Principles of Abstract Interpretation
MIT press

Ch. 33, Static interval analysis

Patrick Cousot

pcousot.github.io

PrAbsInt@gmail.com github.com/PrAbsInt/

These slides are available at
http://github.com/PrAbsInt/slides/slides-33--interval-analysis-PrAbsInt.pdf

Chapter 33

Ch. **33**, Static interval analysis

Interval domain

Interval abstract domain

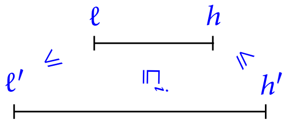
The interval abstract domain $\langle \mathbb{P}^i, \sqsubseteq^i, \perp^i, \top^i, \sqcup^i, \sqcap^i \rangle$ is

- The set $\mathbb{P}^i \triangleq \{[\ell, h] \mid \ell \in \mathbb{Z} \cup \{-\infty\} \wedge \ell \leqslant h \wedge h \in \mathbb{Z} \cup \{\infty\}\} \cup \{\perp^i\}$ of numerical intervals, where $\perp^i = \varnothing$, and excluding $[-\infty, -\infty]$ and $[\infty, \infty]$;
- The empty interval $\perp^i = \varnothing$ can be encoded by any $[\ell, h]$ with $h < \ell$ (*e.g.* normalized to $[\infty, -\infty]$);

Interval abstract domain (cont'd)

The interval abstract domain is a complete lattice

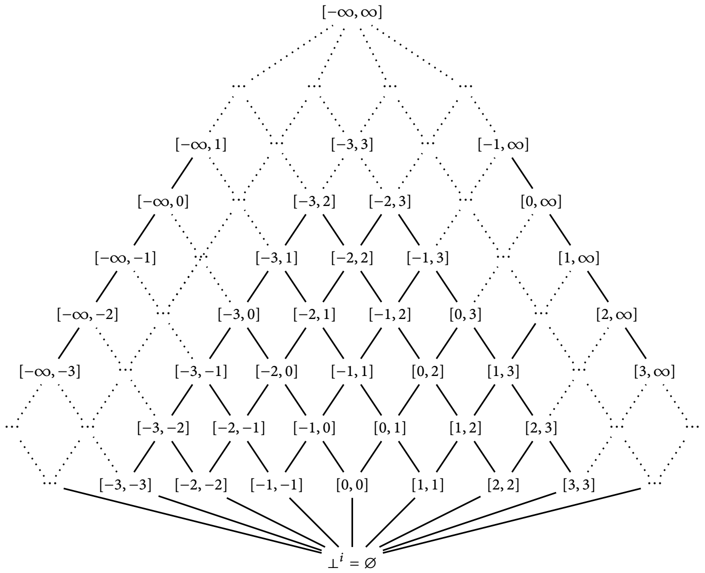
- The partial order $\sqsubseteq^i$ on $\mathbb{P}^i$ is interval inclusion $\bot^i \sqsubseteq^i \bot^i \sqsubseteq^i \langle \ell,\, h\rangle \sqsubseteq^i \langle \ell',\, h'\rangle$ if and only if $\ell' \leqslant \ell \leqslant h \leqslant h'$;



- The infimum is the empty interval $\bot^i = \varnothing$,
- The supremum is $\top^i = [-\infty, \infty]$,
- The join $\bigsqcup^i_{i\in\Delta}[\ell_i, h_i] \triangleq [\min_{i\in\Delta}(\ell_i), \max_{i\in\Delta}(h_i)]$,
- The meet $\bigsqcap^i_{i\in\Delta}[\ell_i, h_i] \triangleq [\max_{i\in\Delta}(\ell_i), \min_{i\in\Delta}(h_i)]$,

where $\min \mathbb{Z} \triangleq -\infty$, $\max \mathbb{Z} \triangleq \infty$, $\min \varnothing \triangleq \infty$, $\max \varnothing \triangleq -\infty$, and $[\ell, h] = \varnothing = \bot^i$ when $h < \ell$.

Interval abstract domain (cont'd)



Interval abstraction

- The interval abstraction is

$$\langle \wp(\mathbb{Z}),\ \subseteq\rangle \xrightleftharpoons[\alpha_i]{\gamma_i} \langle \mathbb{P}^i,\ \sqsubseteq^i\rangle$$

  such that

$$\begin{array}{rcl}\alpha_i(S) & \triangleq & [\min S, \max S]\\ \gamma_i([\ell, h]) & \triangleq & \{z \in \mathbb{Z} \mid \ell - 1 < z < h + 1\}\end{array}$$

  where $-\infty - 1 = -\infty$ and $\infty + 1 = \infty$.

- The abstraction is extended pointwise to $\langle \mathbb{V} \to \wp(\mathbb{Z}),\ \dot{\subseteq}\rangle \xrightleftharpoons[\dot{\alpha}_i]{\dot{\gamma}_i} \langle \mathbb{V} \to \mathbb{P}^i,\ \dot{\sqsubseteq}^i\rangle$ for an assertional reachability analysis.
- (The cartesian abstraction is not expressive enough to be useful for a relational reachability analysis).

Interval cartesian reachability abstract interpreter

- Besides the complete lattice operations $\langle \overline{\mathbb{P}}^{i}, \sqsubseteq^{i}, \bot^{i}, \top^{i}, \sqcup^{i}, \sqcap^{i} \rangle$, the basic forward abstract operations are
  - $1^{i} \quad \triangleq \quad [1, 1]$,
  - $\ominus^{i} \langle [\ell_1, h_1], [\ell_2, h_2] \rangle \quad \triangleq \quad [\ell_1 - h_2, h_1 - \ell_2]$

  For example $\ominus^{i} \langle [0, 1], [-1, 0] \rangle = [0, 2]$ and $\ominus^{i} \langle [1, \infty], [-\infty, 0] \rangle = [1 - 0, \infty - (-\infty)]$ $= [1, \infty]$ since $\infty + \infty = \infty$;

Interval cartesian reachability abstract interpreter

- The basic backward abstract operations are
  - $\overset{i}{\ominus}_1 [a, b]\langle[\ell_1, h_1],\ [\ell_2, h_2]\rangle \ \triangleq\ \langle[\max(\ell_1, a + \ell_2), \min(h_1, b + h_2)],$ $[\max(\ell_2, \ell_1 - b), \min(h_2, h_1 - a)]\rangle$,
  - $\overset{i}{\lessdot}_1 \langle[\ell_1, h_1],\ [\ell_2, h_2]\rangle \ \triangleq\ \langle[\ell_1, \min(h_1, h_2 - 1)],\ [\max(\ell_2, \ell_1 + 1), h_2]\rangle$,
  - $\overline{\lessdot}^{i}_1 \langle[\ell_1, h_1],\ [\ell_2, h_2]\rangle \ \triangleq\ \langle[\max(\ell_1, \ell_2), h_1],\ [\ell_2, \min(h_1, h_2)]\rangle$.

  Examples:
  - $\overset{i}{\lessdot}_1 \langle[0, 0],\ [1, 1]\rangle = \langle[0, 0],\ [1, 1]\rangle$ since $a \in [0, 0]$ is always less than $b \in [1, 1]$.
  - $\overset{i}{\lessdot}_1 \langle[0, 0],\ [0, 0]\rangle = \langle\emptyset,\ \emptyset\rangle$ since $a \in [0, 0]$ is never less than $b \in [0, 0]$.
  - $\overset{i}{\lessdot}_1 \langle[1, 4],\ [0, 3]\rangle = \langle[1, 2],\ [2, 3]\rangle$ since $a \in [1, 4]$ must be less than $\max b \in [2, 3] = 3$ and $b \in [2, 3]$ must be greater than $\min a \in [1, 4] = 1$;

- Observe that the interval cartesian reachability semantics $\mathscr{A}^{i}[\![\mathtt{A}]\!]$ of an arithmetic expression $\mathtt{A}$ in (28.17) does not preverve lubs in general.
- For example
  - $\forall i \in \mathbb{Z} \mathrel{.} \mathscr{A}^{i}[\![\mathtt{x - x}]\!]\overline{\rho}[x \leftarrow [i, i]] = [0, 0]$
  - whereas for $\overline{\rho}[x \leftarrow [-\infty, \infty]] = \dot{\bigsqcup_{i \in \mathbb{Z}}}^{i} \overline{\rho}[x \leftarrow [i, i]]$,
  - $\mathscr{A}^{i}[\![\mathtt{x - x}]\!]\overline{\rho}[x \leftarrow [-\infty, \infty]] = [-\infty, \infty]$.

Static versus dynamic interval arithmetics

- dynamic interval arithmetics of Chapter **32** put bounds on rounding errors in floating point arithmetic at runtime (*i.e.* during one program computation that tests may be split in several traces considered one at a time),

- static interval analysis of this Chapter **33** is before runtime (and must deal with all possible program computations in finite time) [P. Cousot and R. Cousot, 1976].

Divergence

- Consider the simple diverging program

$$\mathtt{P}_1 \quad = \quad \begin{array}{l}\mathtt{while}\ \ell_1\ (\mathsf{tt}) \\ \quad \ell_2\ \mathtt{x = x+1\ ;} \\ \ell_3 \end{array}$$

- By (21.11), the interval static analysis from an initial assignment $\overline{\rho}_0$ of intervals to variables is

$$\widehat{\mathcal{S}}^{\,i}[\![\mathtt{P}_1]\!]\,\overline{\rho}_0 \quad = \quad \mathsf{lfp}^{\sqsubseteq^i}(\mathcal{F}^{\,i}[\![\mathtt{while}\ \ell_1\ (\mathsf{tt})\ \ell_2\ \mathtt{x = x + 1\ ;}]\!]\,\overline{\rho}_0)$$

Divergence (cont'd)

- The interval transformer is

$$
\begin{array}{rl}
& \mathscr{F}^i[\![\texttt{while}\ \ell_1\ (\texttt{tt})\ \ell_2\ \texttt{x = x + 1 ;}]\!]\ \overline{\rho}_0\ X\ \ell' \\
= & (\!|\ \ell' = \ell_1\ ?\ \overline{\rho}_0\ ⨾\ \texttt{x} \in \mathbb{V} \mapsto \bot^i\ |\!)\ \sqcup^i \\
& (\!|\ \ell' \in \{\ell_1, \ell_2\}\ ?\ \widehat{\mathcal{S}}^i[\![\ell_2\ \texttt{x = x + 1 ;}]\!]\ (\texttt{test}^i[\![\texttt{tt}]\!]X(\ell_1))\ \ell'\ ⨾\ \texttt{x} \in \mathbb{V} \mapsto \bot^i\ |\!)\ \sqcup^i \\
& (\!|\ \ell' = \ell_3\ ?\ \overline{\texttt{test}}^i[\![\texttt{tt}]\!]X(\ell_1)\ ⨾\ \texttt{x} \in \mathbb{V} \mapsto \bot^i\ |\!) \\
= & (\!|\ \ell' = \ell_1\ ?\ \overline{\rho}_0\ ⨾\ \texttt{x} \in \mathbb{V} \mapsto \bot^i\ |\!)\ \sqcup^i \\
& (\!|\ \ell' \in \{\ell_1, \ell_2\}\ ?\ \widehat{\mathcal{S}}^i[\![\ell_2\ \texttt{x = x + 1 ;}]\!]\ X(\ell_1)\ \ell'\ ⨾\ \texttt{x} \in \mathbb{V} \mapsto \bot^i\ |\!)\ \sqcup^i \\
& \texttt{x} \in \mathbb{V} \mapsto \bot^i \\
= & (\!|\ \ell' = \ell_1\ ?\ \overline{\rho}_0 \sqcup^i X(\ell_1)[x \leftarrow X(\ell_1)(\texttt{x}) \oplus^i [1, 1]]) \\
& \ \|\ \ell' = \ell_2\ ?\ X(\ell_1) \\
& \ ⨾\ /\star\ \ \ell' = \ell_3\ \ \star/\ \ \texttt{x} \in \mathbb{V} \mapsto \bot^i\ |\!)
\end{array}
$$

by $\texttt{test}^i[\![\texttt{tt}]\!]X(\ell_1) = X(\ell_1)$, and $\overline{\texttt{test}}^i[\![\texttt{tt}]\!]X(\ell_1) = \texttt{x} \in \mathbb{V} \mapsto \bot^i$ which is the identity element (or neutral element) of $\sqcup^i$.

Divergence (cont'd)

- The assignment is

$$\widehat{\mathcal{S}}^{\,i}[\![\ell_2\ \mathtt{x = x + 1\ ;}]\!]\, X(\ell_1)\,\ell \quad = \quad (\!|\ \ell = \ell_2\ ?\ X(\ell_1) \ \|\ \ell = \ell_1\ ?\ X(\ell_1)[x \leftarrow X(\ell_1)(\mathtt{x}) \oplus^i [1,1]]) \ :\ \mathtt{x} \in \mathbb{V} \mapsto \varnothing\ |\!)$$

in (21.7) where,

- $\mathcal{A}^{\,i}[\![\mathtt{x + 1}]\!]X(\ell_1) = X(\ell_1)(\mathtt{x}) \oplus^i [1,1]$,
- $[\ell_1, h_1]) \oplus^i [\ell_2, h_2] = [\ell_1, h_1]) \ominus^i ([0,0] \ominus^i [\ell_2, h_2]) = [\ell_1 + \ell_2, h_1 + h_2]$.

Divergence (cont'd)

- Assume that initially $\overline{\rho}_0(\mathsf{x}) = [0,0]$ and let $x = X(\ell_1)(\mathsf{x})$.
- The fixpoint computation amounts to solving the fixpoint equation

$$x \quad = \quad \mathscr{F}^{\,i}(x) \quad \text{where} \quad \mathscr{F}^{\,i}(x) \quad = \quad [0,0] \sqcup^{i} (x \oplus^{i} [1,1])$$

  iteratively (Theorem 15.21).

Iterations of $x = \mathscr{F}^{i}(x)$ where $\mathscr{F}^{i}(x) = [0,0] \sqcup^{i} (x \oplus^{i} [1,1])$

$$
\begin{array}{rcl}
x^0 & = & \perp^i \\
x^1 & = & \mathscr{F}^{i}(x^0) = [0,0] \sqcup^{i} (x^0 \oplus^{i} [1,1]) = [0,0] \\
x^2 & = & \mathscr{F}^{i}(x^1) = [0,0] \sqcup^{i} (x^1 \oplus^{i} [1,1]) = [0,0] \sqcup^{i} [1,1] = [0,1] \\
x^3 & = & \mathscr{F}^{i}(x^2) = [0,0] \sqcup^{i} (x^2 \oplus^{i} [1,1]) = [0,0] \sqcup^{i} [1,2] = [0,2] \\
\ldots & & \\
x^n & = & [0, n-1] \qquad \text{induction hypothesis} \\
x^{n+1} & = & \mathscr{F}^{i}(x^n) = [0,0] \sqcup^{i} (x^n \oplus^{i} [1,1]) \\
 & & \phantom{\mathscr{F}^{i}(x^n)} = [0,0] \sqcup^{i} [1,n] = [0,(n+1)-1] \\
\ldots & & \\
x^\omega & = & \bigsqcup^{i}_{n<\omega} [0, n-1] = [0, \infty] \qquad \text{limit} \\
x^{\omega+1} & = & \mathscr{F}^{i}(x^\omega) = [0,0] \sqcup^{i} (x^\omega \oplus^{i} [1,1]) = [0,0] \sqcup^{i} [1, \infty + 1] = [0, \infty] \\
 & = & \mathsf{lfp}^{\sqsubseteq^i}\, x \mapsto [0,0] \sqcup^{i} (x \oplus^{i} [1,1])
\end{array}
$$

Divergence (cont'd)

Unfortunalely computerized methods to infer induction hypotheses, to simplify the iteration terms, and to pass to the limit are far from effective. The most popular cases are (a mixture of)

- Consider finitary abstractions only (which limitations are studied in Section **34.11**);
- Ask for human help to get an inductive solution of the fixpoint equation (as considered in the verification methods of Chapter **26**), which can be difficult and costly *e.g*. when the inductive invariant is larger than the program and hard to change when the program is modified;
- To soundly automatize the induction and passage to the limit at the price of a loss of precision to enforce rapid convergence.
- This is the purpose of widenings, narrowings, and their duals.

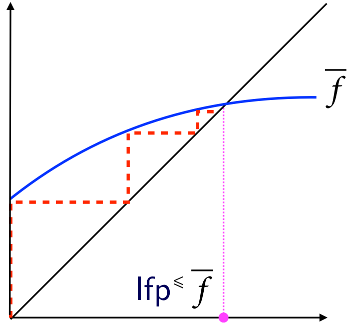
Extrapolation by widening

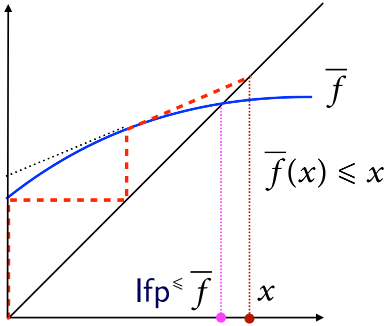
Widening

- The idea of the widening is to extrapolate from an iterate x^n and the next one x^{n+1} to an upper bound $x^n \triangledown x^{n+1}$ so as accelerate or enforce the convergence of the iterates in finitely many steps.
- The price to be paid is a loss of precision
- A sound although imprecise answer being considered better than no answer at all by memory and time resources exhaustion.

Example widening

- In the example below, the derivative is used to accelerate convergence by extrapolation (as in Newton–Raphson iteration method)





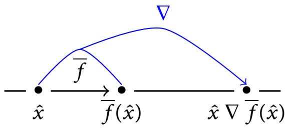
$\overline{f}$
$\overline{f}(x) \leqslant x$
$\mathsf{lfp}^{\leqslant}\,\overline{f}$
x

Interval widening

$$\begin{aligned} \bot^i \nabla^i x &\triangleq x \nabla^i \bot^i \triangleq x \\ [\ell_1, h_1] \nabla^i [\ell_2, h_2] &\triangleq [(\!|\, \ell_2 < \ell_1 \;?\; -\infty : \ell_1 \,|\!), (\!|\, h_2 > h_1 \;?\; \infty : h_1 \,|\!)] \end{aligned} \tag{33.5}$$

that essentially pushes unstable bounds to infinity.

This is an extrapolation



Upward iteration with widening for $x = \mathscr{F}^{i}(x)$ where $\mathscr{F}^{i}(x) \;=\; [0,0] \sqcup^{i} (x \oplus^{i} [1,1])$

$$
\begin{array}{rcl}
\hat{x}^0 &=& \bot^{i} \\
\hat{x}^1 &=& \hat{x}^0 \,\nabla^{i}\, \mathscr{F}^{i}(\hat{x}^0) \quad=\quad \hat{x}^0 \,\nabla^{i}\, ([0,0] \sqcup^{i} (\hat{x}^0 \oplus^{i} [1,1])) \quad=\quad \bot^{i} \,\nabla^{i}\, [0,0] \\
&=& [0,0] \qquad\qquad \text{since } \mathscr{F}^{i}(\hat{x}^0) = [0,0] \not\sqsubseteq^{i} \bot^{i} = \hat{x}^0 \\
\hat{x}^2 &=& \hat{x}^1 \,\nabla^{i}\, \mathscr{F}^{i}(\hat{x}^1) \quad=\quad \hat{x}^1 \,\nabla^{i}\, ([0,0] \sqcup^{i} (\hat{x}^1 \oplus^{i} [1,1])) \\
&=& [0,0] \,\nabla^{i}\, ([0,0] \sqcup^{i} [1,1]) \quad=\quad [0,0] \,\nabla^{i}\, [0,1] \quad=\quad [0,\infty] \\
&& \qquad\qquad \text{since } \mathscr{F}^{i}(\hat{x}^1) = [0,1] \not\sqsubseteq^{i} \hat{x}^1 = [0,0] \\
\hat{x}^n &=& \hat{x}^2, \qquad n \geqslant 2 \\
&& \text{since } \mathscr{F}^{i}(\hat{x}^2) = ([0,0] \sqcup^{i} (\hat{x}^2 \oplus^{i} [1,1])) \quad=\quad ([0,0] \sqcup^{i} ([0,\infty] \oplus^{i} [1,1])) \\
&& = ([0,0] \sqcup^{i} [1,\infty+1]) \quad=\quad [0,\infty] \sqsubseteq^{i} \hat{x}^2 = [0,\infty]
\end{array}
$$

Loss of precision

- In general, the result with widening is less precise than the limit of the infinite iterations.
- An example is

```
P₂  =   while ℓ₁ (x<1001)
          ℓ₂ x = x+1 ;
        ℓ₃
```

- Assuming initially $\overline{\rho}_0(\mathtt{x}) = [0,0]$, $x = X(\ell_1)(\mathtt{x})$, and $y = X(\ell_3)(\mathtt{x})$, the fixpoint computation amounts to solving the fixpoint system of equations

$$\begin{cases} x = \mathscr{F}^i(x) \\ y = x \sqcap^i [1001, \infty] \end{cases} \quad \text{where} \quad \mathscr{F}^i(x) = [0,0] \sqcup^i ((x \sqcap^i [-\infty, 1000]) \oplus^i [1,1])$$

- the least solution is $x = [0, 1001]$ and $y = [1001, 1001]$.

Upward iterates with widening

$$
\begin{array}{lcl}
\hat{x}^0 & = & \bot^i, \quad \hat{y}^0 = \bot^i \\
\hat{x}^1 & = & \hat{x}^0 \,\nabla^i\, \mathscr{F}^i(\hat{x}^0) \;=\; \hat{x}^0 \,\nabla^i\, ([0,0] \sqcup^i ((\hat{x}^0 \sqcap^i [-\infty,1000]) \oplus^i [1,1])) \\
 & = & \bot^i \,\nabla^i\, [0,0] \;=\; [0,0] \qquad \text{since } \mathscr{F}^i(\hat{x}^0) = [0,0] \not\sqsubseteq^i \bot^i = \hat{x}^0 \\
\hat{x}^2 & = & \hat{x}^1 \,\nabla^i\, \mathscr{F}^i(\hat{x}^1) \;=\; \hat{x}^1 \,\nabla^i\, ([0,0] \sqcup^i ((\hat{x}^1 \sqcap^i [-\infty,1000]) \oplus^i [1,1])) \\
 & = & [0,0] \,\nabla^i\, ([0,0] \sqcup^i [1,1]) \;=\; [0,0] \,\nabla^i\, [0,1] \\
 & = & [0,\infty] \qquad \text{since } \mathscr{F}^i(\hat{x}^1) = [0,1] \not\sqsubseteq^i \hat{x}^1 = [0,0] \\
\hat{x}^n & = & \hat{x}^2, \qquad n \geqslant 2
\end{array}
$$

$$
\begin{array}{rcl}
\text{since } \mathscr{F}^i(\hat{x}^2) & = & ([0,0] \sqcup^i ((\hat{x}^2 \sqcap^i [-\infty,1000]) \oplus^i [1,1])) \\
 & = & ([0,0] \sqcup^i (([0,\infty] \sqcap^i [-\infty,1000]) \oplus^i [1,1])) \\
 & = & ([0,0] \sqcup^i [1,1001]) \;=\; [0,1001] \;\sqsubseteq^i\; \hat{x}^2 \;=\; [0,\infty]
\end{array}
$$

$$
\hat{y} = \hat{x}^2 \sqcap^i [1001,\infty] = [0,\infty] \sqcap^i [1001,\infty] = [1001,\infty]
$$

Improving the solution

- The solution found is therefore $\hat{x} = [0, \infty]$ and $\hat{y} = [1001, \infty]$
- This is frustrating since $\mathscr{F}^i(\hat{x}) = [0, 1001]$ provides a better solution.
- This is because the upward iteration with widening was stopped when $\mathscr{F}^i(\hat{x}) \sqsubseteq^i \hat{x}$
- $\mathscr{F}^i$ is increasing so the downward iterates $\check{x} = \hat{x} \sqsupseteq^i \mathscr{F}^i(\check{x}) \sqsupseteq^i \mathscr{F}^i(\mathscr{F}^i(\check{x})) \sqsupseteq^i \ldots$ are all improving the precision of the solution until reaching a fixpoint.

Interpolation by narrowing

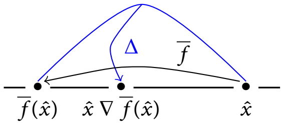
Interval narrowing

- For intervals the decreasing iterates can be very long and, in general, infinite.
- So we can stop at any point in the downwards iterates or use a narrowing to enforce convergence on semantic grounds.
- An example of interval narrowing is

$$\begin{aligned} \bot^i \,\Delta^i\, x &\triangleq x \,\Delta^i\, \bot^i \triangleq \bot^i \\ [\ell_1, h_1] \,\Delta^i\, [\ell_2, h_2] &\triangleq [\llparenthesis\, \ell_1 = -\infty \mathrel{?} \ell_2 \mathrel{:} \ell_1 \,\rrparenthesis, \llparenthesis\, h_1 = \infty \mathrel{?} h_2 \mathrel{:} h_1 \,\rrparenthesis] \end{aligned} \tag{33.7}$$

which attempts to improve infinite bounds only.

- This is an interpolation



Downward iterates with narrowing

$$\begin{array}{lll}
\check{x}^0 & = & \hat{x} = [0,\infty], \quad \check{y} = \hat{y} = [1001,\infty] \\
\check{x}^1 & = & \check{x}^0 \,\Delta^i\, \mathscr{F}^i(\check{x}^0) = \check{x}^0 \,\Delta^i\, ([0,0] \sqcup^i ((\check{x}^0 \sqcap^i [-\infty,1000]) \oplus^i [1,1])) \\
 & = & [0,\infty] \,\Delta^i\, [0,1001] = [0,1001] \\
 & & \quad \text{since } \mathscr{F}^i(\check{x}^0) = [0,1001] \neq [0,\infty] = \check{x}^0 \\
\check{x}^n & = & \check{x}^1, \qquad n \geqslant 1 \\
 & & \quad \text{since } \mathscr{F}^i(\check{x}^1) = ([0,0] \sqcup^i ((\check{x}^2 \sqcap^i [-\infty,1000]) \oplus^i [1,1])) \\
 & & \quad = ([0,0] \sqcup^i (([0,1001] \sqcap^i [-\infty,1000]) \oplus^i [1,1])) \\
 & & \quad = ([0,0] \sqcup^i [1,1001]) = [0,1001] = \check{x}^1 \\
\check{y} & = & \check{x}^1 \sqcap^i [1001,\infty] = [0,1001] \sqcap^i [1001,\infty] = [1001,1001].
\end{array}$$

- Of course no narrowing can in general recover the information lost by the widening.
- An example is

```
P₃  =   ℓ₁ x = 0 ;
        while ℓ₂ (x!=1001)
          ℓ₃ x = x+2 ;
        ℓ₄
```

- The widening can always be improved, see Section **34.5**.

- The widening cannot recover more information than that lost by the interval abstraction of Section **33.2**.
- For example, the interval analysis of the program

```
P₄  =   while ℓ₁ (x<1001)
          ℓ₂ x = x+2 ;
        ℓ₃
```

  will yield $y = [1001, 1002]$ since parity is lost by intervals (but can be recovered by a reduced product with the parity analysis or a arithmetic congruence analysis.
- The abstract domain of intervals can also be refined by anti-intervals (to close the domain by complement with $[-\infty, \infty]$, by lists of intervals such as $\langle[-\infty, 1000], [1002, \infty]\rangle$, by symbolic bounds, and more generally by relational domains.

Interval test reduction

- The interval abstract domain has no infinite strictly decreasing chains but they can be very long.
- So test reduction of Section **29.2** can be time-consuming without narrowing.
- For example the test `(x<y && y<x)` will reduce the bounds of `x` and `y` by 1 at each iteration until reaching $\bot^i$.

Assume variables are initialized to $\top^i = [\texttt{min\_int}, \texttt{max\_int}]$, the analysis of the program `if ((x<y nand y<x) nand (x<y nand y<x)) r=1; else r=3;` has the following test reduction iteration.

```
 [r:T; x:T; y:T]
[r:T; x:[-4611686018427387903, 4611686018427387902]; y:[-4611686018427387903, 4611686018427387902]]
[r:T; x:[-4611686018427387902, 4611686018427387901]; y:[-4611686018427387902, 4611686018427387901]]
[r:T; x:[-4611686018427387901, 4611686018427387900]; y:[-4611686018427387901, 4611686018427387900]]
[r:T; x:[-4611686018427387900, 4611686018427387899]; y:[-4611686018427387900, 4611686018427387899]]
...
[r:T; x:[-3, 3]; y:[-3, 3]]
[r:T; x:[-2, 2]; y:[-2, 2]]
[r:T; x:[-1, 1]; y:[-1, 1]]
[r:T; x:[0, 0]; y:[0, 0]]
[r:\_|\_; x:\_|\_; y:\_|\_]
```

so the test is always false.

```
 if l1: (((x < y) nand (y < x)) nand ((x < y) nand (y < x))) [r:T; x:T; y:T]
   l2: [r:_|_; x:_|_; y:_|_] r = 1;
 else
   l3: [r:T; x:T; y:T] r = 3;
l4: [r:[3, 3]; x:T; y:T]
```

With a narrowing, the iterations are much faster, but the result less precise.

```
      r:T; x:T; y:T]
     [r:T; x:[-4611686018427387903, 4611686018427387902]; y:[-4611686018427387903, 4611686018427387902]]
```

but the test is not found to be always false so the analysis is less precise.

```
      if l1: (((x < y) nand (y < x)) nand ((x < y) nand (y < x))) [r:T; x:T; y:T]
        l2: [r:T; x:[-4611686018427387903, 4611686018427387902]; y:[-4611686018427387903,
                                                                         4611686018427387902]] r = 1;
      else
        l3: [r:T; x:T; y:T] r = 3;
     l4: [r:[1, 3]; x:T; y:T]
```

Bibliography I

Cousot, Patrick and Radhia Cousot (1976). “Static determination of dynamic properties of programs”. In: *Proceedings of the Second International Symposium on Programming*. Dunod, Paris, France, pp. 106–130.

Home work

Read Ch. **33** “Static interval analysis” of

Principles of Abstract Interpretation
Patrick Cousot
MIT Press

The End, Thank you