Dynamic abstract interpretation

Patrick Cousot

NYU, New York
pcousot@cs.nyu.edu cs.nyu.edu/~pcousot

Tuesday, June 22th, 2021
Interval arithmetics

• In scientific computing a **real number** is represented by a **float** (floating point number) [IEEE, 1985].

• Because of **rounding errors**, the floating point computation represents an uncertain real computation.

• Ramon E. Moore [Moore, 1966; Moore, Kearfott, and Cloud, 2009] invented “**interval arithmetic**” to put bounds on rounding errors in floating point computations.

• This guarantees that the uncertain **real computation is between floating point bounds**

• We show that “**interval arithmetic**” is a **sound abstract interpretation** of the program semantics (on reals).

• Maybe the first dynamic analysis of programs.

en.wikipedia.org/wiki/Interval_arithmetic
Abstract interpretation
## Interval abstraction

<table>
<thead>
<tr>
<th>Concrete property</th>
<th>Abstraction</th>
<th>Abstract property</th>
<th>Concretization</th>
<th>Concrete overapproximation</th>
</tr>
</thead>
<tbody>
<tr>
<td>{1.0/9.0, 1.0/7.0, 1.0/3.0}</td>
<td>[0.11110, 0.33334]</td>
<td>{0.11110, ..., 1.0/9.0, ..., 0.14286 ..., ..., 1.0/7.0, ..., 1.0/3.0, ..., 0.33334}</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(\in \wp(\mathbb{R}))</td>
<td>(\in \mathcal{F} \times \mathcal{F})</td>
<td>(\in \wp(\mathbb{R}))</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Galois connection

\[ \langle \mathcal{C}, \subseteq \rangle \overset{\gamma}{\longrightarrow} \langle \mathcal{A}, \preceq \rangle \]

\[ \alpha(c) \preceq a \iff c \subseteq \gamma(a) \]

"Dynamic abstract interpretation" – 5/64 – © P. Cousot, NYU, CIMS, CS, Tuesday, June 22th, 2021
Galois retraction/insertion

\[ \langle C, \subseteq \rangle \xrightarrow{\alpha \circ \gamma} \langle A, \preceq \rangle \]

\[ \alpha(c) \preceq a \iff c \subseteq \gamma(a) \land \alpha \text{ surjective} \]
Interval abstraction
Values

- Programs compute on values $\mathbb{V}$.
- Values $\mathbb{V}$ can be the set of
  - $\mathbb{R}$ of reals.
  - $\mathbb{F}$ of floats
  - $\mathbb{P}_i$ of float intervals

For simplicity, we assume that execution stops in case of error (e.g. when dividing by zero or returning NaN).

Properties

- Properties are sets of values e.g. $\{x \in \mathbb{V} \mid x > 0\}$ is “to be positive”
- A semantics is the strongest property of executions

---

1 We include $\pm$ infinity but exclude NaN, $-0, +0$ for simplicity of the presentation, not hard to handle.
Interval abstraction

• The interval abstraction abstracts a set of numerical values, possibly unbounded, by their minimum and maximal values.

• The interval abstraction is

\[
\alpha_i(S) \triangleq [\min S, \max S]
\]

\[
\gamma_i([x, \bar{x}]) \triangleq \{z \in \mathbb{R} \mid x \leq z \leq \bar{x}\}
\]

Example 1  In interval arithmetics, a real is abstracted by the pair of enclosing floats. This is also the abstraction of the set of reals between these two floats
Abstract domain of numerical intervals

• We let the abstract domain of float intervals be

\[ P^i \triangleq \{ \emptyset \} \cup \{ [x, \overline{x}] \mid x, \overline{x} \in F \setminus \{-\infty, \infty\} \land x \leq \overline{x} \} \cup \{ [\infty, \overline{x}] \mid \overline{x} \in F \setminus \{-\infty\} \} \cup \{ [x, \infty] \mid x \in F \setminus \{\infty\} \} \]

where the empty interval \( \perp^i = \emptyset \) can be encoded by any \( [x, \overline{x}] \) with \( \overline{x} < x \) (e.g. normalized to \( [\infty, -\infty] \)).

• The intervals \( [-\infty, -\infty] \notin P^i \) and \( [\infty, \infty] \notin P^i \) are excluded.
Abstract domain of numerical intervals

- We let the abstract domain of float intervals be

\[ P^i \triangleq \{ \emptyset \} \cup \{ [x, \overline{x}] \mid x, \overline{x} \in \mathbb{F} \setminus \{-\infty, \infty\} \land x \leq \overline{x} \} \cup \{ [\infty, \overline{x}] \mid \overline{x} \in \mathbb{F} \setminus \{-\infty\} \} \cup \{ [x, \infty] \mid x \in \mathbb{F} \setminus \{\infty\} \}\]

where the empty interval \( \perp^i = \emptyset \) can be encoded by any \( [x, \overline{x}] \) with \( \overline{x} < x \) (e.g. normalized to \([\infty, -\infty]\)).

- The intervals \([-\infty, -\infty] \notin P^i \) and \([\infty, \infty] \notin P^i \) are excluded.

- The partial order \( \sqsubseteq^i \) on \( P^i \) is interval inclusion \( \perp^i \sqsubseteq^i \perp^i \sqsubseteq^i [x, \overline{x}] \sqsubseteq^i [y, \overline{y}] \) if and only if \( y \leq x \leq \overline{x} \leq \overline{y} \).

- This is a complete lattice \( \langle P^i, \sqsubseteq^i, \emptyset, [-\infty, +\infty], \sqcap^i, \sqcup^i \rangle \)
Abstract domain of numerical intervals

- We let the abstract domain of float intervals be
  \[ P^i \triangleq \emptyset \cup \{ [x, \bar{x}] \mid x, \bar{x} \in \mathbb{F} \setminus \{-\infty, \infty\} \land x \leq \bar{x} \} \cup \{ [\infty, \bar{x}] \mid \bar{x} \in \mathbb{F} \setminus \{-\infty\} \} \cup \{ [x, \infty) \mid x \in \mathbb{F} \setminus \{\infty\} \} \]

  where the empty interval \( \perp^i = \emptyset \) can be encoded by any \( [x, \bar{x}] \) with \( \bar{x} < x \) (e.g. normalized to \([\infty, -\infty] \)).

- The intervals \([-\infty, -\infty] \notin P^i \) and \([\infty, \infty] \notin P^i \) are excluded.

- The partial order \( \sqsubseteq^i \) on \( P^i \) is interval inclusion \( \perp^i \sqsubseteq^i \perp^i \sqsubseteq^i [x, \bar{x}] \sqsubseteq^i [y, \bar{y}] \) if and only if \( y \leq x \leq \bar{x} \leq \bar{y} \).

- This is a complete lattice \( \langle P^i, \sqsubseteq^i, \emptyset, [\infty, +\infty], \sqcap^i, \sqcup^i \rangle \)

- We have the Galois retraction

\[
\langle \wp(\mathbb{R}), \subseteq \rangle \xleftarrow{\gamma_i} \langle P^i, \sqsubseteq^i \rangle
\]

(2)
Soundness

• Given parameters $x \in [x, \overline{x}]$, $y \in [y, \overline{y}]$, … the interval computation of a function $f \in \mathbb{I}^n \to \mathbb{I}$ must return a sound interval $[\underline{f}, \overline{f}]$ which contains all possible results for all possible values of the parameters.

\[
\{ f(x, y, \ldots) \mid x \in [x, \overline{x}] \wedge y \in [y, \overline{y}] \wedge \ldots \} \subseteq [\underline{f}, \overline{f}]
\]
Soundness

- Given parameters $x \in [x, \overline{x}]$, $y \in [y, \overline{y}]$, … the interval computation of a function $f : \mathbb{R}^n \rightarrow \mathbb{R}$ must return a sound interval $[\underline{f}, \overline{f}]$ which contains all possible results for all possible values of the parameters.

\[
\{ f(x, y, \ldots) \mid x \in [x, \overline{x}] \land y \in [y, \overline{y}] \land \ldots \} \subseteq [\underline{f}, \overline{f}]
\]

- The smaller interval, the better! $\alpha_i$ is the best/most precise abstraction.
Soundness

- Given parameters $x \in [x, \bar{x}], y \in [y, \bar{y}], \ldots$ the interval computation of a function $f \in \mathbb{R}^n \rightarrow \mathbb{R}$ must return a sound interval $[\underline{f}, \overline{f}]$ which contains all possible results for all possible values of the parameters.

$$\left\{ f(x, y, \ldots) \mid x \in [x, \bar{x}] \land y \in [y, \bar{y}] \land \ldots \right\} \subseteq [\underline{f}, \overline{f}]$$

- The smaller interval, the better! $\alpha_i$ is the best/most precise abstraction.

- Formally, the soundness condition is

$$\alpha_i(\left\{ f(x, y, \ldots) \mid x \in \gamma_i([x, \bar{x}]) \land y \in \gamma_i([y, \bar{y}]) \land \ldots \right\}) \sqsubseteq^i [\underline{f}, \overline{f}]$$
Syntax and trace semantics of programs
Syntax

\[ x, y, \ldots \in V \]
\[ A \in A ::= 0.1 \mid x \mid A_1 - A_2 \]
\[ B \in B ::= A_1 < A_2 \mid B_1 \text{ nand } B_2 \]
\[ S \in S ::= \]
\[ \quad x = A; \]
\[ \quad ; \]
\[ \quad \text{if}(B) S \mid \text{if}(B) S \text{ else } S \]
\[ \quad \text{while}(B) S \mid \text{break}; \]
\[ \quad \{ S_l \} \]
\[ S_l \in S_l ::= S_l S \mid \epsilon \]
\[ P \in P ::= S_l \]
\[ S \in P_c ::= S \cup S_l \cup P \]

variable (\( V \) not empty)
arithmetic expression
boolean expression
statement
assignment
skip
conditionals
iteration and break
compound statement
statement list
program
program component

The float constant 0.1 is 0.000(1100)\( ^{\infty} \) in binary so has no exact finite binary representation. It is approximated as 0.10000000149011611938476562500...
Program labelling

Unique labelling to designate (sets of) program points $\ell \in \mathcal{L}$:

- at$[S]$ the program point at which execution of $S$ starts;
- after$[S]$ the program exit point after $S$, at which execution of $S$ is supposed to normally terminate, if ever;
- escape$[S]$ a boolean indicating whether or not the program component $S$ contains a `break ;` statement escaping out of that component $S$;
- break-to$[S]$ the program point at which execution of the program component $S$ goes to when a `break ;` statement escapes out of that component $S$;
- breaks-of$[S]$ the set of labels of all `break ;` statements that can escape out of $S$.
Example of program labelling

\[ S \]
\[ S_b \]

\[ S_1 \]
\[ S_2 \]
\[ S_3 \]
\[ S_4 \]
\[ S_5 \]

while \( \ell_0 (\ldots) \{ \ell_1 \ldots \ell_2 \text{ break}; \ldots \ell_3 \text{ break}; \ldots \} \) \( \ell_5 \ldots \)

\[ \ell_0 = \text{at}[S] = \text{after}[S_4] \]
\[ \ell_1 = \text{at}[S_1] = \text{at}[S_b] \]
\[ \ell_2 = \text{at}[S_2] = \text{after}[S_1] \]
\[ \ell_3 = \text{at}[S_3] \]
\[ \ell_5 = \text{at}[S_5] = \text{break-to}[S_b] = \text{after}[S] \]
\[ \text{escape}[S_b] = \text{tt} \]
\[ \text{breaks-of}[S_b] = \{\ell_2, \ell_3\} \]
\[ \text{escape}[S] = \text{ff} \]
\[ \text{in}[S_b] = \{\ell_1, \ldots, \ell_2, \ldots, \ell_3, \ldots\} \]
\[ \text{in}[S] = \text{labx}[S_b] = \{\ell_0, \ell_1, \ldots, \ell_2, \ldots, \ell_3, \ldots\} \]
\[ \text{labx}[S] = \{\ell_0, \ell_1, \ldots, \ell_2, \ldots, \ell_3, \ldots, \ell_5\} \]
Prefix traces

- Program label: $\ell \in \mathcal{L}$ (locates next step to be executed in the program)
- Environment: $\rho \in \mathcal{E}_V \triangleq V \rightarrow V$ assigns values $\rho(x) \in V$ to variables $x \in V$.
- State: $\langle \ell, \rho \rangle \in S_V \triangleq (\mathcal{L} \times \mathcal{E}_V)$
- Trace: finite or infinite sequence $\pi \in S_V^{+\infty}$ of states
- Example: $\langle \ell_1, \{x \rightarrow 1\}\rangle \langle \ell_2, \{x \rightarrow 2\}\rangle \langle \ell_4, \{x \rightarrow 2\}\rangle$
- Trace concatenation: $\triangleright$

\[
\begin{align*}
\pi_1 \sigma_1 & \triangleright \sigma_2 \pi_2 & \text{undefined if } \sigma_1 \neq \sigma_2 \\
\pi_1 & \triangleright \sigma_2 \pi_2 & \triangleq \pi_1 & \text{if } \pi_1 \in S_V^+ \text{ is infinite} \\
\pi_1 \sigma_1 & \triangleright \sigma_1 \pi_2 & \triangleq \pi_1 \sigma_1 \pi_2 & \text{if } \pi_1 \in T^+ \text{ is finite}
\end{align*}
\]

- In pattern matching, we sometimes need the empty trace $\emptyset$. For example if $\sigma \pi \sigma' = \sigma$ then $\pi = \emptyset$ and $\sigma = \sigma'$.
Evaluation of expressions

- Evaluation of an arithmetic expression (parameterized by $\mathbb{V} = \mathbb{R}$ or $\mathbb{V} = \mathbb{F}$, later intervals)

\[
\mathbb{A}_\mathbb{V}[0.1] \rho \triangleq 0.1_{\mathbb{V}} \\
\mathbb{A}_\mathbb{V}[x] \rho \triangleq \rho(x) \\
\mathbb{A}_\mathbb{V}[A_1 - A_2] \rho \triangleq \mathbb{A}_\mathbb{V}[A_1] \rho - \mathbb{A}_\mathbb{V}[A_2] \rho
\]

- For example, $-_{\mathbb{F}}$ is the difference found on IEEE-754 machines and must take rounding mode (and the machine specificities [Monniaux, 2008]) into account.
Evaluation of expressions

• Evaluation of an arithmetic expression (parameterized by $\mathbb{V} = \mathbb{R}$ or $\mathbb{V} = \mathbb{F}$, later intervals)

$$\mathbb{A}_{\mathbb{V}}[0.1] \rho \triangleq 0.1_\mathbb{V}$$

$$\mathbb{A}_{\mathbb{V}}[x] \rho \triangleq \rho(x)$$

$$\mathbb{A}_{\mathbb{V}}[A_1 - A_2] \rho \triangleq \mathbb{A}_{\mathbb{V}}[A_1] \rho - \mathbb{V} \mathbb{A}_{\mathbb{V}}[A_2] \rho$$

• For example $-\mathbb{F}$ is the difference found on IEEE-754 machines and must take rounding mode (and the machine specificities [Monniaux, 2008]) into account.

• Evaluation of a Boolean expression ($\mathbb{B} \triangleq \{\text{tt, ff}\}$):

$$\mathbb{B}_{\mathbb{V}}[A_1 < A_2] \rho \triangleq \mathbb{A}_{\mathbb{V}}[A_1] \rho < \mathbb{A}_{\mathbb{V}}[A_2] \rho$$

$$\mathbb{B}_{\mathbb{V}}[B_1 \text{ nand } B_2] \rho \triangleq \mathbb{B}_{\mathbb{V}}[B_1] \rho \uparrow \mathbb{B}_{\mathbb{V}}[B_2] \rho$$

where $<$ is strictly less than on reals and floats while $\uparrow$ is the “not and” boolean operator.
Prefix trace semantics

- A prefix trace describes the beginning of a computation
- Assignment $S ::= \ell \ x = A ;$ (where at$[S] = \ell$)

$$S^*_V[S] = \{\langle \ell, \rho \rangle \mid \rho \in Ev\} \cup \{\langle \ell, \rho \rangle \langle \text{after}[S], \rho[x \leftarrow \mathcal{A}_V[A] \rho] \rangle \mid \rho \in Ev\}$$

(2)
Prefix trace semantics (cont’d)

• Break statement $S ::= \ell \text{ break } ;$ (where $\text{at}[S] = \ell)$

$$S^*_V[S] \triangleq \{\langle \ell, \rho \rangle \mid \rho \in E_v\} \cup \{\langle \ell, \rho \rangle \langle\text{break-to}[S], \rho\rangle \mid \rho \in E_v\}$$

(3)
Prefix trace semantics (cont’d)

• Conditional statement $S ::= \text{if } \ell (B) S_t$ (where $at[S] = \ell$)

$$S^*_V[S] \triangleq \{⟨\ell, \rho⟩ | \rho \in E_v\}$$

$$\cup \{⟨\ell, \rho⟩⟨\text{after}[S], \rho⟩ | B_V[B] \rho = \text{ff}\}$$

$$\cup \{⟨\ell, \rho⟩⟨\text{at}[S_t], \rho⟩\pi | B_V[B] \rho = \text{tt} \land ⟨\text{at}[S_t], \rho⟩\pi \in S^*_V[S_t]\}$$

• If the conditional statement $S$ is inside an iteration statement, and $S_t$ has a break, the execution goes on at the $\text{break-to}[S]$ after the iteration.
Prefix trace semantics (cont’d)

• Statement list \( S l ::= S l' \ S \) (where \( \text{at}[S] = \text{after}[S l'] \))

\[
S_v^*[Sl] \triangleq S_v^*[Sl'] \cup S_v^*[S] \quad (7)
\]
\[
S \sim S' \triangleq \{ \pi \circledast \pi' \mid \pi \in S \land \pi' \in S' \land \pi \circledast \pi' \text{ is well-defined} \} \quad (3)
\]

• \( \pi' \in S_v^*[S] \) starts at \( \text{at}[S] = \text{after}[S l'] \) so, by def. \( \sim \), the trace \( \pi \in S_v^*[Sl'] \) must terminate to be able to go on with \( S \).
Prefix trace semantics (cont’d)

- Empty statement list $S_\ell ::= \epsilon$ (where $at[S_\ell] \triangleq after[S_\ell]$)

$$S_\ell^* \triangleq \{\langle at[S_\ell], \rho \rangle \mid \rho \in Ev\}$$
Prefix trace semantics (cont’d)

- Iteration statement $S ::= \textbf{while } \ell (B) \ S_b$ (where $\text{at}[S] = \ell$)

$$S^*_\mathcal{V}[\textbf{while } \ell (B) \ S_b] = \text{lfp}^\mathcal{V} \mathcal{F}^*_\mathcal{V}[\textbf{while } \ell (B) \ S_b]$$ (8)

$$\mathcal{F}^*_\mathcal{V}[\textbf{while } \ell (B) \ S_b] X \triangleq \{(\ell, \rho) \mid \rho \in \mathcal{E}_\mathcal{V}\}$$ (a)

$$\cup \{\pi_2\langle \ell', \rho \rangle \langle \text{after}[S], \rho \rangle \mid \pi_2\langle \ell', \rho \rangle \in X \land \mathcal{B}_\mathcal{V}[B] \rho = \mathsf{ff} \land \ell' = \ell\}$$ (b)

$$\cup \{\pi_2\langle \ell', \rho \rangle \langle \text{at}[S_b], \rho \rangle \cdot \pi_3 \mid \pi_2\langle \ell', \rho \rangle \in X \land \mathcal{B}_\mathcal{V}[B] \rho = \mathsf{tt} \land$$

$$\langle \text{at}[S_b], \rho \rangle \cdot \pi_3 \in S^*_\mathcal{V}[S_b] \land \ell' = \ell\}$$ (c)

(a) either the execution observation stop at$[\text{while } \ell (B) \ S_b] = \ell$, or

(b) after a number of iterations, control is back to $\ell$, the test is false, and the loop is exited, or

(c) after a number of iterations, control is back to $\ell$, the test is true, and the loop body is executed

(This includes the termination of the loop body after$[S_b] = \text{at}[\text{while } \ell (B) \ S_b] = \ell$)
Maximal trace semantics

- Maximal trace semantics
  
  \[ \mathcal{S}_V^+[S] \triangleq \{ \pi(\ell, \rho) \in \mathcal{S}_V^*[S] \mid (\ell = \text{after}[S]) \lor (\text{escape}[S] \land \ell = \text{break-to}[S]) \} \]

  \[ \mathcal{S}_V^\infty[S] \triangleq \lim(\mathcal{S}_V^*[S]) \]

- Limit

  \[ \lim \mathcal{T} \triangleq \{ \pi \in \mathcal{T}^\infty \mid \forall n \in \mathbb{N} . \pi[0..n] \in \mathcal{T} \}. \]
Objective
Objective

- We have defined the value semantics $S_V^*$ of the language (its executions on reals are not implementable/too costly to implement\(^2\))
Objective

- We have defined the value semantics $S^*_V$ of the language (its executions on reals are not implementable/too costly to implement$^2$)

- Next, we define the interval abstraction $\hat{\alpha}_{Pi}$ of a value semantics (replacing reals by float intervals)

---

$^2$e.g. using Bill Gosper’s exact algorithms for continued fraction arithmetic.
Objective

- We have defined the value semantics $\mathcal{S}_V^*$ of the language (its executions on reals are not implementable/too costly to implement\(^2\))

- Next, we define the interval abstraction $\alpha^P_i$ of a value semantics (replacing reals by float intervals)

- The best float interval semantics of the value semantics is $\alpha^P_i(\mathcal{S}_V^*)$ (its executions on interval float abstractions of reals are not implementable)

\(^2\)e.g. using Bill Gosper's exact algorithms for continued fraction arithmetic.
Objective

• We have defined the value semantics $S^*_V$ of the language (its executions on reals are not implementable/too costly to implement\(^2\))

• Next, we define the interval abstraction $\alpha^P_i$ of a value semantics (replacing reals by float intervals)

• The best float interval semantics of the value semantics is $\alpha^P_i(S^*_V)$ (its executions on interval float abstractions of reals are not implementable)

• We define a sound over-approximation partial order $\mathcal{E}^i$ of interval semantics (with larger intervals)

---

\(^2\) e.g. using Bill Gosper's exact algorithms for continued fraction arithmetic.
Objective

• We have defined the **value semantics** $S^*_V$ of the language (its executions on reals are not implementable/too costly to implement\(^2\))

• Next, we define the **interval abstraction** $\alpha^{P_i}$ of a value semantics (replacing reals by float intervals)

• The **best float interval semantics** of the value semantics is $\alpha^{P_i}(S^*_V)$ (its executions on interval float abstractions of reals are not implementable)

• We define a **sound over-approximation** partial order $\sqsubseteq^i$ of interval semantics (with larger intervals)

• Next, we calculate the **interval semantics** $S^*_{P_i}$ of the language (executions on float intervals)

\(^2\)e.g. using Bill Gosper’s exact algorithms for continued fraction arithmetic.
Objective

- We have defined the value semantics $S^*_V$ of the language (its executions on reals are not implementable/too costly to implement\(^2\))

- Next, we define the interval abstraction $\alpha^{P_i}$ of a value semantics (replacing reals by float intervals)

- The best float interval semantics of the value semantics is $\alpha^{P_i}(S^*_V)$ (its executions on interval float abstractions of reals are not implementable)

- We define a sound over-approximation partial order $\preceq^i$ of interval semantics (with larger intervals)

- Next, we calculate the interval semantics $S^*_{P_i}$ of the language (executions on float intervals)

- By construction $\alpha^{P_i}(S^*_V) \preceq^i S^*_{P_i}$, so the interval semantics is a sound abstraction of the value semantics

\(^2\) e.g. using Bill Gosper's exact algorithms for continued fraction arithmetic.
Interval arithmetics
How real computations are performed?

- **Floating point arithmetics**: floating point number representing an uncertain real $x$

- **Interval arithmetics**: the computation is performed with the two ends of a float interval $[x, y]$ with $x \in [x, y]$.

- This is an abstraction of a trace semantics on reals.

- Handling tests:
  - real computation: only one branch taken
  - float computation: only one branch taken, but could be the wrong one
  - interval computation: one or both alternatives taken (hence one real trace can be abstracted into interval several traces).
How real computations are performed?

- **Floating point arithmetics**: floating point number representing an uncertain real $x$

- **Interval arithmetics**: the computation is performed with the two ends of a float interval $[x, \bar{x}]$ with $x \in [x, \bar{x}]$.

- This is an abstraction of a trace semantics on reals
How real computations are performed?

- **Floating point arithmetics**: floating point number representing an uncertain real $x$
- **Interval arithmetics**: the computation is performed with the two ends of a float interval $[x, \overline{x}]$ with $x \in [x, \overline{x}]$.
- This is an abstraction of a trace semantics on reals
- Handling tests:
  - real computation: only one branch taken
  - float computation: only one branch taken, but could be the wrong one
  - interval computation: one or both alternatives taken (hence one real trace can be abstracted into interval several traces).
Constants

• If the program contains a constant $c$, its interval is $[c, c]$.
• However, the compilation may introduce an error i.e. rounding error for a float that must be taken into account.
• For example, the decimal 0.1 is $0.000(1100)\infty$ in binary so has no exact binary representation on finitely many bits.
Addition and substraction

\[ [x, \bar{x}] \oplus^i \emptyset = \emptyset \oplus^i [x, \bar{x}] = [x, \bar{x}] \oplus^i \emptyset = \emptyset \oplus^i [x, \bar{x}] = \emptyset \]

\[ [x, \bar{x}] \oplus^i [y, \bar{y}] = [x + y, \bar{x} + \bar{y}] \]

\[ [x, \bar{x}] \ominus^i [y, \bar{y}] = [x - \bar{y}, \bar{x} - y] \]

\[ \ominus^i [x, \bar{x}] = [-\bar{x}, -x] \]

- We assume that \(-\infty + -\infty = -\infty, -\infty + z = -\infty, \infty + z = \infty, \) and \(\infty + \infty = \infty\) for any \(z \in I\).
- For example, \([10, \infty] \ominus^i [-\infty, 5] = [10 - 5, \infty - (-\infty)] = [5, \infty]\).
- For floating point numbers, the lower bound is rounded towards \(-\infty\) and the upper bound towards \(\infty\).
- This implies that the computed value is always included in the concretization of the interval value.
- Interval arithmetic is imprecise does not identify different occurrences of the same variable.
Multiplication

\[
[x, \bar{x}] \otimes^i \emptyset = \emptyset \otimes^i [x, \bar{x}] = \emptyset
\]

\[
[x, \bar{x}] \otimes^i [y, \bar{y}] = [\min(xy, x\bar{y}, \bar{x}y, \bar{x}\bar{y}), \max(xy, x\bar{y}, \bar{x}y, \bar{x}\bar{y})]
\]

which reduces to \([xy, \bar{x}\bar{y}]\) when the lower bounds \(x\) and \(y\) are greater than zero.
Algebraic properties

- The interval operations have some of the usual algebraic properties of arithmetic operations

\[(x \oplus^i y) \oplus^i z = x \oplus^i (y \oplus^i k z)\]  \quad \text{associativity}

\[(x \ominus^i y) \ominus^i z = x \ominus^i (y \ominus^i z)\]  \quad \text{commutativity}

\[x \oplus^i y = y \oplus^i x\]

\[x \ominus^i y = y \ominus^i x\]

\[x \oplus^i [0,0] = x\]  \quad \text{neutral element}

\[x \ominus^i [1,1] = x\]

- However distributivity does not hold. We have

\[x \otimes^i (y \oplus^i z) \subseteq^i (x \otimes^i y) \oplus^i (x \otimes^i z)\]  \quad \text{subdistributivity}
Conditions

• Although when computing with $I$ only one branch of a conditional will be taken, interval computation with $P^i$ may have to take both.

• This gives, in the worst-case, an exponential number of cases to consider.
Conditions

• Although when computing with $I$ only one branch of a conditional will be taken, interval computation with $P^i$ may have to take both.

• This gives, in the worst-case, an exponential number of cases to consider.

• In most interval arithmetic libraries, this case raises an exception that stops execution, which is a further coarse abstraction of the abstract semantics presented here.

• See e.g. www.boost.org/doc/libs/1_74_0/libs/numeric/interval/doc/interval.htm and www.boost.org/doc/libs/1_74_0/libs/numeric/interval/doc/comparisons.htm.
Conditions (cont’d)

- The boolean comparison operators $x \circledast y$ take two intervals for $x$ and $y$ and return two intervals for $x$ and $y$ such that the comparison may hold (and cannot hold outside these intervals).

$$\begin{align*}
[x, \bar{x}] \circledast_i [y, \bar{y}] &\triangleq \langle \emptyset, \emptyset \rangle \quad \text{if } \bar{x} < \bar{y} \text{ or } \bar{y} < x \\
&\triangleq \langle \left[\max(x, y), \min(\bar{x}, \bar{y})\right], \left[\max(x, y), \min(\bar{x}, \bar{y})\right]\rangle \quad \text{otherwise} \\
[x, \bar{x}] \circledast_i [y, \bar{y}] &\triangleq \langle \emptyset, \emptyset \rangle \quad \text{if } x \geq \bar{y} \\
&\triangleq \langle [x, \min(\bar{x}, \bar{y})], [\max(x, y), \bar{y}] \rangle \quad \text{otherwise, } 1 \neq \mathbb{Z} \\
&\triangleq \langle [x, \min(\bar{x}, \bar{y} - 1)], [\max(x + 1, y), \bar{y}] \rangle \quad \text{otherwise, } 1 = \mathbb{Z}
\end{align*}$$
Float interval abstraction
Float notations

- $\downarrow x$ (which can be $-\infty$) is the largest float smaller than or equal to $x \in \mathbb{R}$ (or $\downarrow x = x$ for $x \in \mathbb{F}$).
- $x\lceil\rceil$ (which can be $\infty$) is the smallest float greater than or equal to $x \in \mathbb{R}$ (or $x\lceil\rceil = x$ for $x \in \mathbb{F}$).
- $\downarrow x$ is the largest floating-point number strictly less than $x \in \mathbb{F}$ (which can be $-\infty$).
- $x\lceil\rceil$ is the smallest floating-point number strictly larger than $x \in \mathbb{F}$ (which can be $\infty$).

We assume

\[
\downarrow x - \mathbb{F} y \downarrow \leq \downarrow (x - \sqrt{y}) \quad (\forall \text{ is } \mathbb{R} \text{ or } \mathbb{F}) \quad (12)
\]

\[
x\lceil\rceil - \mathbb{F} \downarrow y \geq (x - \sqrt{y})\lceil\rceil
\]

\[
(x \in [x, \bar{x}] \land y \in [y, \bar{y}] \land x < y) \Rightarrow (x \in [x, \min(x, \bar{y})] \land y \in [\max(x, \underline{y}), \bar{y}])
\]
Incorrect machine implementations


• For example [Monniaux, 2008, Sect. 6.1.2], we could have

\[(x \in [x, \bar{x}] \land y \in [y, \bar{y}] \land x < y) \Rightarrow (x \in [x, \min(\bar{x}, \bar{y})] \land y \in [\max(\bar{x}, y), \bar{y}])\]  

(13.bis)

en.wikipedia.org/wiki/Pentium_FDIV_bug
Float interval abstraction

\[ \alpha_{Pi}(x) \triangleq [\ell | x, x'] \]  
\[ \gamma_{Pi}([x, \bar{x}]) \triangleq \{ x \in \mathbb{R} \mid x \leq x \leq \bar{x} \} \]  
\[ \dot{\alpha}_{Pi}(\rho) \triangleq x \in \mathcal{V} \mapsto \alpha_{Pi}(\rho(x)) \]  
\[ \dot{\gamma}_{Pi}(\bar{\rho}) \triangleq \{ \rho \in \mathcal{V} \to \mathbb{R} \mid \forall x \in \mathcal{V}. \rho(x) \in \gamma_{Pi}(\bar{\rho}(x)) \} \]  
\[ \ddot{\alpha}_{Pi}(\langle \pi_1 \ldots \pi_n \ldots \rangle) \triangleq \ddot{\alpha}_{Pi}(\pi_1) \ldots \ddot{\alpha}_{Pi}(\pi_n) \ldots \]  
\[ \ddot{\gamma}_{Pi}(\langle \pi_1 \ldots \pi_n \ldots \rangle) \triangleq \{ \pi_1 \ldots \pi_n \ldots \mid |\pi| = |\overline{\pi}| \land \forall i = 1, \ldots, n, \ldots. \pi_i \in \gamma_{Pi}^{\prime}(\pi_i) \} \]  

Because the floats are a subset of the reals, we can use \( \alpha_{Pi} \) to abstract both real and float traces (i.e. \( \mathcal{V} \) be \( \mathbb{R} \) or \( \mathbb{F} \)).

\[ \langle \rho(S^+_{\infty}), \subseteq \rangle \xrightarrow{\gamma_{Pi}} \langle \rho(S^+_{\infty}), \subseteq \rangle \]  

© P. Cousot, NYU, CIMS, CS, Tuesday, June 22th, 2021
\( \subseteq \) is correct by inadequate for approximation in the abstract

- Program: \( \ell_1 \ x = x - x ; \ell_2 \)

- Concrete semantics:

\[
\Pi = \{ \langle \ell_1, x = 0.1_R \rangle \langle \ell_2, x = 0.0_R \rangle, \quad \langle \ell_1, x = -0.1_R \rangle \langle \ell_2, x = 0.0_R \rangle \}
\]

- Sound abstract semantics on floats:

\[
\overline{\Pi}_1 = \{ \langle \ell_1, x = [0.09, 0.11] \rangle \langle \ell_2, x = [0.00, 0.00] \rangle, \quad \langle \ell_1, x = [-0.11, -0.09] \rangle \langle \ell_2, x = [0.00, 0.00] \rangle \}
\]

\[
\overline{\Pi}_2 = \{ \langle \ell_1, x = [-0.11, 0.11] \rangle \langle \ell_2, x = [-0.02, 0.20] \rangle \}
\]

\( \Pi \subseteq \gamma^p_i (\overline{\Pi}_1) \)

\( \Pi \subseteq \gamma^p_i (\overline{\Pi}_2) \)

- \( \overline{\Pi}_1 \) and \( \overline{\Pi}_2 \) are not comparable as abstract elements of \( \langle \varphi(\mathbb{S}^+_p), \subseteq \rangle \)

- So \( \subseteq \) does not allow over approximating \( \overline{\Pi}_1 \) by \( \overline{\Pi}_2 \)!
Sound over-approximation in the concrete

• Concrete semantics:

\[ \Pi = \{ \langle \ell_1, x = 0.1_\mathbb{R} \rangle, \langle \ell_2, x = 0.0_\mathbb{R} \rangle, \langle \ell_1, x = -0.1_\mathbb{R} \rangle, \langle \ell_2, x = 0.0_\mathbb{R} \rangle \} \]

• Sound abstract semantics on floats:

\[ \Pi_1 = \{ \langle \ell_1, x = [0.09, 0.11] \rangle, \langle \ell_2, x = [0.00, 0.00] \rangle, \langle \ell_1, x = [-0.11, -0.09] \rangle, \langle \ell_2, x = [0.00, 0.00] \rangle \} \]

\[ \Pi_2 = \{ \langle \ell_1, x = [-0.11, 0.11] \rangle, \langle \ell_2, x = [-0.02, 0.20] \rangle \} \]

\[ \Pi \subseteq \gamma^{\Pi_1}(\Pi_1) \]

\[ \Pi \subseteq \gamma^{\Pi_2}(\Pi_2) \]

• By comparison in the concrete, \( \Pi_1 \) is more precise than \( \Pi_2 \), written \( \Pi_1 \preceq \Pi_2 \)

\[ \Pi_1 \preceq \Pi_2 \iff \gamma^{\Pi_1}(\Pi_1) \subseteq \gamma^{\Pi_2}(\Pi_2) \]

\[ = \forall \pi_1 \in \Pi_1, \forall \pi \in \gamma^{\Pi_1}(\pi_1) \exists \pi_2 \in \Pi_2, \pi \in \gamma^{\Pi_2}(\pi_2) \]
Sound over-approximation in the abstract

- We express \( \preceq^i \) in the abstract, without referring to the concretization \( \preceq^i \).
- We define \( \Pi \preceq^i \Pi' \) so that the traces of \( \Pi' \) have the same control as the traces of \( \Pi \) but intervals are larger (and \( \Pi' \) may contain extra traces due to the imprecision of interval tests).
- \( \preceq^i \) is Hoare preorder [Winskel, 1983] on sets of traces.

\[
[x, \bar{x}] \preceq^i [y, \bar{y}] \triangleq y \leq x \leq \bar{x} \leq \bar{y} \quad (18)
\]

\[
\rho \preceq^i \rho' \triangleq \forall x \in \mathcal{V}. \rho(x) \preceq^i \rho'(x)
\]

\[
\langle \ell, \rho \rangle \preceq^i \langle \ell', \rho' \rangle \triangleq (\ell = \ell') \land (\rho \preceq^i \rho')
\]

\[
\pi \preceq^i \pi' \triangleq (|\pi| = |\pi'|) \land (\forall i \in [0, |\pi|]. \pi_i \preceq^i \pi'_i)
\]

\[
\Pi \preceq^i \Pi' \triangleq \forall \pi \in \Pi. \exists \pi' \in \Pi'. \pi \preceq^i \pi'
\]
Sound over-approximation in the abstract

- We express $\preceq_i$ in the abstract, without referring to the concretization $\vec{\gamma}^{E_i}$.
- We define $\vec{\Pi} \trianglelefteq_i \vec{\Pi}'$ so that the traces of $\vec{\Pi}'$ have the same control as the traces of $\vec{\Pi}$ but intervals are larger (and $\vec{\Pi}'$ may contain extra traces due to the imprecision of interval tests).
- $\trianglelefteq_i$ is Hoare preorder [Winskel, 1983] on sets of traces.

\[
\begin{align*}
[x, \bar{x}] \preceq_i [y, \bar{y}] & \triangleq y \leq x \leq \bar{x} \leq \bar{y} \\
\rho \trianglelefteq_i \rho' & \triangleq \forall x \in \mathcal{V} . \rho(x) \preceq_i \rho'(x) \\
\langle \ell, \rho \rangle \trianglelefteq_i \langle \ell', \rho' \rangle & \triangleq (\ell = \ell') \wedge (\rho \trianglelefteq_i \rho') \\
\pi \trianglelefteq_i \pi' & \triangleq (|\pi| = |\pi'|) \wedge (\forall i \in [0,|\pi|]. \pi_i \trianglelefteq_i \pi_i') \\
\vec{\Pi} \trianglelefteq_i \vec{\Pi}' & \triangleq \forall \pi \in \vec{\Pi} . \exists \pi' \in \vec{\Pi}' . \pi \trianglelefteq_i \pi'
\end{align*}
\]

Lemma 6 \((\vec{\Pi} \trianglelefteq_i \vec{\Pi}') \Rightarrow (\vec{\Pi} \trianglelefteq_i \vec{\Pi}')\).\hfill \square
Sound over-approximation in the abstract (cont’d)

• Strictly weaker

• Example:

\[
\begin{align*}
\Pi_1 & = \{ \langle \ell_1, x = [0.0, 1.0] \rangle, \\
& \quad \langle \ell_1, x = [1.0, 2.0] \rangle \}\ \\
\Pi_2 & = \{ \langle \ell_1, x = [0.0, 0.5] \rangle, \\
& \quad \langle \ell_1, x = [0.5, 2.0] \rangle \} \\
\end{align*}
\]

• \( \Pi_1 \subseteq^i \Pi_2 \) (same concrete traces)

• \( \Pi_1 \not\subseteq^i \Pi_2 \) (no inclusion of abstract traces)

• \( \Pi_2 \not\subseteq^i \Pi_1 \)
Soundness and calculational design

• Value (real/float) concrete semantics: $S^*_V[S]$

• Interval abstract semantics: $S^*_P[S]$

• **Soundness:** all value (real/float) traces are included in the interval traces:

$$\alpha^P_i(S^*_V[S]) \subseteq^i S^*_P[S]$$

$$\Rightarrow \alpha^P_i(S^*_V[S]) \subseteq^i S^*_P[S]$$

\text{\{lemma 6\}}

$$\Rightarrow \gamma^P_i(\alpha^P_i(S^*_V[S])) \subseteq \gamma^P_i(S^*_P[S])$$

\text{\{def. $\subseteq^i$\}}

$$\Rightarrow S^*_V[S] \subseteq \gamma^P_i(S^*_P[S])$$  \( \text{Galois connection } \langle \phi(S^+_V\infty), \subseteq \rangle \underbrace{\longleftrightarrow}_{\alpha^P_i} \langle \phi(S^+_P\infty), \subseteq \rangle, \ (15) \}$$
Soundness and calculational design

• Value (real/float) concrete semantics: $S^*_V[S]$

• Interval abstract semantics: $S^*_P[S]$

• **Soundness**: all value (real/float) traces are included in the interval traces:

  \[
  \alpha^P_i (S^*_V[S]) \subseteq i S^*_P[S] \\
  \Rightarrow \alpha^P_i (S^*_V[S]) \subseteq i S^*_P[S] \quad \{ \text{lemma 6} \} \\
  \Rightarrow \gamma^P_i (\alpha^P_i (S^*_V[S])) \subseteq \gamma^P_i (S^*_P[S]) \quad \{ \text{def. } \subseteq i \} \\
  \Rightarrow S^*_V[S] \subseteq \gamma^P_i (S^*_P[S]) \quad \{ \text{Galois connection } \langle \rho(S^+_V), \subseteq \rangle \leftrightarrow \gamma^P_i \langle \rho(S^+_P), \subseteq \rangle, \ (15) \}
  \]

• **Calculational design**:
  • Calculate $\alpha^P_i (S^*_V[S])$
  • Over approximate by $\subseteq i$ to eliminate all concrete operations
Calculational design of the float interval trace semantics
Float interval abstraction of an arithmetic expression semantics

• Let \( V \) be \( \mathbb{R} \) or \( \mathbb{F} \).

\[
\mathcal{A}_V^i [1] \rho \triangleq 1^i_p \\
\mathcal{A}_V^i [0.1] \rho \triangleq 0.1^i_p \quad \text{where } 0.1^i_p \triangleq [\uparrow 0.1, 0.1 \downarrow] \\
\mathcal{A}_V^i [x] \rho \triangleq \rho(x) \\
\mathcal{A}_V^i [A_1 - A_2] \rho \triangleq \mathcal{A}_V^i [A_1] \rho \ominus \mathcal{A}_V^i [A_2] \rho \quad \text{where } [x, \bar{x}] \ominus [y, \bar{y}] \triangleq [x - \bar{y}, \bar{x} - y]
\]

(with rounding towards \(-\infty/\infty\)) is such that

\[
\alpha^i_v (\mathcal{A}_V [A] \rho) \sqsubseteq^i \mathcal{A}_V^i [A] \alpha^i_v (\rho). \tag{21}
\]

• \( \mathcal{A}_V^i [A] \) is \( \preceq^i \)-increasing (but does not preserves least upper bounds).
Proof

\[ \alpha^i(\mathcal{A}_V[0.1] \rho) \]
\[ = \alpha^i(0.1 \psi) \]
\[ = [\|0.1 \psi, 0.1 \psi]\] \\(\triangleq\) \(\mathcal{A}_I[0.1](\alpha^i(\rho))\)

\[ \triangleq \alpha^i(\mathcal{A}_V[0.1] \rho) \]
\[ = \alpha^i(\rho(x)) \]
\[ = \alpha^i(\rho)(x) \]
\[ \triangleq \mathcal{A}_I[x](\alpha^i(\rho)) \]

\[ \triangleq \alpha^i(\mathcal{A}_V[A_1 - A_2] \rho) \]
\[ = \alpha^i(\mathcal{A}_V[A_1] \rho - \mathcal{A}_V[A_2] \rho) \]
\[ = [\|\mathcal{A}_V[A_1] \rho - \mathcal{A}_V[A_2] \rho, \mathcal{A}_V[A_1] \rho - \mathcal{A}_V[A_2] \rho]\]
\[ \triangleq i \mathcal{A}_I[A_1] \alpha^i(\rho) \text{ and } \{y, \bar{y}\} = \mathcal{A}_I[A_2] \alpha^i(\rho) \text{ in } [\bar{x} - F y, x - F y]\]

\[ \triangleq \mathcal{A}_I[A_1] \alpha^i(\rho) - \mathcal{A}_I[A_2] \alpha^i(\rho) \]
\[ \triangleq \mathcal{A}_I[A_1 - A_2] \alpha^i(\rho) \]

\[ \alpha^i(\{\rho(x) - \rho(y) \mid \rho \in \gamma(\bar{\rho})\}) \triangleq [\bar{x} - F y, x - F y] \]

Approximation:

\[ \alpha^i(\{\rho(x) - \rho(y) \mid \rho \in \gamma(\bar{\rho})\}) \triangleq [\bar{y}, \bar{x}] \]

\[ \alpha^i(\{\rho(x) - \rho(y) \mid \rho \in \gamma(\bar{\rho})\}) \triangleq [\bar{x} - F y, x - F y] \]

\[ \alpha^i(\{\rho(x) - \rho(y) \mid \rho \in \gamma(\bar{\rho})\}) \triangleq [\bar{y}, \bar{x}] \]

\[ \alpha^i(\{\rho(x) - \rho(y) \mid \rho \in \gamma(\bar{\rho})\}) \triangleq [\bar{x} - F y, x - F y] \]
Float interval abstraction of an assignment semantics

- $S ::= \ell \ x = A$

- Concrete semantics on reals ($V = \mathbb{R}$) or float ($V = \mathbb{F}$):

$$S^*_V[S] = \{ \langle \ell, \rho \rangle | \rho \in E_v^V \} \cup \{ \langle \ell, \rho \rangle \langle \text{after}[S], \rho[x \leftarrow A_v^V[A]\rho] \rangle | \rho \in E_v^V \}$$

- Abstract semantics on intervals ($V = \mathbb{P}^i$)

$$S^*_P^i[S] \triangleq \{ \langle \ell, \bar{\rho} \rangle | \bar{\rho} \in E_v^{P^i} \} \cup \{ \langle \ell, \bar{\rho} \rangle \langle \text{after}[S], \bar{\rho}[x \leftarrow A_v^{P^i}[A] \bar{\rho}] \rangle | \bar{\rho} \in E_v^{P^i} \}$$

- Same traces except for computing on intervals rather than values
Proof

We can now abstract the semantics of real ($V=R$) or float ($V=F$) assignments by float intervals.

\[
\alpha^I([x = A ;]) = \{\alpha^I(\pi) \mid \pi \in [x = A ;] \} \\
= \{\alpha^I(\pi) \mid \pi \in \{\ell, \rho\} \setminus \rho \in E_v \} \cup \{\ell, \rho\} \langle \text{after}[S], \rho[x \leftarrow A \pi] \rangle \mid \rho \in E_v \} \\
= \{\ell, \rho\} \langle [S], \alpha^I(z \leftarrow A \rho) \rangle \mid \rho \in E_v \} \\
= \{\ell, \rho\} \langle [S], \alpha^I(z \leftarrow A \rho) \rangle \mid \rho \in E_v \} \\
= \{\ell, \rho\} \langle [S], \alpha^I(z \leftarrow A \rho) \rangle \mid \rho \in E_v \} \\
= \{\ell, \rho\} \langle [S], \alpha^I(z \leftarrow A \rho) \rangle \mid \rho \in E_v \} \\
= \{\ell, \rho\} \langle [S], \alpha^I(z \leftarrow A \rho) \rangle \mid \rho \in E_v \} \\
= \{\ell, \rho\} \langle [S], \alpha^I(z \leftarrow A \rho) \rangle \mid \rho \in E_v \} \\
= \{\ell, \rho\} \langle [S], \alpha^I(z \leftarrow A \rho) \rangle \mid \rho \in E_v \} \\
= \{\ell, \rho\} \langle [S], \alpha^I(z \leftarrow A \rho) \rangle \mid \rho \in E_v \} \\
\Delta S^I [x = A ;]
\]

Approximation $\mathcal{E}^i$:

- value $\mathcal{A}_V$ to interval arithmetic $\mathcal{A}_I$
- value to interval environments
Float interval abstraction of an arithmetic expression semantics

- A test is true or false for $V = R$ and $V = F$
- For intervals a test is imprecise (e.g. $<$ is handled as $\leq$), may yield a split, and overlap.
- The abstract interpretation $B_{pl}[B]$ of a boolean expression $B$ is defined such that

\[
\text{let } \langle \rho_{tt}, \rho_{ff} \rangle = B_{pl}[B] \alpha^{pl}(\rho) \text{ in } \]
\[
\alpha^{pl}(\rho) \sqsubseteq^{i} \rho_{tt} \quad \text{if } B_{V}[B] \rho = \texttt{tt}
\]
\[
\alpha^{pl}(\rho) \sqsubseteq^{i} \rho_{ff} \quad \text{if } B_{V}[B] \rho = \texttt{ff}
\]
\[
\text{and } \langle \rho_{tt}, \rho_{ff} \rangle = B_{pl}[B] \rho \Rightarrow (\rho_{tt} \sqsubseteq^{i} \rho \land \rho_{ff} \sqsubseteq^{i} \rho)
\]

- No concrete state passing the test is omitted in the abstract, and
- The postcondition $\rho_{tt}$ or $\rho_{ff}$ is stronger than the precondition $\rho$ (no side effects)
Float interval abstraction of a conditional

• Conditional statement $S ::= \text{if} \ \ell \ (B) \ S_t$ (where $\text{at}[S] = \ell$)\(^3\)

$$S\overset{*}{\text{P}}[S] \triangleq \{\langle \ell, \bar{\rho} \rangle | \bar{\rho} \in \mathbb{V}_{P,i} \} \cup \{\langle \ell, \bar{\rho} \rangle \langle \text{after}[S], \bar{\rho}_{\text{ff}} \rangle | \exists \bar{\rho}_{\text{tt}}. \mathcal{B}_{\text{P}}[B] \bar{\rho} = \langle \bar{\rho}_{\text{tt}}, \bar{\rho}_{\text{ff}} \rangle \land \bar{\rho}_{\text{ff}} \neq \emptyset \}$$

$$\cup \{\langle \ell, \bar{\rho} \rangle \langle \text{at}[S_t], \bar{\rho}_{\text{tt}} \rangle \pi | \exists \bar{\rho}_{\text{ff}}. \mathcal{B}_{\text{P}}[B] \bar{\rho} = \langle \bar{\rho}_{\text{tt}}, \bar{\rho}_{\text{ff}} \rangle \land \bar{\rho}_{\text{tt}} \neq \emptyset \land$$

$$\langle \text{at}[S_t], \bar{\rho}_{\text{tt}} \rangle \pi \in S\overset{*}{\text{P}}[S_t] \}$$

• Most libraries raise an error exception in case of split (or chose only one branch).

$$S\overset{*}{\text{P}}[S] \triangleq \ldots$$

$$\cup \{\langle \ell, \bar{\rho} \rangle \pi | \exists \bar{\rho}_{\text{tt}}, \bar{\rho}_{\text{ff}}. \mathcal{B}_{\text{P}}[B] \bar{\rho} = \langle \bar{\rho}_{\text{tt}}, \bar{\rho}_{\text{ff}} \rangle \land \bar{\rho}_{\text{ff}} \neq \emptyset \land$$

$$\bar{\rho}_{\text{tt}} \neq \emptyset \land \pi \in S_{P,i}^{+\infty} \}$$

\(^3\)We assume that $\text{const}^i(\emptyset) = \emptyset$.
Float interval abstraction of an iteration

- Iteration statement $S ::= \textbf{while } \ell (B) S_b$ (where $\text{at}\llbracket S \rrbracket = \ell$)

\[
S^*_{Pi} \llbracket \textbf{while } \ell (B) S_b \rrbracket = \text{lfp} \subseteq \mathcal{F}^*_{Pi} \llbracket \textbf{while } \ell (B) S_b \rrbracket
\]

(8bis)

\[
\mathcal{F}^*_{Pi} \llbracket \textbf{while } \ell (B) S_b \rrbracket X \triangleq \left\{ (\ell, \rho) \mid \rho \in \mathbb{E}_{Pi} \right\}
\]

\[
\cup \left\{ \pi_2 \langle \ell', \rho \rangle \langle \text{after}\llbracket S \rrbracket, \rho_{ff} \rangle \mid \pi_2 \langle \ell', \rho \rangle \in X \land \exists \overline{\rho}_{tt}. \mathcal{B}_{Pi} \llbracket B \rrbracket \overline{\rho} = \langle \overline{\rho}_{tt}, \overline{\rho}_{ff} \rangle \land \rho_{ff} \neq \emptyset \land \ell' = \ell \right\}
\]

\[
\cup \left\{ \pi_2 \langle \ell', \rho \rangle \langle \text{at}\llbracket S_b \rrbracket, \rho_{tt} \rangle \pi_3 \mid \pi_2 \langle \ell', \rho \rangle \in X \land \exists \rho_{ff}. \mathcal{B}_{Pi} \llbracket B \rrbracket \overline{\rho} = \langle \overline{\rho}_{tt}, \overline{\rho}_{ff} \rangle \land \rho_{ff} \neq \emptyset \land
\langle \text{at}\llbracket S_b \rrbracket, \rho_{tt} \rangle \pi_3 \in S^*_{Pi} \llbracket S_b \rrbracket \land \ell' = \ell \right\}
\]
Abstraction to a transition system
Abstraction to a transition system

- Abstraction to a transition system
  \[
  \alpha_t(\pi) \triangleq \{ \langle \sigma_1, \sigma_2 \rangle \mid \exists \pi_1, \pi_2. \pi = \pi_1 \sigma_1 \sigma_2 \pi_2 \}\]
  \[
  \alpha_T(\Pi) \triangleq \bigcup_{\pi \in \Pi} \alpha_t(\pi)
  \]

- Provides a small-step operational semantics of the program (specifying an implementation)
- We used trace abstractions so there is no need for [bi-]simulations, etc. in the proof of correctness of the implementation
Improving precision
Affine arithmetic

• Interval arithmetic is imprecise.
• For example, if $x \in [1, 4]$ then $x - x \in [1 - 4, 4 - 1] = [-3, 3]$ instead of $[0, 0]$.
• The problem as that the arguments of functions cannot be correlated by a cartesian abstraction.
• So we have to independently take into consideration all possible values of variables within their interval of variation.
• And the problem cumulates over time along traces.
Affine arithmetic

- Interval arithmetic is imprecise.
- For example, if $x \in [1, 4]$ then $x - x \in [1 - 4, 4 - 1] = [-3, 3]$ instead of $[0, 0]$.
- The problem as that the arguments of functions cannot be correlated by a cartesian abstraction.
- So we have to independently take into consideration all possible values of variables within their interval of variation.
- And the problem cumulates over time along traces.
- Several solutions have been proposed to solves this imprecision problem [Nedialkov, Kreinovich, and Starks, 2004].
Affine arithmetic (cont’d)

• One of them, affine arithmetics [Comba and Stolfi, 1993; Stolfi and Figueiredo, 2003], represents an interval \( x \in [\underline{x}, \overline{x}] \) by

\[
x = a_0 + a_1 \varepsilon_x \quad \text{where} \quad a_0 = \frac{\overline{x} + \underline{x}}{2}, \quad a_1 = \frac{\overline{x} - \underline{x}}{2}, \quad \text{and} \quad \varepsilon_x \in [-1, 1] \quad \text{is a fresh auxiliary variable.}
\]
Affine arithmetic (cont’d)

- One of them, *affine arithmetics* [Comba and Stolfi, 1993; Stolfi and Figueiredo, 2003], represents an interval \( x \in [\underline{x}, \overline{x}] \) by

\[
x = a_0 + a_1 \epsilon_x \text{ where } a_0 = \frac{\underline{x} + \overline{x}}{2}, \quad a_1 = \frac{\overline{x} - \underline{x}}{2}, \quad \text{and } \epsilon_x \in [-1, 1] \text{ is a fresh auxiliary variable.}
\]

- Then \( x - x = (a_0 + a_1 \epsilon_x) - (a_0 + a_1 \epsilon_x) = 0 + 0 \epsilon_x \), as required.
Affine arithmetic (cont’d)

- In general a program involves several variables so we have an affine form
  \[ x = a_0 + a_1 \varepsilon_1 + a_2 \varepsilon_2 + \cdots + a_n \varepsilon_n. \]

- This implies \( x \in [a_0 - d, a_0 + d] \) where \( d = \sum_{i=1}^{n} |a_i| \) is the total deviation of \( x \).

- This is, by interval arithmetic, the smallest interval that contains all possible values of \( x \), assuming that each \( \varepsilon_i \) ranges independently over the interval \([-1, +1]\).
Affine arithmetic (cont’d)

• In general a program involves several variables so we have an affine form
\[ x = a_0 + a_1 \epsilon_1 + a_2 \epsilon_2 + \cdots + a_n \epsilon_n. \]

• This implies \( x \in [a_0 - d, a_0 + d] \) where \( d = \sum_{i=1}^{n} |a_i| \) is the total deviation of \( x \).

• This is, by interval arithmetic, the smallest interval that contains all possible values of \( x \), assuming that each \( \epsilon_i \) ranges independently over the interval \([-1, +1]\).

• For \( m \) variables, the affine constraints determine a zonotope [McMullen, 1971], a center-symmetric convex polytope in \( \mathbb{R}^m \), whose faces are themselves center-symmetric [Beck and Robins, 2015, Ch. 9].

• As was the case for interval arithmetic, zonotope arithmetic is an abstract interpretation of the real/float semantics (used in Fluctuat).

Example of zonotope: octagonal zonogon

en.wikipedia.org/wiki/Zonohedron#Zonotopes
Conclusion
Conclusion

• **Interval arithmetics** in scientific computing put bounds on rounding errors in floating point arithmetic [Moore, 1966].

• It is an **abstract interpretation** of the trace semantics and can be computed at runtime for one trace at a time.

• **Tests** may have to consider many executions, which can be quite inefficient (and often considered an error in practice).

• A further abstract yields the **static interval** analysis (by joining states on paths at each program point to get invariants).

• More generally, this provides a **framework for dynamic analysis** (their static over approximation, and the combination of the two).

• **Soundness** guarantee!
Bibliography


Bibliography II


The End, Thank you

The slides are available at: