1, γ_1) and (α_2, γ_2) on $\mathcal{D}_1^\sharp$ and $\mathcal{D}_2^\sharp$ we define the reduction operator ρ as:

$$\rho : \mathcal{D}_{1\times 2}^\sharp \to \mathcal{D}_{1\times 2}^\sharp$$
$$\rho(\mathcal{X}_1^\sharp, \mathcal{X}_2^\sharp) \stackrel{\text{def}}{=} (\alpha_1(\gamma_1(\mathcal{X}_1^\sharp) \cap \gamma_2(\mathcal{X}_2^\sharp)), \alpha_2(\gamma_1(\mathcal{X}_1^\sharp) \cap \gamma_2(\mathcal{X}_2^\sharp)))$$

ρ propagates information between domains.

Application:

We can reduce the result of each abstract operator, except $\triangledown$:

- $(\mathcal{X}_1^\sharp, \mathcal{X}_2^\sharp) \cup_{1\times 2}^\sharp (\mathcal{Y}_1^\sharp, \mathcal{Y}_2^\sharp) \stackrel{\text{def}}{=} \rho(\mathcal{X}_1^\sharp \cup_1^\sharp \mathcal{Y}_1^\sharp, \mathcal{X}_2^\sharp \cup_2^\sharp \mathcal{Y}_2^\sharp)$,
- $\mathsf{C}^\sharp[\![c]\!]_{1\times 2}(\mathcal{X}_1^\sharp, \mathcal{X}_2^\sharp) \stackrel{\text{def}}{=} \rho(\mathsf{C}^\sharp[\![c]\!]_1(\mathcal{X}_1^\sharp), \mathsf{C}^\sharp[\![c]\!]_2(\mathcal{X}_2^\sharp))$.

We refrain from reducing after a widening $\triangledown$,
this may jeopardize the convergence (octagon domain example).

Fully-reduced product example

Reduction example: between the interval and congruence domains:

Noting: $a' \stackrel{\text{def}}{=} \min \{ x \geq a \,|\, x \equiv d \ [c] \}$

$b' \stackrel{\text{def}}{=} \max \{ x \leq b \,|\, x \equiv d \ [c] \}$

We get:

$$\rho_b([a,b], c\mathbb{Z}+d) \stackrel{\text{def}}{=} \begin{cases} (\perp_b^\sharp, \perp_b^\sharp) & \text{if } a' > b' \\ ([a',a'], 0\mathbb{Z}+a') & \text{if } a' = b' \\ ([a',b'], c\mathbb{Z}+d) & \text{if } a' < b' \end{cases}$$

extended point-wisely to ρ on $\mathcal{D}^\sharp$.

Application:

- $\rho_b([10,11],\ 2\mathbb{Z}+1) = ([11,11],\ 0\mathbb{Z}+11)$
  (proves that the branch is never taken on our example)
- $\rho_b([1,3],\ 4\mathbb{Z}) = (\perp_b^\sharp,\ \perp_b^\sharp)$

Partially-reduced product

Definition: of a partial reduction:

any function $\rho : \mathcal{D}^\sharp_{1\times 2} \to \mathcal{D}^\sharp_{1\times 2}$ such that:

$$(\mathcal{Y}_1^\sharp, \mathcal{Y}_2^\sharp) = \rho(\mathcal{X}_1^\sharp, \mathcal{X}_2^\sharp) \implies \left\{ \begin{array}{l} \gamma_{1\times 2}(\mathcal{Y}_1^\sharp, \mathcal{Y}_2^\sharp) = \gamma_{1\times 2}(\mathcal{X}_1^\sharp, \mathcal{X}_2^\sharp) \\ \gamma_1(\mathcal{Y}_1^\sharp) \subseteq \gamma_1(\mathcal{X}_1^\sharp) \\ \gamma_2(\mathcal{Y}_2^\sharp) \subseteq \gamma_2(\mathcal{X}_2^\sharp) \end{array} \right.$$

Useful when:

- there is no Galois connection, or
- a full reduction exists but is expensive to compute.

Partial reduction example:

$$\rho(\mathcal{X}_1^\sharp, \mathcal{X}_2^\sharp) \stackrel{\text{def}}{=} \left\{ \begin{array}{ll} (\perp^\sharp, \perp^\sharp) & \text{if } \mathcal{X}_1^\sharp = \perp^\sharp \text{ or } \mathcal{X}_2^\sharp = \perp^\sharp \\ (\mathcal{X}_1^\sharp, \mathcal{X}_2^\sharp) & \text{otherwise} \end{array} \right.$$

(works on all domains)

For more complex examples, see [Blan03].

Bibliography

Bibliography

[Anco10] **C. Ancourt, F. Coelho & F. Irigoin**. *A modular static analysis approach to affine loop invariants detection.* In Proc. NSAD'10, ENTCS, Elsevier, 2010.

[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.

[Bour93a] **F. Bourdoncle**. *Efficient chaotic iteration strategies with widenings.* In Proc. FMPA'93, LNCS 735, 128–141, Springer, 1993.

[Bour93b] **F. Bourdoncle**. *Assertion-based debugging of imperative programs by abstract interpretation.* In Proc. ESEC'93, 501–516, Springer, 1993.

Bibliography (cont.)

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

[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.

[Gran92] **P. Granger**. *Improving the results of static analyses of programs by local decreasing iterations.* In Proc. FSTTCSC'92, LNCS 652, 68–79, Springer, 1992.

Bibliography (cont.)

[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`.

[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.

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