Abstract Interpretation II

Semantics and Application to Program Verification

Antoine Miné

École normale supérieure, Paris

year 2015–2016

Course 11

13 May 2016

Overview

- **Interval domain**
  - lattice structure $(\sqsubseteq, \sqcup, \ldots)$
  - interval operators $(+, \times, \ldots)$
  - interval analysis $(\mathsf{S}^\sharp[\![\, V \leftarrow e \,]\!])$
- **Loop analysis**
  - widening $(\triangledown)$
  - advanced loop iterations
- Analysis with equation systems (project)
- Backward analysis (project extension)
- Implementation details (TP, project)

Interval lattice

Reminder: Intervals

Idea: abstract program states by the bounds of each variable

- non-relational abstraction

  (aka. attribute-independent, no relation between variables)

- sufficient to express freedom from overflow

  (e.g., computations in machine integers or floats, array accesses)

Intervals: abstraction of sets of integers $\mathcal{P}(\mathbb{Z})$

$$\mathbb{I} \stackrel{\text{def}}{=} \{\, [a,b] \mid a \in \mathbb{Z} \cup \{-\infty\}, b \in \mathbb{Z} \cup \{+\infty\}, a \leq b \,\} \cup \{\bot\}$$

- $-\infty$, $+\infty$ bounds are needed to abstract unbounded sets

  $[-\infty,+\infty] = \top$ represents $\mathbb{Z}$, $[0,+\infty]$ represents $\mathbb{N}$, etc.

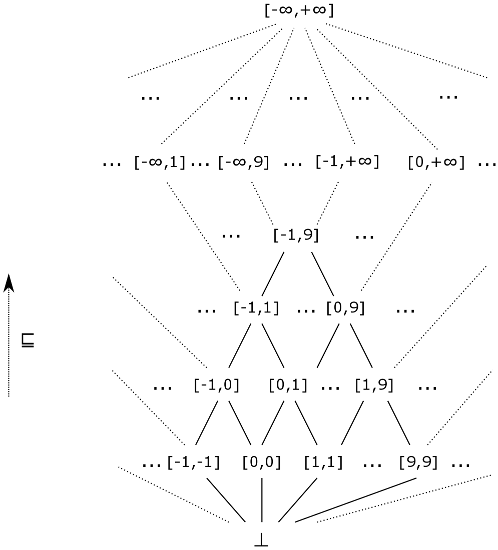
  $\Longrightarrow$ any integer set may be over-approximated in $\mathbb{I}$

  (we can always resort to $\top$)

- $\bot$ (uniquely) represents $\emptyset$

  (in $[a,b]$, we have $a \leq b$ so that non$-\bot$ intervals are never empty)

The interval lattice



Algebraic structure

Partial order: $\sqsubseteq$

- $\forall I \in \mathbb{I}: \bot \sqsubseteq I$
- $[a, b] \sqsubseteq [c, d] \iff a \geq c \wedge b \leq d$

  where $\leq$ is extended naturally to $\mathbb{Z} \cup \{-\infty, +\infty\}$ as: $\forall c \in \mathbb{Z}: -\infty < c < +\infty$

Lattice structure: $\sqcup, \sqcap$

- least upper bound $\sqcup$ for $\sqsubseteq$
  - $\forall I \in \mathbb{I}: \bot \sqcup I = I \sqcup \bot = I$
  - $[a, b] \sqcup [c, d] = [\min(a, c), \max(b, d)]$
- greatest lower bound $\sqcap$:
  - $\forall I: \bot \sqcap I = I \sqcap \bot = \bot$
  - $[a, b] \sqcap [c, d] =$

$$\begin{cases} [\max(a, c), \min(b, d)] & \text{if } \max(a, c) \leq \min(b, d) \\ \bot & \text{if } \max(a, c) > \min(b, d) \end{cases}$$

Algebraic structure

Notes:

- the lattice is complete
  $\forall I \subseteq \mathbb{I}$: $\sqcup\ I$ and $\sqcap\ I$ exist
  $\sqcup \{ [a_j, b_j] \mid j \in J \} = [\min_{j \in J} a_j, \max_{j \in J} b_j]$
  $\sqcap \{ [a_j, b_j] \mid j \in J \} = [\max_{j \in J} a_j, \min_{j \in J} b_j]$ if max $\leq$ min, or $\perp$ otherwise

- intervals are closed by $\cap$
  $[a, b] \cap [c, d] = [a, b] \sqcap [c, d]$

$\Longrightarrow$ this will be useful to define best interval approximations

- intervals are not closed by $\cup$
  $[0, 0] \cup [2, 2] = \{0, 2\}$, which is not an interval; $[0, 0] \sqcup [2, 2] = [0, 2]$

- $\sqcup$ and $\sqcap$ are not distributive
  $([0, 0] \sqcup [2, 2]) \sqcap [1, 1] = [0, 2] \sqcap [1, 1] = [1, 1]$
  but $([0, 0] \sqcap [1, 1]) \sqcup ([2, 2] \sqcap [1, 1]) = \emptyset \sqcup \emptyset = \emptyset$

$\Longrightarrow$ this can be a cause of precision loss

Reminder: Interval Galois connection

Interval Galois connection: $(\mathcal{P}(\mathbb{Z}), \subseteq) \underset{\alpha}{\overset{\gamma}{\leftrightarrows}} (\mathbb{I}, \sqsubseteq)$

- $\begin{cases} \gamma(\bot) \overset{\text{def}}{=} \emptyset \\ \gamma([a,b]) \overset{\text{def}}{=} \{\, x \in \mathbb{Z} \mid a \leq x \leq b \,\} \end{cases}$
- $\alpha(X) \overset{\text{def}}{=} \begin{cases} \bot & \text{if } X = \emptyset \\ [\min X, \max X] & \text{if } X \neq \emptyset \end{cases}$
- Galois connection definition: $\forall I, X\colon \alpha(X) \sqsubseteq I \iff X \subseteq \gamma(I)$
- main property: $\alpha(X)$ is the best abstraction in $\mathbb{I}$ of $X \subseteq \mathbb{Z}$

Proof: that $\alpha(X) \sqsubseteq I \iff X \subseteq \gamma(I)$

$$\begin{array}{lll} & \alpha(X) \sqsubseteq (a,b) & \\ \iff & \min X \geq a \wedge \max X \leq b & (\textit{def. } \alpha, \sqsubseteq) \\ \iff & \forall x \in X\colon a \leq x \leq b & (\textit{def. } \min, \max) \\ \iff & \forall x \in X\colon x \in \{\, y \mid a \leq y \leq b \,\} & \\ \iff & \forall x \in X\colon x \in \gamma([a,b]) & (\textit{def. } \gamma) \\ \iff & X \subseteq \gamma([a,b]) & (\textit{prop. } \subseteq) \end{array}$$

Reminder: Sound, optimal, exact abstractions

Given $F : \mathcal{P}(\mathbb{Z}) \to \mathcal{P}(\mathbb{Z})$, how do we construct $F^\sharp : \mathbb{I} \to \mathbb{I}$?

Optimality: define $F^\sharp$ as $F^\sharp = \alpha \circ F \circ \gamma$

then $\forall I \in \mathbb{I}$: $F^\sharp(I) = \min_\sqsubseteq \{ I' \in \mathbb{I} \mid F(\gamma(I)) \subseteq \gamma(I') \}$

(by definition of Galois connections)

this implies: $\forall I \in \mathbb{I}$: $\gamma(F^\sharp(I)) = \min_\subseteq \{ \gamma(I') \mid F(\gamma(I)) \subseteq \gamma(I') \}$

($F^\sharp$ outputs the smallest interval encompassing all the concrete results)

Note: not all domains have a best abstraction α!

we will see abstract interpretation with just γ

in that case, we say that $F^\sharp$ is optimal if
$\gamma(F^\sharp(I)) = \min_\subseteq \{ \gamma(I') \mid F(\gamma(I)) \subseteq \gamma(I') \}$

(such a $F^\sharp$ may not always exist, nor be unique)

Reminder: Sound, optimal, exact abstractions

Soundness: core property of abstract operators

$\alpha \circ F \circ \gamma \sqsubseteq F^\sharp$

this is equivalent to $F \circ \gamma \subseteq \gamma \circ F^\sharp$

(an abstract step $F^\sharp$ over-approximates a concrete one F)

if no α exist, we take $F \circ \gamma \subseteq \gamma \circ F^\sharp$ as definition of soundness

Exactness: $\forall I \in \mathbb{I}$: $F(\gamma(I)) = \gamma(F^\sharp(I))$

- quite rare: $\forall I \in \mathbb{I}$: $F(\gamma(I))$ must be exactly representable in $\mathbb{I}$
- $\alpha \circ F \circ \gamma$ is not always exact
- even if it exists, such a $F^\sharp$ may be difficult to compute

Summary:

- α provides a systematic way to define an optimal $F^\sharp$
- it may not be possible or practicable to use $\alpha \circ F \circ \gamma$
  $\Longrightarrow$ we settle for a sound $F^\sharp$ instead of an optimal one

Concrete integer operations

Goal: design interval versions of core operators

building blocks to design an interval semantics

Concrete arithmetic operators:

$+, -, \times, /$, lifted to sets $(\mathcal{P}(\mathbb{Z}))^n \to \mathcal{P}(\mathbb{Z})$

$$\begin{array}{lcl}
\overline{-}\, X & \stackrel{\text{def}}{=} & \{\, -x \mid x \in X \,\} \\
X \mathbin{\overline{+}} Y & \stackrel{\text{def}}{=} & \{\, x + y \mid x \in X, y \in Y \,\} \\
X \mathbin{\overline{-}} Y & \stackrel{\text{def}}{=} & \{\, x - y \mid x \in X, y \in Y \,\} \\
X \mathbin{\overline{\times}} Y & \stackrel{\text{def}}{=} & \{\, x \times y \mid x \in X, y \in Y \,\} \\
X \mathbin{\overline{/}} Y & \stackrel{\text{def}}{=} & \{\, x / y \mid x \in X, y \in Y, y \neq 0 \,\}
\end{array}$$

where $/$ rounds towards 0 (truncation)

Set operators: $\cup, \cap, \subseteq, =$

Interval set operators

Optimal binary operators: $A_1 \diamond^\sharp A_2 \stackrel{\text{def}}{=} \alpha(\gamma(A_1) \diamond \gamma(A_2))$

- $\cap^\sharp = \sqcap$

  as $\gamma([a,b] \sqcap [c,d]) = \gamma([a,b]) \cap \gamma([c,d])$

- $\cup^\sharp = \sqcup$

  as
  $$\begin{aligned} & \alpha(\gamma([a,b]) \cup \gamma([c,d])) \\ &= \alpha(\{\, x \mid a \le x \le b \vee c \le x \le d \,\}) \\ &= [\min \{\, x \mid a \le x \le b \vee c \le x \le d \,\}, \max \{\, x \mid a \le x \le b \vee c \le x \le d \,\}] \\ &= [\min(a,c), \max(b,d)] \\ &= [a,b] \sqcup [c,d] \end{aligned}$$

Optimal predicates: $A_1 \bowtie^\sharp A_2 \stackrel{\text{def}}{\iff} \gamma(A_1) \bowtie \gamma(A_2)$

- $\subseteq^\sharp$ is $\sqsubseteq$

  as $\gamma([a,b]) \subseteq \gamma([c,d]) \iff a \ge c \wedge b \le d \iff [a,b] \sqsubseteq [c,d]$

- $=^\sharp$ is $=$

  as $\gamma([a,b]) = \gamma([c,d]) \iff a = c \wedge b = d$

Note: for soundness, $A_1 \bowtie^\sharp A_2 \implies \gamma(A_1) \bowtie \gamma(A_2)$ is actually sufficient

Interval arithmetic: addition, subtraction

- $-^\sharp\ [a, b] = [-b, -a]$
- $[a, b]\ +^\sharp\ [c, d] = [a + c, b + d]$
- $[a, b]\ -^\sharp\ [c, d] = [a - d, b - c]$
- $\forall I \in \mathbb{I}$: $-^\sharp\ \bot = \bot\ +^\sharp\ I = I\ +^\sharp\ \bot = \cdots = \bot$ (strictness)

where: $+$ and $-$ is extended to $+\infty$, $-\infty$ as:
$\forall x \in \mathbb{Z}$: $(+\infty) + x = +\infty,\ (-\infty) + x = -\infty,\ -(+\infty) = (-\infty), \ldots$

Proof: optimality of $+^\sharp$

$$\begin{aligned}
&\alpha(\gamma([a, b])\ \overline{+}\ \gamma([c, d])) \\
&= \alpha(\{\, x \mid a \leq x \leq b \,\}\ \overline{+}\ \{\, y \mid c \leq y \leq d \,\}) \\
&= \alpha(\{\, x + y \mid a \leq x \leq b \wedge c \leq y \leq d \,\}) \\
&= [\min \{\, x + y \mid a \leq x \leq b \wedge c \leq y \leq d \,\}, \max \{\, x + y \mid a \leq x \leq b \wedge c \leq y \leq d \,\}] \\
&= [a + c, b + d] \\
&= [a, b]\ +^\sharp\ [c, d]
\end{aligned}$$

Interval arithmetic: multiplication

- $[a,b] \ \times^\sharp \ [c,d] = [\ \min(a \times c,\ a \times d,\ b \times c,\ b \times d),$
  $\max(a \times c,\ a \times d,\ b \times c,\ b \times d)\ ]$

where $\times$ is extended to $+\infty$ and $-\infty$ by the rule of signs:

$c \times (+\infty) = (+\infty)$ if $c > 0$, $(-\infty)$ if $c < 0$

$c \times (-\infty) = (-\infty)$ if $c > 0$, $(+\infty)$ if $c < 0$

we also need the non-standard rule: $0 \times (+\infty) = 0 \times (-\infty) = 0$

Proof sketch: by decomposition into negative and positive intervals

$$\begin{array}{rcll}
a \geq 1 & \wedge & c \geq 1 & \Longrightarrow \quad [a,b] \ \times^\sharp \ [c,d] = [a \times c, b \times d] \\
b \leq -1 & \wedge & c \geq 1 & \Longrightarrow \quad [a,b] \ \times^\sharp \ [c,d] = [a \times d, b \times c] \\
a \geq 1 & \wedge & d \leq -1 & \Longrightarrow \quad [a,b] \ \times^\sharp \ [c,d] = [b \times c, a \times d] \\
b \leq -1 & \wedge & d \leq -1 & \Longrightarrow \quad [a,b] \ \times^\sharp \ [c,d] = [b \times d, a \times c] \\
a = b = 0 & \vee & c = d = 0 & \Longrightarrow \quad [a,b] \ \times^\sharp \ [c,d] = [0,0]
\end{array}$$

$$\begin{aligned}
[a,b] \times^\sharp [c,d] = {} & ([a,b] \sqcap [1,+\infty]) \times^\sharp ([c,d] \sqcap [1,+\infty]) \sqcup \\
& ([a,b] \sqcap [1,+\infty]) \times^\sharp ([c,d] \sqcap [0,0]) \sqcup \\
& ([a,b] \sqcap [1,+\infty]) \times^\sharp ([c,d] \sqcap [-\infty,-1]) \ \sqcup \cdots
\end{aligned}$$

Interval arithmetic: division

- $/^\sharp$ by case split:
  $([a,b] \;/^\sharp\; ([c,d] \sqcap [1,+\infty])) \quad \sqcup \quad ([a,b] \;/^\sharp\; ([c,d] \sqcap [-\infty,-1]))$
  where
  $$[a,b] \;/^\sharp\; [c,d] = \begin{cases} [\min(a/c, a/d), \max(b/c, b/d)] & \text{if } 1 \leq c \\ [\min(b/c, b/d), \max(a/c, a/d)] & \text{if } d \leq -1 \end{cases}$$

where $/$ is extended to $+\infty$ and $-\infty$ by the rule of signs:

$c/(+\infty) = c/(-\infty) = 0$, including $(+\infty)/(+\infty) = 0$
$(+\infty)/c = (+\infty)$ if $c > 0$, $(-\infty)$ if $c < 0$
$(-\infty)/c = (-\infty)$ if $c > 0$, $(+\infty)$ if $c < 0$

Examples:

$[-5,5]/^\sharp[0,0] = \bot$

$[5,10]/^\sharp[-1,1] = ([5,10]/^\sharp[1,1]) \sqcup ([5,10]/^\sharp[-1,-1]) = [5,10] \sqcup [-10,-5] = [-10,10]$

Interval operator exactness

- exact interval operations: $\cap^\sharp$, $+^\sharp$, $-^\sharp$
- non-exact interval operations: $\cup^\sharp$, $\times^\sharp$, $/^\sharp$

$[0,1] \cup^\sharp [10,11] = [0,11]$ but $\gamma([0,1]) \cup \gamma([10,11]) = \{0,1,10,11\}$

$[0,1] \times^\sharp [2,2] = [0,2]$ but $\gamma([0,1]) \overline{\times} \gamma([2,2]) = \{0,2\}$

$[10,10] /^\sharp [-1,1] = [-10,10]$ but $\gamma([10,10]) \overline{/} \gamma([-1,1]) = \{-10,10\}$

Note: $F^\sharp$ is exact if it is optimal and $\forall a \in A$: $F(\gamma(a)) \in \{\, \gamma(x) \mid x \in A \,\}$

Operator composition

- if $F^\sharp$ and $G^\sharp$ are sound and F is monotonic, then $F^\sharp \circ G^\sharp$ is sound

  Proof:
  $G(\gamma(I)) \subseteq \gamma(G^\sharp(I))$, so: $F(G(\gamma(I))) \subseteq F(\gamma(G^\sharp(I))) \subseteq \gamma(F^\sharp(G^\sharp(I)))$

- if $F^\sharp$ and $G^\sharp$ are exact, then $F^\sharp \circ G^\sharp$ is exact

  Proof: $F(G(\gamma(I))) = F(\gamma(G^\sharp(I))) = \gamma(F^\sharp(G^\sharp(I)))$

- if $F^\sharp$ and $G^\sharp$ are optimal, then $F^\sharp \circ G^\sharp$ is sound
  but not necessarily optimal!

  Example:
  $F(X) \stackrel{\text{def}}{=} \{\, 2x \mid x \in X \,\}$ and $G(X) \stackrel{\text{def}}{=} \{\, x \in X \mid x \geq 1 \,\}$
  $F^\sharp([a,b]) = [2a, 2b]$ and $G^\sharp([a,b]) = [a,b] \cap^\sharp [1, +\infty]$ are optimal
  but $G^\sharp(F^\sharp([0,1])) = [0,2] \cap^\sharp [1,+\infty] = [1,2]$
  while $\alpha(G(F(\gamma([0,1])))) = [2,2]$

$\Longrightarrow$ decomposing the semantics into more fined-grained operators

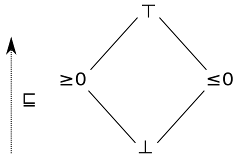
- simplifies analysis design and enhances reusability
- but can degrade the precision

Side-note: Meet closure and optimality

Reminder: $\gamma(a \sqcap a') = \gamma(a) \cap \gamma(a')$

$\implies \{\, \gamma(a) \mid a \in A \,\}$ must be closed under $\cap$

Counter-example: invalid sign domain



$A \stackrel{\text{def}}{=} \{\bot, \leq 0, \geq 0, \top\}$

$\gamma(\leq 0) \cap \gamma(\geq 0) = \{0\} \notin \gamma(A)$

no best abstraction for $\{0\}$

$\implies$ no Galois connection

possible fixes:

- complete A by $\cap$: $A \stackrel{\text{def}}{=} \{\bot, 0, \leq 0, \geq 0, \top\}$
- hollow A, removing elements: $A \stackrel{\text{def}}{=} \{\bot, \geq 0, \top\}$
- fix elements: $A \stackrel{\text{def}}{=} \{\bot, < 0, \geq 0, \top\}$

Side-note: Complete meet closure and optimality

Reminder: $\gamma(\sqcap X) = \cap \{ \gamma(x) \mid x \in X \}$

$\implies \{ \gamma(a) \mid a \in A \}$ must be closed under arbitrary $\cap$

α can be actually defined as $\alpha(c) = \sqcap \{ a \in A \mid c \subseteq \gamma(a) \}$

Counter-example: rational intervals

$\square \stackrel{\text{def}}{=} \{ [a,b] \mid a \in \mathbb{Q} \cup \{-\infty\}, b \in \mathbb{Q} \cup \{+\infty\}, a \leq b \} \cup \{\bot\}$

$X = \{ c \mid c^2 \leq 2 \}$ has no best abstraction
because $\max X = \sqrt{2} \notin \mathbb{Q}$
$\implies$ no Galois connection

we can still define optimal $\cup^\sharp$, $\cap^\sharp$, $+^\sharp$, $-^\sharp$, $\times^\sharp$, $/^\sharp$
such that $\forall a_1, a_2 \colon \gamma(a_1 \diamond^\sharp a_2) = \min_\subseteq \{ \gamma(a) \mid \gamma(a) \subseteq \gamma(a_1) \diamond \gamma(a_2) \}$

but some operators, such as $F(X) \stackrel{\text{def}}{=} \{ \sqrt{x} \mid x \in X \}$, have no best abstraction

$\implies$ we can study abstract domains wrt. the functions they can abstract precisely

Interval analysis

Reminder: Language

Expressions and conditions

$$
\begin{array}{lcll}
expr & ::= & V & V \in \mathbb{V} \\
 & \mid & c & c \in \mathbb{Z} \\
 & \mid & -expr & \\
 & \mid & expr \diamond expr & \diamond \in \{+,-,\times,/\} \\
 & \mid & \mathbf{rand}(a,b) & a,b \in \mathbb{Z} \\
cond & ::= & expr \bowtie expr & \bowtie \in \{\leq,\geq,=,\neq,<,>\} \\
 & \mid & \neg cond & \\
 & \mid & cond \diamond cond & \diamond \in \{\wedge,\vee\}
\end{array}
$$

Statements

$$
\begin{array}{lcl}
stat & ::= & V \leftarrow expr \\
 & \mid & \mathbf{if}\ cond\ \mathbf{then}\ stat\ \mathbf{else}\ stat \\
 & \mid & \mathbf{while}\ cond\ \mathbf{do}\ stat \\
 & \mid & stat; stat \\
 & \mid & \mathbf{skip}
\end{array}
$$

Reminder: Concrete semantics of expressions

Classic non-deterministic concrete semantics, in denotational style:

$\mathsf{E}[\![\, expr \,]\!] : \mathcal{E} \to \mathcal{P}(\mathbb{Z})$ *(arithmetic expressions)*

$$
\begin{array}{lcl}
\mathsf{E}[\![\, V \,]\!]\,\rho & \stackrel{\text{def}}{=} & \{\rho(V)\} \\
\mathsf{E}[\![\, c \,]\!]\,\rho & \stackrel{\text{def}}{=} & \{c\} \\
\mathsf{E}[\![\, \mathbf{rand}(a,b) \,]\!]\,\rho & \stackrel{\text{def}}{=} & \{\, x \mid a \leq x \leq b \,\} \\
\mathsf{E}[\![\, -e \,]\!]\,\rho & \stackrel{\text{def}}{=} & \{\, -v \mid v \in \mathsf{E}[\![\, e \,]\!]\,\rho \,\} \\
\mathsf{E}[\![\, e_1 \diamond e_2 \,]\!]\,\rho & \stackrel{\text{def}}{=} & \{\, v_1 \diamond v_2 \mid v_1 \in \mathsf{E}[\![\, e_1 \,]\!]\,\rho,\ v_2 \in \mathsf{E}[\![\, e_2 \,]\!]\,\rho,\ \diamond \neq / \vee v_2 \neq 0 \,\}
\end{array}
$$

$\mathsf{C}[\![\, cond \,]\!] : \mathcal{E} \to \mathcal{P}(\{\text{true}, \text{false}\})$ *(boolean conditions)*

$$
\begin{array}{lcl}
\mathsf{C}[\![\, \neg c \,]\!]\,\rho & \stackrel{\text{def}}{=} & \{\, \neg v \mid v \in \mathsf{C}[\![\, c \,]\!]\,\rho \,\} \\
\mathsf{C}[\![\, c_1 \diamond c_2 \,]\!]\,\rho & \stackrel{\text{def}}{=} & \{\, v_1 \diamond v_2 \mid v_1 \in \mathsf{C}[\![\, c_1 \,]\!]\,\rho,\ v_2 \in \mathsf{C}[\![\, c_2 \,]\!]\,\rho \,\} \\
\mathsf{C}[\![\, e_1 \bowtie e_2 \,]\!]\,\rho & \stackrel{\text{def}}{=} & \{\, \text{true} \mid \exists v_1 \in \mathsf{E}[\![\, e_1 \,]\!]\,\rho,\ v_2 \in \mathsf{E}[\![\, e_2 \,]\!]\,\rho \colon v_1 \bowtie v_2 \,\} \cup \\
& & \{\, \text{false} \mid \exists v_1 \in \mathsf{E}[\![\, e_1 \,]\!]\,\rho,\ v_2 \in \mathsf{E}[\![\, e_2 \,]\!]\,\rho \colon v_1 \not\bowtie v_2 \,\}
\end{array}
$$

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

Reminder: Concrete semantics of statements

$\mathsf{S}[\![\, stat \,]\!] : \mathcal{P}(\mathcal{E}) \to \mathcal{P}(\mathcal{E})$

$$
\begin{array}{lcl}
\mathsf{S}[\![\, \textbf{skip} \,]\!]\, R & \stackrel{\text{def}}{=} & R \\
\mathsf{S}[\![\, s_1;\; s_2 \,]\!]\, R & \stackrel{\text{def}}{=} & \mathsf{S}[\![\, s_2 \,]\!]\, (\mathsf{S}[\![\, s_1 \,]\!]\, R) \\
\mathsf{S}[\![\, V \leftarrow e \,]\!]\, R & \stackrel{\text{def}}{=} & \{\, \rho[V \mapsto v] \mid \rho \in R,\ v \in \mathsf{E}[\![\, e \,]\!]\, \rho \,\} \\
\mathsf{S}[\![\, \textbf{if } c \textbf{ then } s_1 \textbf{ else } s_2 \,]\!]\, R & \stackrel{\text{def}}{=} & \mathsf{S}[\![\, s_1 \,]\!]\, (\mathsf{S}[\![\, c? \,]\!]\, R) \cup \mathsf{S}[\![\, s_2 \,]\!]\, (\mathsf{S}[\![\, \neg c? \,]\!]\, R) \\
\mathsf{S}[\![\, \textbf{while } c \textbf{ do } s \,]\!]\, R & \stackrel{\text{def}}{=} & \mathsf{S}[\![\, \neg c? \,]\!]\, (\mathrm{lfp}\ \lambda I.R \cup \mathsf{S}[\![\, s \,]\!]\, (\mathsf{S}[\![\, c? \,]\!]\, I))
\end{array}
$$

where

$$
\mathsf{S}[\![\, c? \,]\!]\, R \stackrel{\text{def}}{=} \{\, \rho \in R \mid \mathsf{true} \in \mathsf{C}[\![\, c \,]\!]\, \rho \,\}
$$

$\mathsf{S}[\![\, stat \,]\!]$ is a $\cup-$morphism in the complete lattice $(\mathcal{P}(\mathcal{E}), \subseteq, \cup, \cap, \emptyset, \mathcal{E})$

Reminder: Non-relational abstraction

Reminder: we compose two abstractions:

- $\mathcal{P}(\mathbb{V} \to \mathbb{Z})$ is abstracted as $\mathbb{V} \to \mathcal{P}(\mathbb{Z})$ (forget relationship)
- $\mathcal{P}(\mathbb{Z})$ is abstracted as intervals $\mathbb{I}$ (keep only bounds)

Cartesian lattice:

- $\mathcal{E}^\sharp \stackrel{\text{def}}{=} \mathbb{V} \to \mathbb{I}$
- point-wise order: $X_1^\sharp \mathrel{\dot\sqsubseteq} X_2^\sharp \iff \forall V \in \mathbb{V}: X_1^\sharp(V) \sqsubseteq X_2^\sharp(V)$
- join: $X_1^\sharp \mathrel{\dot\sqcup} X_2^\sharp \stackrel{\text{def}}{=} \lambda V. X_1^\sharp(V) \sqcup X_2^\sharp(V)$
- meet: $X_1^\sharp \mathrel{\dot\sqcap} X_2^\sharp \stackrel{\text{def}}{=} \lambda V. X_1^\sharp(V) \sqcap X_2^\sharp(V)$

$\Longrightarrow$ we still have a complete lattice

Cartesian Galois connection: $(\mathcal{P}(\mathbb{V} \to \mathbb{Z}), \subseteq) \underset{\dot\alpha}{\overset{\dot\gamma}{\leftrightarrows}} (\mathbb{V} \to \mathbb{I}, \dot\sqsubseteq)$

- $\dot\alpha(E) \stackrel{\text{def}}{=} \lambda V. \alpha(\{\, \rho(V) \mid \rho \in R \,\})$
- $\dot\gamma(X^\sharp) \stackrel{\text{def}}{=} \{\, \rho \mid \forall V \in \mathbb{V}: \rho(V) \in \gamma(X^\sharp(V)) \,\}$

Interval expression evaluation

$E^\sharp[\![\, expr \,]\!] : \mathcal{E}^\sharp \to \mathbb{I}$

interval version of $E[\![\, expr \,]\!] : \mathcal{E} \to \mathcal{P}(\mathbb{Z})$

Definition by structural induction, very similar to $E[\![\, expr \,]\!]$

$$
\begin{array}{lcl}
E^\sharp[\![\, V \,]\!]\, X^\sharp & \stackrel{\text{def}}{=} & X^\sharp(V) \\
E^\sharp[\![\, c \,]\!]\, X^\sharp & \stackrel{\text{def}}{=} & [c, c] \\
E^\sharp[\![\, \mathbf{rand}(a, b) \,]\!]\, X^\sharp & \stackrel{\text{def}}{=} & [a, b] \\
E^\sharp[\![\, -e \,]\!]\, X^\sharp & \stackrel{\text{def}}{=} & -^\sharp\ E^\sharp[\![\, e \,]\!]\, X^\sharp \\
E^\sharp[\![\, e_1 \diamond e_2 \,]\!]\, X^\sharp & \stackrel{\text{def}}{=} & E^\sharp[\![\, e_1 \,]\!]\, X^\sharp\ \diamond^\sharp\ E^\sharp[\![\, e_2 \,]\!]\, X^\sharp
\end{array}
$$

Soundness of interval expression evaluation

Soundness: $\cup \{ \mathsf{E}[\![e]\!] \rho \mid \rho \in \dot\gamma(X^\sharp) \} \subseteq \gamma(\mathsf{E}^\sharp[\![e]\!] X^\sharp)$

Proof:

by induction on the expression syntax, using the soundness of abstract operators;
the base cases $\mathsf{E}^\sharp[\![V]\!]$, $\mathsf{E}^\sharp[\![c]\!]$, $\mathsf{E}^\sharp[\![\mathbf{rand}(a,b)]\!]$ are straightforward;
for $+$ (the same holds for $-$, $\times$, $/$):

$$
\begin{array}{ll}
\gamma(\mathsf{E}^\sharp[\![e_1 + e_2]\!] X^\sharp) & \\
= \gamma(\mathsf{E}^\sharp[\![e_1]\!] X^\sharp +^\sharp \mathsf{E}^\sharp[\![e_2]\!] X^\sharp) & (\textit{def. } \mathsf{E}^\sharp[\![\;]\!]) \\
\supseteq \{ v_1 + v_2 \mid v_1 \in \gamma(\mathsf{E}^\sharp[\![e_1]\!] X^\sharp),\ v_2 \in \gamma(\mathsf{E}^\sharp[\![e_2]\!] X^\sharp) \} & (\textit{sound. } +^\sharp) \\
\supseteq \{ v_1 + v_2 \mid \exists \rho_1, \rho_2 \in \dot\gamma(X^\sharp): v_1 \in \mathsf{E}[\![e_1]\!] \rho_1,\ v_2 \in \mathsf{E}[\![e_2]\!] \rho_2 \} & (\textit{induction}) \\
\supseteq \{ v_1 + v_2 \mid \exists \rho \in \dot\gamma(X^\sharp): v_1 \in \mathsf{E}[\![e_1]\!] \rho,\ v_2 \in \mathsf{E}[\![e_2]\!] \rho \} & (\textit{prop. } \exists) \\
= \cup \{ \mathsf{E}[\![e_1 + e_2]\!] \rho \mid \rho \in \dot\gamma(X^\sharp) \} & (\textit{def. } \mathsf{E}[\![\;]\!])
\end{array}
$$

Non-optimality except in rare cases because:

- the composition of optimal operators is not always optimal
- of the core non-relational abstraction: $\mathcal{P}(\mathbb{V} \to \mathbb{Z}) \underset{\alpha}{\overset{\gamma}{\leftrightarrows}} \mathbb{V} \to \mathcal{P}(\mathbb{Z})$

  e.g.: $\mathsf{E}^\sharp[\![V - V]\!] [V \mapsto [0,1]] = [0,1] -^\sharp [0,1] = [-1,1]$
  but $\alpha(\cup \{ \mathsf{E}[\![V - V]\!] \rho \mid \rho \in \dot\gamma([V \mapsto [0,1]]) \}) = [0,0]$

Abstract interval statements: first part

$S^\sharp[\![\, stat \,]\!] : \mathcal{E}^\sharp \to \mathcal{E}^\sharp$ interval version of $S[\![\, stat \,]\!] : \mathcal{P}(\mathcal{E}) \to \mathcal{P}(\mathcal{E})$

- $S[\![\, \textbf{skip} \,]\!]\, R \stackrel{\text{def}}{=} R$
  $S^\sharp[\![\, \textbf{skip} \,]\!]\, X^\sharp \stackrel{\text{def}}{=} X^\sharp$ (identity)
- $S[\![\, s_1;\ s_2 \,]\!]\, R \stackrel{\text{def}}{=} S[\![\, s_2 \,]\!]\, (S[\![\, s_1 \,]\!]\, R)$
  $S^\sharp[\![\, s_1; s_2 \,]\!]\, X^\sharp \stackrel{\text{def}}{=} S^\sharp[\![\, s_2 \,]\!]\, (S^\sharp[\![\, s_1 \,]\!]\, X^\sharp)$ (composition)
- $S[\![\, V \leftarrow e \,]\!]\, R \stackrel{\text{def}}{=} \{\, \rho[V \mapsto v] \mid \rho \in R,\ v \in E[\![\, e \,]\!]\, \rho \,\}$
  $S^\sharp[\![\, V \leftarrow e \,]\!]\, X^\sharp \stackrel{\text{def}}{=} \begin{cases} X^\sharp[V \mapsto E^\sharp[\![\, e \,]\!]\, X^\sharp] & \text{if } E^\sharp[\![\, e \,]\!]\, X^\sharp \neq \bot \\ \dot\bot & \text{if } E^\sharp[\![\, e \,]\!]\, X^\sharp = \bot \end{cases}$

(tests and loops are more complex, they are presented in the next slides)

Soundness proof: i.e., $S[\![\, s \,]\!]\, (\dot\gamma(X^\sharp)) \subseteq \dot\gamma(S^\sharp[\![\, s \,]\!]\, X^\sharp)$

obvious for **skip**; by composition of soundness for $s_1; s_2$;
for $V \leftarrow e$ we derive:

$$\begin{aligned} & \dot\gamma(S^\sharp[\![\, V \leftarrow e \,]\!]\, X^\sharp) \\ &= \{\, \rho[V \mapsto v] \mid \forall W\colon \rho(W) \in \gamma(X^\sharp(W)),\ v \in \gamma(E^\sharp[\![\, e \,]\!]\, X^\sharp) \,\} && (\textit{def. } \dot\gamma,\ S^\sharp[\![\ ]\!]) \\ &\supseteq \{\, \rho[V \mapsto v] \mid \forall W\colon \rho(W) \in \gamma(X^\sharp(W)),\ v \in E[\![\, e \,]\!]\, \rho \,\} && (\textit{sound. } E^\sharp[\![\ ]\!]) \\ &= S[\![\, V \to e \,]\!]\, \dot\gamma(X^\sharp) && (\textit{def. } \dot\gamma,\ S[\![\ ]\!]) \end{aligned}$$

Tests

Abstract tests

conditionals and loops use the auxiliary "test" statement:

$\mathsf{S}[\![\, c?\,]\!]\, R \stackrel{\text{def}}{=} \{\, \rho \in R \mid \text{true} \in \mathsf{C}[\![\, c\,]\!]\, \rho \,\}$

Abstract tests: $\mathsf{S}^\sharp[\![\, c?\,]\!]$

Preprocessing: remove $\neg$, $=$, $\neq$, $>$, $\geq$, $<$

- $\neg$ can be removed using De Morgan's law:

  $\neg(c_1 \vee c_2) \rightsquigarrow \neg c_1 \wedge \neg c_2$

  $\neg(c_1 \wedge c_2) \rightsquigarrow \neg c_1 \vee \neg c_2$

  $\neg(e_1 \leq e_2) \rightsquigarrow e_1 > e_2 \ldots$

- $=$, $\neq$, $>$, $\geq$, $<$ can be expressed using only $\leq$, $\vee$ and $\wedge$:

  $e_1 < e_2 \rightsquigarrow e_1 \leq (e_2 - 1)$

  $e_1 \geq e_2 \rightsquigarrow e_2 \leq e_1$

  $e_1 > e_2 \rightsquigarrow e_2 \leq (e_1 - 1)$

  $e_1 = e_2 \rightsquigarrow (e_1 \leq e_2) \wedge (e_2 \leq e_1)$

  $e_1 \neq e_2 \rightsquigarrow (e_1 \leq (e_2 - 1)) \vee (e_2 \leq (e_1 - 1))$

Interval test (cont.)

Handling boolean operators: by induction

- $\mathsf{S}^\sharp[\![\, c_1 \vee c_2 ?\,]\!]\, X^\sharp \stackrel{\text{def}}{=} (\mathsf{S}^\sharp[\![\, c_1 ?\,]\!]\, X^\sharp)\ \dot{\cup}^\sharp\ (\mathsf{S}^\sharp[\![\, c_2 ?\,]\!]\, X^\sharp)$
- $\mathsf{S}^\sharp[\![\, c_1 \wedge c_2 ?\,]\!]\, X^\sharp \stackrel{\text{def}}{=} (\mathsf{S}^\sharp[\![\, c_1 ?\,]\!]\, X^\sharp)\ \dot{\cap}^\sharp\ (\mathsf{S}^\sharp[\![\, c_2 ?\,]\!]\, X^\sharp)$

Simple tests: comparing a variable to a variable or a constant

assuming $X^\sharp(V) = [a, b]$ and $X^\sharp(W) = [c, d]$

- $\mathsf{S}^\sharp[\![\, V \leq v ?\,]\!]\, X^\sharp \stackrel{\text{def}}{=} \begin{cases} X^\sharp[V \mapsto [a, \min(b, v)]] & \text{if } a \leq v \\ \dot{\bot} & \text{if } a > v \end{cases}$
- $\mathsf{S}^\sharp[\![\, V \leq W ?\,]\!]\, X^\sharp \stackrel{\text{def}}{=} \begin{cases} X^\sharp[\ V \mapsto [a, \min(b, d)], & \text{if } a \leq d \\ \qquad W \mapsto [\max(a, c), d]\ ] & \\ \dot{\bot} & \text{if } a > d \end{cases}$

  (W's upper bound refines V's, V's lower bound refines W's)

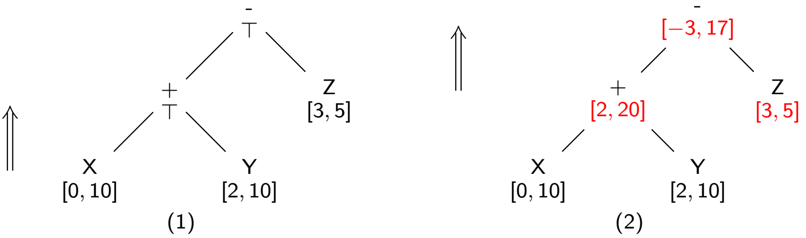
(next slides: how to handle tests with arbitrary expressions)

Example of complex interval test

Example: $S^\sharp[\![\, X + Y - Z \leq 0 \,]\!]\, X^\sharp$

with $X^\sharp = \{\, X \mapsto [0, 10], Y \mapsto [2, 10], Z \mapsto [3, 5] \,\}$

First step: annotate the expression tree



bottom-up evaluation using abstract interval operators $+^\sharp$, $-^\sharp$, etc.

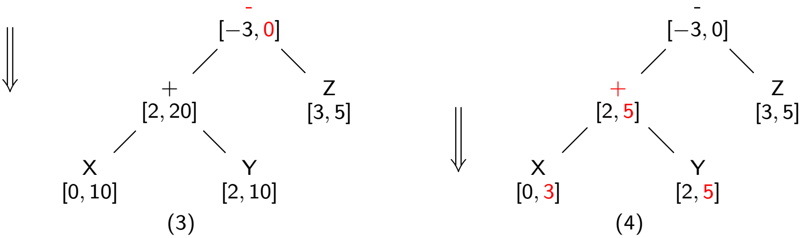
(similar to interval assignment)

Example of complex interval test (cont.)

Example: $S^\sharp[\![\, X + Y - Z \leq 0 \,]\!]\, X^\sharp$

with $X^\sharp = \{\, X \mapsto [0,10], Y \mapsto [2,10], Z \mapsto [3,5] \,\}$

Second step: refine the expression tree top-down



- refine the root: we know the result is negative
- propagate refined nodes downward toward leaves
- use refined variable values: $\{\, X \mapsto [0,3], Y \mapsto [2,5], Z \mapsto [3,5] \,\}$

$\Longrightarrow$ we need new abstract operators to model refinement $\overleftarrow{\leq 0}^\sharp$, $\overleftarrow{+}^\sharp$, etc.

Backward arithmetic and comparison operators

Sound backward arithmetic and comparison operators that refine their argument given a result.

- Backward comparison operators, applied at the root:

$$X^{\sharp\prime} = \overleftarrow{\leq 0}^{\sharp}(X^{\sharp})$$
$$\implies \{\, x \in \gamma(X^{\sharp}) \,|\, x \leq 0 \,\} \subseteq \gamma(X^{\sharp\prime}) \subseteq \gamma(X^{\sharp})$$

- Backward arithmetic operators, applied at internal expression nodes:

$$X^{\sharp\prime} = \overleftarrow{-}^{\sharp}(X^{\sharp}, R^{\sharp})$$
$$\implies \{\, x \,|\, x \in \gamma(X^{\sharp}),\, -x \in \gamma(R^{\sharp}) \,\} \subseteq \gamma(X^{\sharp\prime}) \subseteq \gamma(X^{\sharp})$$

$$(X^{\sharp\prime}, Y^{\sharp\prime}) = \overleftarrow{+}^{\sharp}(X^{\sharp}, Y^{\sharp}, R^{\sharp})$$
$$\implies \{\, x \in \gamma(X^{\sharp}) \,|\, \exists y \in \gamma(Y^{\sharp}),\, x + y \in \gamma(R^{\sharp}) \,\} \subseteq \gamma(X^{\sharp\prime}) \subseteq \gamma(X^{\sharp})$$
$$\text{and } \{\, y \in \gamma(Y^{\sharp}) \,|\, \exists x \in \gamma(X^{\sharp}),\, x + y \in \gamma(R^{\sharp}) \,\} \subseteq \gamma(Y^{\sharp\prime}) \subseteq \gamma(Y^{\sharp})$$

$$\vdots$$

Note: best backward operators can be designed with α

e.g. for $\overleftarrow{+}^{\sharp}$: $X^{\sharp\prime} = \alpha(\{\, x \in \gamma(X^{\sharp}) \,|\, \exists y \in \gamma(Y^{\sharp}),\, x + y \in \gamma(R^{\sharp}) \,\})$

Generic backward operator construction

Synthesizing (non optimal) backward arithmetic operators from forward arithmetic operators

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

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

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

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

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

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

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

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

Interval backward operators

Applying the generic construction to the interval domain, we get:

- $\overleftarrow{\leq 0}^\sharp([a,b]) \stackrel{\text{def}}{=} \begin{cases} [a, \min(b,0)] & \text{if } a \geq 0 \\ \bot & \text{otherwise} \end{cases}$
- $\overleftarrow{-}^\sharp([a,b],[r,s]) \stackrel{\text{def}}{=} [a,b] \cap^\sharp [-s,-r]$
- $\overleftarrow{+}^\sharp([a,b],[c,d],[r,s]) \stackrel{\text{def}}{=} ([a,b] \cap^\sharp [r-d, s-c], \; [c,d] \cap^\sharp [r-b, s-a])$
- ...

Interval conditionals

Concrete semantics:

$$\mathsf{S}[\![\, \textbf{if } c \textbf{ then } s_1 \textbf{ else } s_2 \,]\!]\, R$$
$$\stackrel{\text{def}}{=} \mathsf{S}[\![\, s_1 \,]\!]\, (\mathsf{S}[\![\, c? \,]\!]\, R) \cup \mathsf{S}[\![\, s_2 \,]\!]\, (\mathsf{S}[\![\, \neg c? \,]\!]\, R)$$

Abstract semantics: compose existing abstract operators:

$$\mathsf{S}^\sharp[\![\, \textbf{if } c \textbf{ then } s_1 \textbf{ else } s_2 \,]\!]\, X^\sharp$$
$$\stackrel{\text{def}}{=} \mathsf{S}^\sharp[\![\, s_1 \,]\!]\, (\mathsf{S}^\sharp[\![\, c? \,]\!]\, X^\sharp) \;\dot{\cup}^\sharp\; \mathsf{S}^\sharp[\![\, s_2 \,]\!]\, (\mathsf{S}^\sharp[\![\, \neg c? \,]\!]\, X^\sharp)$$

Soundness proof:

by soundness of the composition of sound operators

Example: $stat \stackrel{\text{def}}{=} V \leftarrow 2 \times \textbf{rand}(0,1);\ \textbf{if } V > 1 \textbf{ then } V \leftarrow 0 \textbf{ else skip}$

given $E^\sharp \stackrel{\text{def}}{=} [V \mapsto [-\infty, +\infty]]$
we get: $\mathsf{S}^\sharp[\![\, stat \,]\!]\, E^\sharp = [V \mapsto [0,1]]$

note that $\mathsf{S}[\![\, stat \,]\!]\, \mathcal{E} = \{[V \mapsto 0]\}$
$\Longrightarrow \mathsf{S}^\sharp[\![\, stat \,]\!]$ is sound but not optimal

Loops

Interval loops

Concrete semantics:

$\mathsf{S}[\![\, \textbf{while } c \textbf{ do } s \,]\!]\, R \stackrel{\text{def}}{=} \mathsf{S}[\![\, \neg c? \,]\!]\, (\text{lfp } F)$

where $F(I) \stackrel{\text{def}}{=} R \cup \mathsf{S}[\![\, s \,]\!]\, (\mathsf{S}[\![\, c? \,]\!]\, I))$

Reminder: lfp F exists because F is monotonic
in fact, lfp $F = \cup_{n \in \mathbb{N}} F^n(\emptyset)$ because F is a $\cup-$morphism

Abstract fixpoint computation:

given a sound abstraction $F^\sharp$ of F, how can we abstract lfp F?

- lfp $F^\sharp$ may not exist
  $\Longrightarrow$ we seek only $X^\sharp$ such that $F^\sharp(X^\sharp) \sqsubseteq X^\sharp$ (post fixpoint)
- $F^\sharp$ may be non monotonic (example presented later)
  $\Longrightarrow$ we compute $X^\sharp_{n+1} \stackrel{\text{def}}{=} X^\sharp_n \sqcup F^\sharp(X^\sharp_n)$ (abstract iterations)
- $X^\sharp_n$ may increase infinitely (e.g., $F^\sharp(X^\sharp) = X^\sharp +^\sharp [1,1]$)
  $\Longrightarrow$ we use convergence acceleration

Convergence acceleration

Widening: binary operator $\triangledown : \mathcal{E}^\sharp \times \mathcal{E}^\sharp \to \mathcal{E}^\sharp$ such that:

- $\gamma(X^\sharp) \cup \gamma(Y^\sharp) \subseteq \gamma(X^\sharp \triangledown Y^\sharp)$ (sound abstraction of $\cup$)
- for any sequence $(X^\sharp_n)_{n\in\mathbb{N}}$, the sequence $(Y^\sharp_n)_{n\in\mathbb{N}}$

$$\begin{cases} Y^\sharp_0 & \stackrel{\text{def}}{=} X^\sharp_0 \\ Y^\sharp_{n+1} & \stackrel{\text{def}}{=} Y^\sharp_n \triangledown X^\sharp_{n+1} \end{cases}$$

stabilizes in finite time: $\exists N \in \mathbb{N}$: $Y^\sharp_N = Y^\sharp_{N+1}$

Fixpoint approximation theorem:

- the sequence $X^\sharp_{n+1} \stackrel{\text{def}}{=} X^\sharp_n \triangledown F^\sharp(X^\sharp_n)$ stabilizes in finite time
- when $X^\sharp_{N+1} \sqsubseteq X^\sharp_N$, then $X^\sharp_N$ abstracts lfp F

Soundness proof: assume $X^\sharp_{N+1} \sqsubseteq X^\sharp_N$, then

$\gamma(X^\sharp_N) \supseteq \gamma(X^\sharp_{N+1}) = \gamma(X^\sharp_N \triangledown F^\sharp(X^\sharp_N)) \supseteq \gamma(F^\sharp(X^\sharp_N)) \supseteq F(\gamma(X^\sharp_N))$

$\gamma(X^\sharp_N)$ is a post-fixpoint of F, but lfp F is F's least post-fixpoint, so, $\gamma(X^\sharp_N) \supseteq$ lfp F

Interval loops (cont.)

Concrete semantics:

$S[\![\, \textbf{while } c \textbf{ do } s \,]\!]\, R$

$\stackrel{\text{def}}{=} S[\![\, \neg c? \,]\!]\, (\text{lfp}\, \lambda I.R \cup S[\![\, s \,]\!]\, (S[\![\, c? \,]\!]\, I)))$

Abstract semantics

compose existing sound abstractions
employ convergence acceleration $\triangledown$

$S^\sharp[\![\, \textbf{while } c \textbf{ do } s \,]\!]\, X^\sharp$

$\stackrel{\text{def}}{=} S^\sharp[\![\, \neg c? \,]\!]\, (\lim \lambda I^\sharp.I^\sharp \triangledown (X^\sharp \,\dot{\cup}^\sharp\, S^\sharp[\![\, s \,]\!]\, (S^\sharp[\![\, c? \,]\!]\, I^\sharp)))$

(where $\lim F^\sharp$ iterates the function $F^\sharp$ from $\dot{\bot}$ until $F^\sharp(X^\sharp) \sqsubseteq X^\sharp$)

Interval widening

Interval widening $\triangledown : \mathbb{I} \times \mathbb{I} \to \mathbb{I}$

$$\forall I \in \mathbb{I}: \bot \triangledown I = I \triangledown \bot = I$$

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

- an unstable lower bound is put to $-\infty$
- an unstable upper bound is put to $+\infty$
- once at $-\infty$ or $+\infty$, the bound becomes stable

Point-wise lifting: $\dot{\triangledown} : \mathcal{E}^\sharp \times \mathcal{E}^\sharp \to \mathcal{E}^\sharp$

$$X^\sharp \mathbin{\dot{\triangledown}} Y^\sharp \stackrel{\text{def}}{=} \lambda V \in \mathbb{V}. X^\sharp(V) \triangledown Y^\sharp(V)$$

extrapolate each variable independently

$\Longrightarrow$ stabilization in at most $2|\mathbb{V}|$ iterations

Analysis example with widening

Example

$V \leftarrow 1;$
while $V \leq 50$ **do** $V \leftarrow V + 2$

We must compute $S^\sharp[\![V > 50]\!] (\lim \lambda I^\sharp . I^\sharp \mathbin{\dot\triangledown} F^\sharp(I^\sharp))$
where $F^\sharp(I^\sharp) \stackrel{\text{def}}{=} [1,1] \mathbin{\dot\cup^\sharp} S^\sharp[\![V \leftarrow V + 2]\!] (S^\sharp[\![V \leq 50]\!] I^\sharp)$

iterates with widening:

$$\begin{array}{lllll}
I_0^\sharp = \bot & & & & \\
I_1^\sharp = I_0^\sharp \triangledown F^\sharp(I_0^\sharp) & = \bot \triangledown [1,1] & & = [1,1] & \\
I_2^\sharp = I_1^\sharp \triangledown F^\sharp(I_1^\sharp) & = [1,1] \triangledown ([1,1] \cup^\sharp [3,3]) & & = [1,1] \triangledown [1,3] & = [1,+\infty] \\
I_3^\sharp = I_2^\sharp \triangledown F^\sharp(I_2^\sharp) & = [1,+\infty] \triangledown ([1,1] \cup^\sharp [3,52]) & & = [1,+\infty] \triangledown [1,52] & = [1,+\infty] = I_2^\sharp
\end{array}$$

$\Longrightarrow \lim \lambda I^\sharp . I^\sharp \triangledown F^\sharp(I^\sharp) = [1,+\infty]$

At the end of the program, we find $S^\sharp[\![V > 50]\!] I_3^\sharp = [51,+\infty]$

The concrete semantics would give $\{51\}$

Intuitions behind the widening

Inductive reasoning (philosophical logic)

- induction = generalization from a small set of observations
  e.g., if the upper bound is increasing, it is probably unbounded
  major cognitive process
- $\neq$ induction in mathematics, which is deductive by nature
  (apply an induction axiom)
- in philosophy, induction is unreliable (finite observation)
  but in abstract interpretation, widening is always sound!

Inductive invariants

- lfp F defines the most precise invariant (concrete semantics)
- X such that lfp $F \subseteq X$ is a (possibly less precise) invariant
- X such that $F(X) \subseteq X$ is an inductive invariant
  (X is an invariant, and it can be proved to be invariant without computing lfp F)
- $X^\sharp$ such that $F^\sharp(X^\sharp) \sqsubseteq X^\sharp$ is an abstract inductive invariant
  ($\gamma(X^\sharp)$ can be proved to be invariant in the abstract, without computing lfp F)

Side-node: Non-monotonicity of widening

Example: Consider again *stat* $\stackrel{\text{def}}{=}$ **while** $V \leq 50$ **do** $V \leftarrow V + 2$

we have $\mathsf{S}^\sharp[\![\, stat \,]\!]\, X^\sharp \;=\; \mathsf{S}^\sharp[\![\, V > 50 \,]\!]\, (\lim \lambda I^\sharp . I^\sharp \,\dot{\triangledown}\, F^\sharp(X^\sharp, I^\sharp))$

where $F^\sharp(X^\sharp, I^\sharp) \stackrel{\text{def}}{=} X^\sharp \,\dot{\cup}^\sharp\, \mathsf{S}^\sharp[\![\, V \leftarrow V + 2 \,]\!]\, (\mathsf{S}^\sharp[\![\, V \leq 50 \,]\!]\, I^\sharp)$

$\triangledown$ is not monotonic in its left argument:
(e.g., $[1,1] \triangledown [1,52] = [1,+\infty]$, but $[1,52] \triangledown [1,52] = [1,52]$)

- if $X^\sharp = [1,1]$, $F^\sharp$'s iterates are: $\bot$, $[1,1]$, $[1,+\infty]$
  ($[1,1] \triangledown F^\sharp([1,1],[1,1]) = [1,1] \triangledown ([1,1] \cup^\sharp [3,3]) = [1,1] \triangledown [1,3] = [1,+\infty]$)
  $\Longrightarrow \mathsf{S}^\sharp[\![\, stat \,]\!]\, ([1,1]) = [51,+\infty]$
- if $X^\sharp = [1,52]$, $F^\sharp$'s iterates are: $\bot$, $[1,52]$, $[1,52]$
  ($[1,52] \triangledown F^\sharp([1,52],[1,52]) = [1,52] \triangledown ([1,1] \cup^\sharp [3,52]) = [1,52] \triangledown [1,52] = [1,52]$)
  $\Longrightarrow \mathsf{S}^\sharp[\![\, stat \,]\!]\, ([1,52]) = [51,52]$

$\Longrightarrow \mathsf{S}^\sharp[\![\, stat \,]\!]$ **is not monotonic**

(thankfully, $\triangledown$ can over-approximate lfp F given a non-monotonic abstraction $F^\sharp$ of F)

Widening delay

Example

$V \leftarrow 0;$
while $\cdots$ **do**
  if $V = 0$ **then** $V \leftarrow 1;$
  $\cdots$

V is only increased once, from 0 to 1

Problem: $\triangledown$ will set V to $[0, +\infty] \implies$ loss of precision
(because $[0, 0] \triangledown [0, 1] = [0, +\infty]$)

Solution: **delay** the widening for one (or more) iteration(s):

$$X_{n+1} \stackrel{\text{def}}{=} \begin{cases} X_n^\sharp \cup^\sharp F^\sharp(X_n^\sharp) & \text{if } n < N \\ X_n^\sharp \triangledown F^\sharp(X_n^\sharp) & \text{if } n \geq N \end{cases}$$

(e.g.: $X_1^\sharp = [0, 0] \cup^\sharp [1, 1] = [0, 1]$, $X_2^\sharp = [0, 1] \triangledown [0, 1] = [0, 1] = X_1^\sharp$)

using $\triangledown$ after a fixed number N of iterations is sufficient to ensure stabilization

Loop unrolling

Example

$V \leftarrow 1;$
while $\cdots$ **do**
  if $V = 1$ **then** $(V \leftarrow 0; X \leftarrow 0)$;
  stat;
  $X \leftarrow X + 1$

X is initialized in the first loop iteration; then it is incremented
$\Longrightarrow X \geq 0$ when *stat* is executed

Imprecision:

$X \in [-\infty, +\infty]$ when entering the loop

$\Longrightarrow$ the most precise non-relational loop invariant is:
$V \in [0, 1] \wedge X \in [-\infty, +\infty]$

at *stat*, we have: $V = 0 \wedge X \in [-\infty, +\infty]$ (not $X \in [0, +\infty]$)

Loop unrolling

Example

$V \leftarrow 1;$
while $\cdots$ **do**
  if $V = 1$ **then** $(V \leftarrow 0; X \leftarrow 0);$
  stat;
  $X \leftarrow X + 1$

Solution: loop unrolling

Analyze the N first loop iterations separately

Compute an abstract invariant only for the iterates $\geq N$

We compute $Y^\sharp \stackrel{\text{def}}{=} \mathsf{S}^\sharp[\![\, \textbf{while } c \textbf{ do } s \,]\!]\, X^\sharp$ as:

$U_0^\sharp \stackrel{\text{def}}{=} X^\sharp$ (loop entry)

$U_{n+1}^\sharp \stackrel{\text{def}}{=} \mathsf{S}^\sharp[\![\, s \,]\!]\,(\mathsf{S}^\sharp[\![\, c? \,]\!]\,(U_n^\sharp))$ (n−th unrolling)

$A^\sharp = \stackrel{\text{def}}{=} \lim \lambda I^\sharp . I^\sharp \,\triangledown\, (U_N^\sharp \,\dot{\cup}^\sharp\, \mathsf{S}^\sharp[\![\, s \,]\!]\,(\mathsf{S}^\sharp[\![\, c? \,]\!]\, I^\sharp))$ (inv. after N unrollings)

$Y^\sharp \stackrel{\text{def}}{=} \mathsf{S}^\sharp[\![\, \neg c? \,]\!]\, A^\sharp \cup^\sharp (\cup^\sharp_{n<N} \mathsf{S}^\sharp[\![\, \neg c? \,]\!]\, U_n^\sharp)$ (loop exit)

Decreasing iterations

Example

$V \leftarrow 1;$
while $V \leq 50$ **do** $V \leftarrow V + 2$

Imprecision

In this example, we found $V \in [1, +\infty]$ as loop invariant
but the most precise interval invariant is $V \in [1, 52]$

Solution: decreasing iterations

after stabilizing an iteration with widening
we can continue iterating without the widening to gain precision

- compute as before $X^\sharp \stackrel{\text{def}}{=} \lim \lambda I^\sharp . I^\sharp \triangledown F^\sharp(I^\sharp)$
  we get an abstract post-fixpoint $X^\sharp \sqsupseteq F^\sharp(X^\sharp)$, so $F(\gamma(X^\sharp)) \subseteq \gamma(X^\sharp)$
- then compute $Y^\sharp_n \stackrel{\text{def}}{=} F^{\sharp n}(X^\sharp)$
  by soundness, $\gamma(Y^\sharp_n)$ is also a post-fixpoint of F for every n
  we stop after a fixed finite n, or when $Y^\sharp_{n+1} = Y^\sharp_n$

Decreasing iterations

Example

$V \leftarrow 1;$
while $V \leq 50$ **do** $V \leftarrow V + 2$

Imprecision

In this example, we found $V \in [1, +\infty]$ as loop invariant
but the most precise interval invariant is $V \in [1, 52]$

Solution: decreasing iterations

here: $F^\sharp(I^\sharp) \stackrel{\text{def}}{=} [1,1] \,\dot{\cup}^\sharp\, \mathsf{S}^\sharp[\![\, V \leftarrow V + 2 \,]\!]\, (\mathsf{S}^\sharp[\![\, V \leq 50 \,]\!]\, I^\sharp)$

$$X^\sharp \stackrel{\text{def}}{=} \lim \lambda I^\sharp . I^\sharp \,\triangledown\, F^\sharp(I^\sharp) = [1, +\infty]$$
$$Y_1^\sharp = F^\sharp(X^\sharp) = [1,1] \cup^\sharp [2,52] = [1,52]$$
$$Y_2^\sharp = F^\sharp(Y_1^\sharp) = [1,52] = Y_1^\sharp$$

we find the most precise loop invariant expressible using intervals!

at the loop exit, we get: $\mathsf{S}^\sharp[\![\, V > 50 \,]\!]\, ([1,52]) = [51,52]$

Widening with thresholds

Example

$V \leftarrow 40;$
while $V \neq 0$ **do** $V \leftarrow V - 1$

Imprecision

V decreases form 40 (to 0)

$\Longrightarrow$ iterations with widening find the loop invariant: $V \in [-\infty, 40]$

$S^\sharp[\![V \leftarrow V - 1]\!](S^\sharp[\![V \neq 0]\!][-\infty, 40]) = [-\infty, 39]$

$\Longrightarrow$ decreasing iterations are ineffective

Note: this is caused by the $\neq 0$ test instead of ≥ 0
with $\neq 0$, every set $[a, 40] \setminus \{-1\}$ for $a \leq 0$ is a fixpoint
with ≥ 0, we have a single fixpoint: $[0, 40]$

Widening with thresholds

Example

$V \leftarrow 40;$
while $V \neq 0$ **do** $V \leftarrow V - 1$

Solution widening with **thresholds** T

T: fixed finite set of integers containing $-\infty$ and $+\infty$

$\triangledown$ "jumps" to the next value in T

$\implies \triangledown$ tests the stability of values in T

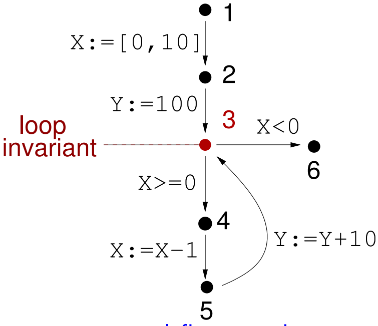
$$[a,b] \triangledown [c,d] \stackrel{\text{def}}{=} \left[\begin{cases} a & \text{if } a \leq c \\ \max \{ t \in T \mid t \leq c \} & \text{if } a > c \end{cases}, \begin{cases} b & \text{if } b \geq d \\ \min \{ t \in T \mid t \geq d \} & \text{if } b < d \end{cases} \right]$$

In our example, we find as loop invariant: $[\max \{ t \in T \mid t \leq 0 \}, 40]$
if $0 \in T$, we find the most precise invariant $[0, 40]$

Solving equation systems with widening

Program semantics as equation system

Alternate view of program semantics:



control flow graph

$$\begin{cases} R_1 = (\{X, Y\} \to \mathbb{Z}) \\ R_2 = \mathsf{S}[\![X \leftarrow [0,10]]\!] R_1 \\ R_3 = \mathsf{S}[\![Y \leftarrow 100]\!] R_2 \cup \\ \qquad \mathsf{S}[\![Y \leftarrow Y + 10]\!] R_5 \\ R_4 = \mathsf{S}[\![X \geq 0]\!] R_3 \\ R_5 = \mathsf{S}[\![X \leftarrow X - 1]\!] R_4 \\ R_6 = \mathsf{S}[\![X < 0]\!] R_3 \end{cases}$$

equation system

- the system has a unique smallest solution
  (it's a least fixpoint in the complete lattice $\mathcal{P}(\{X, Y\} \to \mathbb{Z})$!)
- R_i is the best invariant at program point i
  (e.g. $R_3 = \{ \rho \mid \rho(X) \in [0,10],\ 10\rho(X) + \rho(Y) \in [100,200] \cap 10\mathbb{Z} \}$)
- one big equation system of the form $R_i = F_i(R_1, \ldots, R_n)$

Resolution

Concrete resolution: iterations R^k

$$\begin{cases} R_i^0 \stackrel{\text{def}}{=} \emptyset \\ R_i^{k+1} \stackrel{\text{def}}{=} F_i(R_1^k, \ldots, R_n^k) \end{cases}$$

may not converge in finite time...

Abstract resolution: iterations $X^{\sharp k}$ in the abstract with widening

- choose an abstract domain and sound versions $F_i^\sharp$ of the F_i
- choose a set of widening points W
  every cycle in the CFG should pass through W
  e.g., choose loop heads as W

$$\begin{cases} X_i^{\sharp 0} \stackrel{\text{def}}{=} \bot \\ X_i^{\sharp k+1} \stackrel{\text{def}}{=} F_i^\sharp(X_1^{\sharp k}, \ldots, X_n^{\sharp k}) & \text{if } i \notin W \\ X_i^{\sharp k+1} \stackrel{\text{def}}{=} X_i^{\sharp k} \triangledown F_i^\sharp(X_1^{\sharp k}, \ldots, X_n^{\sharp k}) & \text{if } i \in W \end{cases}$$

$\Longrightarrow$ converges in finite time

(more clever algorithms exist: worklist iterator, see project)

Backward analysis

Forward versus backward analysis

Example

$Y \leftarrow 0;$
while $Y \leq X$ **do** $Y \leftarrow Y + 1$

Forward analysis:

- given $X \in [-10, 10]$ at the beginning of the program
  $Y \in [0, 11]$ at the end of the program

Backward analysis:

- to have $Y \in [10, 20]$ at the end of the program
  we must have $X \in [9, 19]$ at the beginning of the program

Backward-forward combination

Goal: given initial states I and finial states F
consider only executions that start in I and end in F

Application: analysis specialization to remove false alarms

Example

$X \leftarrow \textbf{rand}(-100, 100);$
$\textbf{if } X = 0 \textbf{ then } X \leftarrow 1;$
• $Y \leftarrow 100/X$

Analysis: using the interval domain

- a forward analysis finds $X \in [-100, 100]$ at •
  $\Longrightarrow$ false alarm for division by zero
- backward analysis from • assuming $X = 0$
  we find $\bot$ at the program entry
  $\Longrightarrow$ no execution can trigger the division by zero
  (we have removed the false alarm)

more complex combinations exist, such as iterated forward and backward analyses

Reminder: Forward denotational concrete semantics

$\mathsf{S}[\![\, stat \,]\!] : \mathcal{P}(\mathcal{E}) \to \mathcal{P}(\mathcal{E})$

$$
\begin{array}{lcl}
\mathsf{S}[\![\, \mathbf{skip} \,]\!] \, R & \stackrel{\text{def}}{=} & R \\
\mathsf{S}[\![\, s_1; \; s_2 \,]\!] \, R & \stackrel{\text{def}}{=} & \mathsf{S}[\![\, s_2 \,]\!] \, (\mathsf{S}[\![\, s_1 \,]\!] \, R) \\
\mathsf{S}[\![\, V \leftarrow e \,]\!] \, R & \stackrel{\text{def}}{=} & \{ \, \rho[V \mapsto v] \, | \, \rho \in R, \; v \in \mathsf{E}[\![\, e \,]\!] \, \rho \, \} \\
\mathsf{S}[\![\, c? \,]\!] \, R & \stackrel{\text{def}}{=} & \{ \, \rho \in R \, | \, \mathsf{true} \in \mathsf{C}[\![\, c \,]\!] \, \rho \, \} \\
\mathsf{S}[\![\, \mathbf{if} \; c \; \mathbf{then} \; s_1 \; \mathbf{else} \; s_2 \,]\!] \, R & \stackrel{\text{def}}{=} & \mathsf{S}[\![\, s_1 \,]\!] \, (\mathsf{S}[\![\, c? \,]\!] \, R) \cup \mathsf{S}[\![\, s_2 \,]\!] \, (\mathsf{S}[\![\, \neg c? \,]\!] \, R) \\
\mathsf{S}[\![\, \mathbf{while} \; c \; \mathbf{do} \; s \,]\!] \, R & \stackrel{\text{def}}{=} & \mathsf{S}[\![\, \neg c? \,]\!] \, (\mathrm{lfp} \; \lambda I. R \cup \mathsf{S}[\![\, s \,]\!] \, (\mathsf{S}[\![\, c? \,]\!] \, I))
\end{array}
$$

- $\mathsf{S}[\![\, stat \,]\!] \, R$
  set of all possible states at the program end
  when starting in a state in R

Backward denotational concrete semantics

$\overleftarrow{S}[\![\, stat \,]\!] : \mathcal{P}(\mathcal{E}) \to \mathcal{P}(\mathcal{E})$

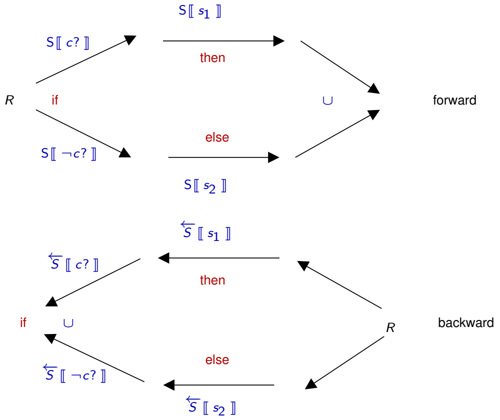
$$
\begin{array}{lcl}
\overleftarrow{S}[\![\, \textbf{skip} \,]\!]\, F & \stackrel{\text{def}}{=} & F \\
\overleftarrow{S}[\![\, s_1;\ s_2 \,]\!]\, F & \stackrel{\text{def}}{=} & \overleftarrow{S}[\![\, s_1 \,]\!]\,(\overleftarrow{S}[\![\, s_2 \,]\!]\, F) \\
\overleftarrow{S}[\![\, V \leftarrow e \,]\!]\, F & \stackrel{\text{def}}{=} & \{\, \rho \mid \exists v \in \mathrm{E}[\![\, e \,]\!]\, \rho \colon \rho[V \mapsto v] \in F \,\} \\
\overleftarrow{S}[\![\, c? \,]\!]\, F & \stackrel{\text{def}}{=} & \{\, \rho \in F \mid \text{true} \in \mathrm{C}[\![\, c \,]\!]\, \rho \,\} \\
\overleftarrow{S}[\![\, \textbf{if } c \textbf{ then } s_1 \textbf{ else } s_2 \,]\!]\, F & \stackrel{\text{def}}{=} & \overleftarrow{S}[\![\, c? \,]\!]\,(\overleftarrow{S}[\![\, s_1 \,]\!]\, F) \cup \overleftarrow{S}[\![\, \neg c? \,]\!]\,(\overleftarrow{S}[\![\, s_2 \,]\!]\, F) \\
\overleftarrow{S}[\![\, \textbf{while } c \textbf{ do } s \,]\!]\, F & \stackrel{\text{def}}{=} & \mathrm{lfp}\ \lambda I.\, \overleftarrow{S}[\![\, \neg c? \,]\!]\, F \cup \overleftarrow{S}[\![\, c? \,]\!]\,(\overleftarrow{S}[\![\, s \,]\!]\, I)
\end{array}
$$

- $\overleftarrow{S}[\![\, stat \,]\!]\, F$
  set of all the states at the program entry
  such that at least one execution ends in a state in F
- $\iota \in \overleftarrow{S}[\![\, stat \,]\!]\, \{\phi\} \iff \phi \in \mathrm{S}[\![\, stat \,]\!]\, \{\iota\}$

Note: – the order of statements reversed (s_2 before s_1, s_1 before $c?$, etc.)
– $\overleftarrow{S}[\![\, c? \,]\!]$ is unchanged

Concrete semantics: flow intuition

Intuition: information propagation for **if** $\cdots$ **then** $\cdots$ **else**



$$\mathsf{S}[\![\,\textbf{if } c \textbf{ then } s_1 \textbf{ else } s_2\,]\!]\,R = \mathsf{S}[\![\,s_1\,]\!]\,(\mathsf{S}[\![\,c?\,]\!]\,R) \cup \mathsf{S}[\![\,s_2\,]\!]\,(\mathsf{S}[\![\,\neg c?\,]\!]\,R)$$

$$\overleftarrow{S}[\![\,\textbf{if } c \textbf{ then } s_1 \textbf{ else } s_2\,]\!]\,F = \overleftarrow{S}[\![\,c?\,]\!]\,(\overleftarrow{S}[\![\,s_1\,]\!]\,F) \cup \overleftarrow{S}[\![\,\neg c?\,]\!]\,(\overleftarrow{S}[\![\,s_2\,]\!]\,F)$$

Backward abstraction denotational semantics

Goal: construct $\overleftarrow{S}^\sharp[\![\, stat \,]\!]$ that soundly approximates $\overleftarrow{S}[\![\, stat \,]\!]$

We can define, by induction:

$\overleftarrow{S}^\sharp[\![\, \textbf{skip} \,]\!]\, F^\sharp \stackrel{\text{def}}{=} F^\sharp$

$\overleftarrow{S}^\sharp[\![\, s_1;\ s_2 \,]\!]\, F^\sharp \stackrel{\text{def}}{=} \overleftarrow{S}^\sharp[\![\, s_1 \,]\!]\, (\overleftarrow{S}^\sharp[\![\, s_2 \,]\!]\, F^\sharp)$

$\overleftarrow{S}^\sharp[\![\, c? \,]\!]\, F^\sharp \stackrel{\text{def}}{=} \mathsf{S}^\sharp[\![\, c? \,]\!]\, F^\sharp$

$\overleftarrow{S}^\sharp[\![\, \textbf{if}\ c\ \textbf{then}\ s_1\ \textbf{else}\ s_2 \,]\!]\, F^\sharp \stackrel{\text{def}}{=} \overleftarrow{S}^\sharp[\![\, c? \,]\!]\, (\overleftarrow{S}^\sharp[\![\, s_1 \,]\!]\, F^\sharp) \cup^\sharp \overleftarrow{S}^\sharp[\![\, \neg c? \,]\!]\, (\overleftarrow{S}^\sharp[\![\, s_2 \,]\!]\, F^\sharp)$

$\overleftarrow{S}^\sharp[\![\, \textbf{while}\ c\ \textbf{do}\ s \,]\!]\, F^\sharp \stackrel{\text{def}}{=} \lim \lambda I^\sharp. I^\sharp \mathbin{\triangledown} (\overleftarrow{S}^\sharp[\![\, \neg c? \,]\!]\, F^\sharp \cup^\sharp \overleftarrow{S}^\sharp[\![\, c? \,]\!]\, (\overleftarrow{S}^\sharp[\![\, s \,]\!]\, I^\sharp))$

Abstract operators:

- we can reuse $\cup^\sharp$, $\triangledown$ and $\mathsf{S}^\sharp[\![\, c? \,]\!]$
- only $\overleftarrow{S}^\sharp[\![\, V \leftarrow e \,]\!]$ needs to be defined on a per-domain basis
- assuming forward-backward combination, we can use the pre-condition $X^\sharp$ discovered in the forward phase:

  $$\overleftarrow{S}^\sharp[\![\, X \leftarrow e \,]\!]\, (X^\sharp, F^\sharp) \text{ approximates } \gamma(X^\sharp) \cap \overleftarrow{S}[\![\, X \leftarrow e \,]\!]\, \gamma(F^\sharp)$$

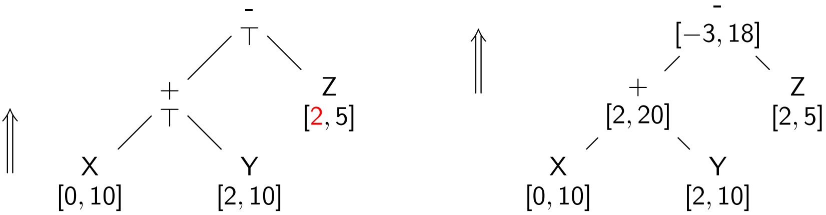
  (makes $\overleftarrow{S}^\sharp[\![\, X \leftarrow e \,]\!]$ easier to implement and more precise: see next slide)

Backward interval assignment

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

- before the assignment $X^\sharp = \{\, X \mapsto [0,10], Y \mapsto [2,10], Z \mapsto [1,5] \,\}$
- after the assignment $F^\sharp = \{\, X \mapsto [-6,6], Y \mapsto [2,10], Z \mapsto [2,6] \,\}$
- returns: subset $X^{\sharp\prime}$ of $X^\sharp$ that result in $F^\sharp$ after assignment

Similar to test.
Firstly: bottom-up evaluation

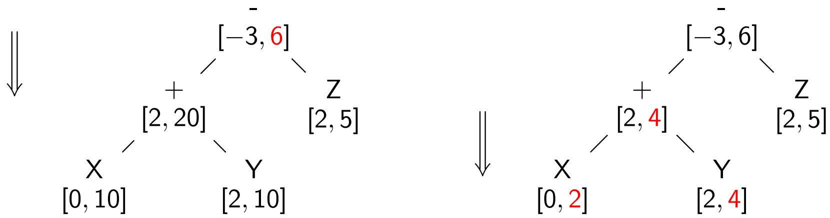


Backward interval assignment

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

- before the assignment $X^\sharp = \{ X \mapsto [0, 10], Y \mapsto [2, 10], Z \mapsto [1, 5] \}$
- after the assignment $F^\sharp = \{ X \mapsto [-6, 6], Y \mapsto [2, 10], Z \mapsto [2, 6] \}$
- returns: subset $X^{\sharp\prime}$ of $X^\sharp$ that result in $F^\sharp$ after assignment

Similar to test.
Secondly: top-down refinement



returns $X^{\sharp\prime} = \{ X \mapsto [0, 2], Y \mapsto [2, 4], Z \mapsto [2, 5] \}$

Conclusion

Conclusion

Summary:

- systematic design of abstract operators (Galois connection)
- optimal and non-optimal (practical) abstractions
- abstract tests through abstract refinement operators
- backward assignment
- fixpoint approximation by iteration with widening
  $\Longrightarrow$ ensure termination even for infinite-height domains!
- application to interval analysis
  (but can be used on any non-relational analysis, e.g., constants)

Next lecture: relational domains (polyhedra)

Practical session: implement the interval domain

(also useful for the project)

TP implementation suggestions

Summary of the (forward) abstract semantics

$$S^\sharp[\![\,\textbf{skip}\,]\!]\,X^\sharp \stackrel{\text{def}}{=} X^\sharp$$

$$S^\sharp[\![\,s_1; s_2\,]\!]\,X^\sharp \stackrel{\text{def}}{=} S^\sharp[\![\,s_2\,]\!]\,(S^\sharp[\![\,s_1\,]\!]\,X^\sharp)$$

$$S^\sharp[\![\,V \leftarrow e\,]\!]\,X^\sharp \stackrel{\text{def}}{=} \begin{cases} X^\sharp[V \mapsto E^\sharp[\![\,e\,]\!]\,X^\sharp] & \text{if } E^\sharp[\![\,e\,]\!]\,X^\sharp \neq \bot \\ \dot{\bot} & \text{if } E^\sharp[\![\,e\,]\!]\,X^\sharp = \bot \end{cases}$$

$$S^\sharp[\![\,\textbf{if } c \textbf{ then } s_1 \textbf{ else } s_2\,]\!]\,X^\sharp \stackrel{\text{def}}{=} S^\sharp[\![\,s_1\,]\!]\,(S^\sharp[\![\,c?\,]\!]\,X^\sharp)\ \dot{\cup}^\sharp\ S^\sharp[\![\,s_2\,]\!]\,(S^\sharp[\![\,\neg c?\,]\!]\,X^\sharp)$$

$$S^\sharp[\![\,\textbf{while } c \textbf{ do } s\,]\!]\,X^\sharp \stackrel{\text{def}}{=} S^\sharp[\![\,\neg c?\,]\!]\,(\lim \lambda I^\sharp.I^\sharp\ \dot{\triangledown}\ (X^\sharp\ \dot{\cup}^\sharp\ S^\sharp[\![\,s\,]\!]\,(S^\sharp[\![\,c?\,]\!]\,I^\sharp)))$$

$E^\sharp[\![\,e\,]\!]$ by induction on the syntax of expressions

$S^\sharp[\![\,c?\,]\!]$ by bottom-up evaluation followed by top-down refinement
(for the project only, not required in the practical session)

Value domain signature

```
module type VALUE_DOMAIN = sig
  type t                              // { [a,b] | a ∈ ℤ ∪ {−∞}, b ∈ ℤ ∪ {+∞}, a ≤ b } ∪ {⊥}
  (* constructors *)
  val top:   t                        // [−∞, +∞]
  val bottom:   t                     // ⊥
  val const:   int -> t               // c ↦ [c, c]
  val rand:   int -> int -> t         // l ↦ h ↦ [l, h]
  (* order *)
  val subset:   t -> t -> bool        // ⊑
  (* set-theoretic operations *)
  val join:   t -> t -> t             // ∪♯
  val meet:   t -> t -> t             // ∩♯
  val widen:   t -> t -> t            // ▽
  (* arithmetic operations *)
  val neg:   t -> t                   // unary −♯
  val add:   t -> t -> t              // +♯
  val sub:   t -> t -> t              // −♯
  val mul:   t -> t -> t              // ×♯
  val div:   t -> t -> t              // /♯
  (* boolean test *)
  val leq:   t -> t -> t * t          // [a,b] ↦ [c,d] ↦ ([a, min(b,d)], [max(a,c), d])
end
```

Environment domain signature

```
module type ENVIRONMENT_DOMAIN = sig
  type t                                          // ℰ♯
  (* constructors *)
  val init:  id list -> t                         // ∀V ∈ 𝕍: ρ(V) = 0
  (* abstract operators *)
  val assign:  t -> id -> expr -> t               // S♯⟦ id ← expr ⟧
  val compare:  t -> expr -> expr -> t            // S♯⟦ expr ≤ expr? ⟧
  (* set-theoretic operations *)
  val join:  t -> t -> t                          // ∪̇♯
  val meet:  t -> t -> t                          // ∩̇♯
  val widen:  t -> t -> t                         // ∇̇
  (* order *)
  val subset:  t -> t -> bool                     // ⊑̇
end
```

Environment domain implementation details

```
module NonRelational(V : VALUE_DOMAIN) = (struct
  module Map = Mapext.Make                          // maps
    (struct type t = id let compare = compare end)
  type env = V.t Map.t                              // 𝕍 → (𝔻 \ {⊥})
  type t = Env of env | BOT                         // ℰ♯ ≝ (𝕍 → (𝔻 \ {⊥})) ∪ {⊥̇}
  (* utilities *)
  val eval:  env -> expr -> V.t                     // E♯⟦ expr ⟧
  val is_bot:  V.t -> bool                          // whether γ(v♯) = ∅
  val strict:  (env -> t) -> t -> t                 // maps ⊥̇ to ⊥̇
  (* operators *)
  let join a b = match a,b with                     // ∪̇♯
  | BOT,x | x,BOT -> x
  | Env m,Env n -> Env (Map.map2z (fun _ x y -> V.join x y) m n)
  ...
end:  ENVIRONMENT_DOMAIN)
```

Generic functor to lift a `VALUE_DOMAIN` to an `ENVIRONMENT_DOMAIN`

Uses a Map as data-structure for environment (functional array)

and a binary map iterator `map2z f`
(optimized for idempotent functions: $\forall x$: $\texttt{f}\ k\ x\ x = x$)