Abstract Interpretation II

Semantics and Application to Program Verification

Antoine Miné

École normale supérieure, Paris
year 2013–2014

Course 10

7 May 2014

Overview

Program: Abstract interpretation

continue on the static analysis by abstract interpretation
using more a complex non-relational domain:
the interval domain

- Interval abstraction
  systematic design of abstract operators
  soundness, optimality, exactness
- Analyzing a more general language
  more general assignments and tests
  assertions
  local variables
- Convergence acceleration
- Practical session
  implement the interval domain for the project
  functorize the analysis for improved modularity

Interval abstraction

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

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

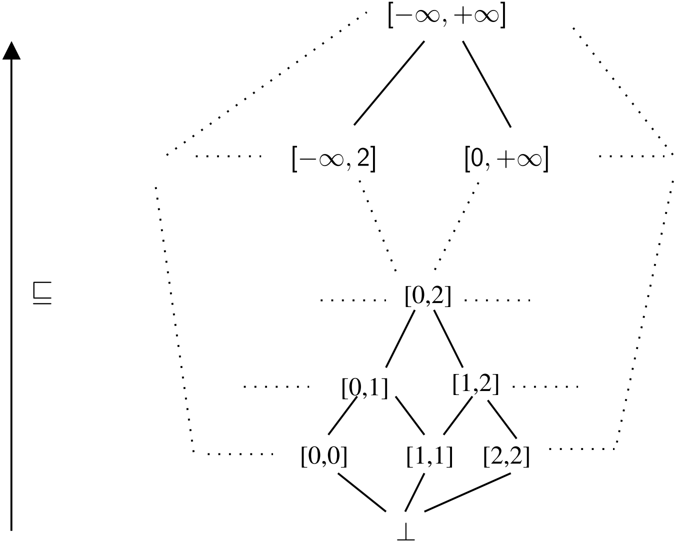
  $\implies$ any integer set may be over-approximated in $\mathbb{I}$

  (we can always resort to $[-\infty,+\infty]$, i.e., $\top$)

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

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

Interval lattice



Algebraic structure

partial order: $\sqsubseteq$

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

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

lattice structure: $\sqcup$, $\sqcap$

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

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

Algebraic structure

Notes:

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

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

$\implies$ 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; $[0, 0] \sqcup [1, 1] = [0, 2]$

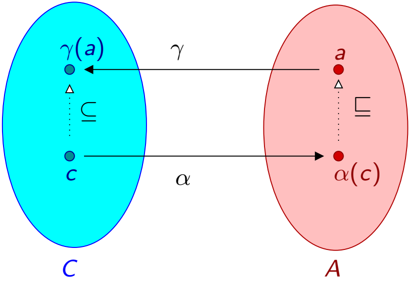
- $\sqcup$ and $\sqcap$ are not distributive
  $([0, 0] \sqcup [2, 2]) \sqcap [1, 1] = [0, 2] \sqcap [1, 1] = [1, 1]$
  but $([0, 0] \sqcap [1, 1]) \sqcup ([2, 2] \sqcap [1, 1]) = \emptyset \sqcup \emptyset = \emptyset$

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

Reminder: Galois Connection

Galois Connection $(C, \subseteq) \xleftarrow[\alpha]{\gamma} (A, \sqsubseteq)$

- $\alpha : C \to A$ (abstraction)
- $\gamma : A \to C$ (concretization)
- $\forall c \in C,\ a \in A$: $\alpha(c) \sqsubseteq a \iff c \subseteq \gamma(a)$ (duality)



$\alpha(c)$ is the best abstraction in A of $c \in C$

$\alpha(c)$ is the smallest a for $\sqsubseteq$ that over-approximates c, i.e., such that $c \subseteq \gamma(a)$

Interval Galois connection

Interval Galois connection:

- $\begin{cases} \gamma(\bot) \stackrel{\text{def}}{=} \emptyset \\ \gamma([a,b]) \stackrel{\text{def}}{=} \{\, x \in \mathbb{Z} \mid a \leq x \leq b \,\} \end{cases}$
- $\alpha(X) \stackrel{\text{def}}{=} \begin{cases} \bot & \text{if } X = \emptyset \\ [\min X, \max X] & \text{if } X \neq \emptyset \end{cases}$

Proof:

$\alpha(X) \sqsubseteq (a,b)$

$\iff \min X \geq a \wedge \max X \leq b$ *(def. α, $\sqsubseteq$)*

$\iff \forall x \in X : a \leq x \leq b$ *(def. min, max)*

$\iff \forall x \in X : x \in \{\, y \mid a \leq y \leq b \,\}$

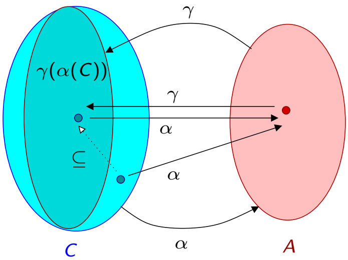
$\iff \forall x \in X : x \in \gamma([a,b])$ *(def. γ)*

$\iff X \subseteq \gamma([a,b])$ *(prop. $\subseteq$)*

Side-note: Galois Insertion

In fact: γ is one-to-one, α is onto, $\alpha \circ \gamma = id$
$\implies$ A is isomorphic to a subset of C: $A \simeq \gamma(A) = \gamma(\alpha(C)) \subseteq C$

We call it a Galois Insertion, denoted: as $(C, \subseteq) \underset{\alpha}{\overset{\gamma}{\leftrightarrows}} (A, \sqsubseteq)$



Otherwise, several abstract elements may represent the same concrete one

Example: alternate intervals $\mathbb{J} \stackrel{\text{def}}{=} \{ [a, b] \mid a, b \in \mathbb{Z} \cup \{-\infty, +\infty\} \}$
$\mathbb{J}$ has several abstractions of $\emptyset$, i.e., any $[a, b]$ when $a > b$;
the "best" one is $\alpha(\emptyset) = [\max \emptyset, \min \emptyset] = [+\infty, -\infty]$

Reminder: Sound, optimal, exact abstractions

Given $F : C \to C$, how to construct $F^\sharp : A \to A$?

Soundness: core property of abstract operators

$F^\sharp$ over-approximates F in the abstract, formally:

$$\forall a \in A: F(\gamma(a)) \subseteq \gamma(F^\sharp(a))$$

$\implies$ any property proved with $F^\sharp$ is also true of F

Example property:

we want to prove $F(c) \subseteq c'$, i.e., $\{c\}\ F\ \{c'\}$
assuming $c = \gamma(a)$, $c' = \gamma(a')$, we only need to prove $F^\sharp(a) \sqsubseteq a'$

Proof

$F^\sharp(a) \sqsubseteq a'$
$\implies \gamma(F^\sharp(a)) \subseteq \gamma(a')$ (by monotony of γ)
$\implies F(\gamma(a)) \subseteq \gamma(a')$ (by soundness of $F^\sharp$)
$\implies F(c) \subseteq c'$

The converse does not hold:
we can have $F(c) \subseteq c'$ but $F^\sharp(a) \not\sqsubseteq a'$
$\implies$ in this case, the property cannot be proved in the abstract

Reminder: Sound, optimal, exact abstractions

Exactness: $\forall a \in A: F(\gamma(a)) = \gamma(F^\sharp(a))$

- quite rare: $\forall a \in A: F(\gamma(a))$ must be exactly representable in A
- even if it exists, such a $F^\sharp$ may be difficult to compute!

Optimality: $\forall a \in A: \gamma(F^\sharp(a)) = \min_\subseteq \{\, \gamma(a') \mid F(\gamma(a)) \subseteq \gamma(a') \,\}$

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

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

(by definition of Galois connections)

which implies $\gamma(F^\sharp(a)) = \min_\subseteq \{\, \gamma(a') \mid F(\gamma(a)) \subseteq \gamma(a') \,\}$

(by monotony of γ)

Notes:

- $\min_\sqsubseteq$ is slightly stronger than $\min_\subseteq$
- α provides a systematic way to design $F^\sharp$
- when no α exist, soundness, optimality, exactness can still be defined

Concrete integer operations

Goal: design interval versions of core operators

building blocks to design an interval semantics

Concrete arithmetic operators:

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

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

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

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

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

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

- $=^\sharp$ 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 $+^\sharp$

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

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

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

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

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

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

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

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

- 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

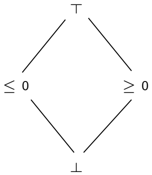
- simplifies analysis design and enhances reusability
- but can degrade the precision

Side-note: Meet closure and optimality

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

$\Longrightarrow \{\, \gamma(a) \mid a \in A \,\}$ 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 $\{0\}$

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

$\Longrightarrow \{ \gamma(a) \mid a \in A \}$ 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

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

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

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

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

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

Interval analysis

Language

Expressions and conditions

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

Statements

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

Concrete semantics

Classic non-deterministic concrete semantics, in denotational style:

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

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

$\mathsf{C}[\![\, cond \,]\!] : \mathcal{E} \to \mathcal{P}(\{\text{true}, \text{false}\})$ *(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 $\mathcal{E} \stackrel{\text{def}}{=} \mathbb{V} \to \mathbb{Z}$

Concrete semantics

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

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

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 abstractions

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: $X_1^\sharp \dot{\sqsubseteq} X_2^\sharp \iff \forall V \in \mathbb{V}: X_1^\sharp(V) \sqsubseteq X_2^\sharp(V)$
- join: $X_1^\sharp \dot{\sqcup} X_2^\sharp \stackrel{\text{def}}{=} \lambda V. X_1^\sharp(V) \sqcup X_2^\sharp(V)$
- meet: $X_1^\sharp \dot{\sqcap} X_2^\sharp \stackrel{\text{def}}{=} \lambda V. X_1^\sharp(V) \sqcap X_2^\sharp(V)$

$\Longrightarrow$ we still have a complete lattice

Cartesian Galois connection:

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

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

Side-note: Coalescent $\bot$

Note: $(\mathcal{P}(\mathbb{V} \to \mathbb{Z}), \subseteq) \underset{\dot{\alpha}}{\overset{\dot{\gamma}}{\leftrightarrows}} (\mathbb{V} \to \mathbb{I}, \dot{\sqsubseteq})$ is not a Galois insertion

$\emptyset$ has several representations:

(e.g., $\gamma([X \mapsto \bot, Y \mapsto [0,1]]) = \gamma([X \mapsto [0,1], Y \mapsto \bot]) = \emptyset$)

there is a canonical representation: $\dot{\bot} \overset{\text{def}}{=} \lambda V \in \mathbb{V}.\bot$

(smallest for $\dot{\sqsubseteq}$)

to obtain a Galois insertion, we **coalesce** all representations of $\emptyset$:

$$\mathcal{E}^\sharp \overset{\text{def}}{=} (\mathbb{V} \to (\mathbb{I} \setminus \{\bot\})) \cup \{\dot{\bot}\}$$

Benefits:

- $=^\sharp$ is reduced to testing physical equality

  (no special case for $\emptyset$)

- the optimal $\dot{\cup}^\sharp$ is reduced to $\dot{\sqcup}$

  without coalescing, we get:
  $[X \mapsto \bot, Y \mapsto [0,1]] \mathbin{\dot{\sqcup}} [X \mapsto [0,1], Y \mapsto \bot] = [X \mapsto [0,1], Y \mapsto [0,1]]$

  but $\gamma([X \mapsto \bot, Y \mapsto [0,1]]) \cup \gamma([X \mapsto [0,1], Y \mapsto \bot]) = \emptyset$

Interval expression evaluation

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

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

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

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

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 $-$, $\times$, $/$):

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

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

$S^\sharp[\![\, stat \,]\!] : \mathcal{E}^\sharp \to \mathcal{E}^\sharp$

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)

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

Soundess 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:

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

Interval test

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.)

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

Boolean operators in c are handled 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 interval tests: assuming $X^\sharp(V) = [a,b]$ and $X^\sharp(W) = [c,d]$

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

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

For other tests, fall-back to $\mathsf{S}^\sharp[\![\, c?\,]\!]\, X^\sharp \stackrel{\text{def}}{=} X^\sharp$

Soundness proof for the fall-back operator

$\forall R: \mathsf{S}[\![\, c?\,]\!]\, R \subseteq R$, hence $\forall X^\sharp: \mathsf{S}[\![\, c?\,]\!]\, (\dot{\gamma}(X^\sharp)) \subseteq \dot{\gamma}(X^\sharp) = \dot{\gamma}(\mathsf{S}^\sharp[\![\, c?\,]\!]\, X^\sharp)$

Interval conditionals

Concrete semantics:

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

Abstract semantics: compose existing abstract operators:

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

Soundness proof:

by soundness of the composition of sound operators

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

given $E^\sharp \overset{\text{def}}{=} [V \mapsto [-\infty, +\infty]]$
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

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 $(X_n^\sharp)_{n\in\mathbb{N}}$, the sequence $(Y_n^\sharp)_{n\in\mathbb{N}}$

$$\begin{cases} Y_0^\sharp & \stackrel{\text{def}}{=} X_0^\sharp \\ Y_{n+1}^\sharp & \stackrel{\text{def}}{=} Y_n^\sharp \triangledown X_{n+1}^\sharp \end{cases}$$

stabilizes in finite time: $\exists N \in \mathbb{N}$: $Y_N^\sharp = Y_{N+1}^\sharp$

Fixpoint approximation theorem:

- the sequence $X_{n+1}^\sharp \stackrel{\text{def}}{=} X_n^\sharp \triangledown F^\sharp(X_n^\sharp)$ stabilizes in finite time
- when $X_{N+1}^\sharp \sqsubseteq X_N^\sharp$, then $X_N^\sharp$ abstracts $\operatorname{lfp} F$

Soundness proof: assume $X_{N+1}^\sharp \sqsubseteq X_N^\sharp$, then

$\gamma(X_N^\sharp) \supseteq \gamma(X_{N+1}^\sharp) = \gamma(X_N^\sharp \triangledown F^\sharp(X_N^\sharp)) \supseteq \gamma(F^\sharp(X_N^\sharp)) \supseteq F(\gamma(X_N^\sharp))$

$\gamma(X_N^\sharp)$ is a post-fixpoint of F, but $\operatorname{lfp} F$ is F's least post-fixpoint, so, $\gamma(X_N^\sharp) \supseteq \operatorname{lfp} F$

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 $\triangledown$

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

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

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

- $\text{lfp}\ F$ defines the most precise invariant (concrete semantics)
- X such that $\text{lfp}\ F \subseteq X$ 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 $\text{lfp}\ F$)
- $X^\sharp$ 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 $\text{lfp}\ F$)

Non-monotonicity of widening

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

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

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

$\triangledown$ 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: $\bot$, $[1,1]$, $[1,+\infty]$

  $([1,1] \triangledown F^\sharp([1,1],[1,1]) = [1,1] \triangledown ([1,1] \cup^\sharp [3,3]) = [1,1] \triangledown [1,3] = [1,+\infty])$

  $\implies \mathsf{S}^\sharp[\![\, stat \,]\!]\,([1,1]) = [51,+\infty]$

- if $X^\sharp = [1,52]$, $F^\sharp$'s iterates are: $\bot$, $[1,52]$, $[1,52]$

  $([1,52] \triangledown F^\sharp([1,52],[1,52]) = [1,52] \triangledown ([1,1] \cup^\sharp [3,52]) = [1,52] \triangledown [1,52] = [1,52])$

  $\implies \mathsf{S}^\sharp[\![\, stat \,]\!]\,([1,52]) = [51,52]$

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

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

Summary of the abstract semantics

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

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

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

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

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

(next slides: extending the language with assertions and local variables)

Assertions

Assertion statement: **assert**$_\ell$ *cond*

- checks that the boolean condition is satisfied
- if satisfied, continue program execution
- if not, abort the program with an error at location ℓ

Semantics: returns a value and has an effect

$\mathsf{S}[\![\,\textbf{assert}_\ell\ c\,]\!]\,R \stackrel{\text{def}}{=}$
  if $\mathsf{S}[\![\,\neg c?\,]\!]\,R \neq \emptyset$ then **print** "error at ℓ";
  $\mathsf{S}[\![\,c?\,]\!]\,R$

$\mathsf{S}^\sharp[\![\,\textbf{assert}_\ell\ c\,]\!]\,X^\sharp \stackrel{\text{def}}{=}$
  if $\mathsf{S}^\sharp[\![\,\neg c?\,]\!]\,X^\sharp \neq \dot\perp$ then **print** "error at ℓ";
  $\mathsf{S}^\sharp[\![\,c?\,]\!]\,X^\sharp$

Notes:

- due to non-determinism, ρ can satisfy both c and $\neg c$
- if **assert** occurs within a loop, avoid printing at each iterate; print only after the iterates stabilize!

Variable creation and destruction

In practice, the set $\mathbb{V}$ of variables varies during program execution

Local variables: **local** V **in** *stat*

Add a new variable V, then execute *stat*, then remove V

$\Longrightarrow$ the set of variables at each statement is uniquely defined by the nesting of **local** statements

(to simplify, we assume that all variables V appearing in **local** statements are distinct)

$$\begin{array}{rll} & \mathsf{S}[\![\, \textbf{local } V \textbf{ in } s \,]\!] & \stackrel{\text{def}}{=} \mathsf{S}[\![\, \textbf{del } V \,]\!] \circ \mathsf{S}[\![\, s \,]\!] \circ \mathsf{S}[\![\, \textbf{add } V \,]\!] \\ \text{where} & \mathsf{S}[\![\, \textbf{add } V \,]\!]\, R & \stackrel{\text{def}}{=} \{\, \rho[V \mapsto v] \mid \rho \in R,\ v \in \mathbb{Z} \,\} \\ \text{and} & \mathsf{S}[\![\, \textbf{del } V \,]\!]\, R & \stackrel{\text{def}}{=} \{\, \rho_{|\mathrm{dom}(\rho) \setminus \{V\}} \mid \rho \in R \,\} \end{array}$$

Abstraction:

$$\begin{array}{rll} & \mathsf{S}^\sharp[\![\, \textbf{local } V \textbf{ in } s \,]\!] & \stackrel{\text{def}}{=} \mathsf{S}^\sharp[\![\, \textbf{del } V \,]\!] \circ \mathsf{S}^\sharp[\![\, s \,]\!] \circ \mathsf{S}^\sharp[\![\, \textbf{add } V \,]\!] \\ \text{where} & \mathsf{S}^\sharp[\![\, \textbf{add } V \,]\!]\, X^\sharp & \stackrel{\text{def}}{=} X^\sharp[V \mapsto [-\infty, +\infty]] \\ \text{and} & \mathsf{S}^\sharp[\![\, \textbf{del } V \,]\!]\, X^\sharp & \stackrel{\text{def}}{=} X^\sharp_{|\mathrm{dom}(X^\sharp) \setminus \{V\}} \end{array}$$

Project: implementation suggestions

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 cost:  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
```

Interval domain implementation details

```
module Intervals = (struct
  type bound = Int of int | PINF | MINF            // ℤ ∪ {+∞, −∞}
  type t = Itv of bound * bound | BOT              // { [a, b] | a ≤ b } ∪ {⊥}

  (* utilities *)
  val bound_cmp:  bound -> bound -> int            // as OCaml's compare
  val bound_neg:  bound -> bound                   // unary −
  val bound_add:  bound -> bound -> bound          // +
  ...
  val strict:  (bound -> bound -> t) -> t -> t     // maps ⊥ to ⊥

  (* domain implementation *)
  let neg = strict                                 // unary −♯
    (fun a b -> Itv (bound_neg b, bound_neg a))

  let subset a b = match a,b with                  // ⊑
  | BOT,_ -> true | _,BOT -> false
  | Itv (a,b), Itv (c,d) ->
      bound_cmp a c >= 0 && bound_cmp b d >= 0

  ...
end:  VALUE_DOMAIN)
```

Environment domain signature

```
module type ENVIRONMENT_DOMAIN = sig
  type t                                      // ℰ♯
  (* constructors *)
  val init:  t                                // 𝕍 = ∅
  (* variable management *)
  val add_variable:  t -> id -> t             // S♯⟦ add id ⟧
  val remove_variable:  t -> id -> t          // S♯⟦ del id ⟧
  (* 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: $\forall x$: f $k\ x\ x = x$)

More on loop analysis

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
$\implies X \geq 0$ when *stat* is executed

Imprecision:

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

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

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

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

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

$Y^\sharp \stackrel{\text{def}}{=} S^\sharp[\![\neg c?]\!] A^\sharp \cup^\sharp (\cup_{n<N}^\sharp S^\sharp[\![\neg c?]\!] U_n^\sharp)$ (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 $Y_n^\sharp \stackrel{\text{def}}{=} F^{\sharp n}(X^\sharp)$
  by soundness, $\gamma(Y_n^\sharp)$ is also a post-fixpoint of F for every n
  we stop after a fixed finite n, or when $Y_{n+1}^\sharp = Y_n^\sharp$

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

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

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 $[0, 40]$

Conclusion

Conclusion

Summary:

- systematic design of abstract operators (Galois connection)
- optimal and non-optimal (practical) abstractions
- fixpoint approximation by iteration with widening
  $\Longrightarrow$ ensure termination even for infinite-height domains!

Next lecture: relational domains

(polyhedra)

Practical session: implement the interval domain