1 Task1: Designing Difference and Interval Analysis

1.1 Interval Analysis

The lattice for interval analysis is defined as

$$interval = lift([l, h] \mid l, h \in N \wedge l \leq h) \tag{1}$$

where $lift(L)$ operator adds a new bottom element $\bot$ to the complete lattice L (see Fig.1).
and

$$N = \{-\infty, ..., -2, -1, 0, 1, 2, ..., -\infty\} \tag{2}$$

the partial order for interval lattice are defined as

$$[l_1, h_1] \sqsubseteq [l_2, h_2] \Leftrightarrow l_2 \leq l_2 \wedge h_1 \leq h_2 \tag{3}$$

and

$$\top = [-\infty, \infty] \tag{4}$$

$$[l_1, h_1] \cup [l_2, h_2] = [min(l_1, h_1), max(l_2, h_2)] \tag{5}$$

$$[l_1, h_1] \cap [l_2, h_2] = \begin{cases} [max(l_1, l_2), min(h_1, h_2)] & if\ max(l_1, l_2) \leq min(h_1, h_2) \\ \bot & otherwise \end{cases} \tag{6}$$

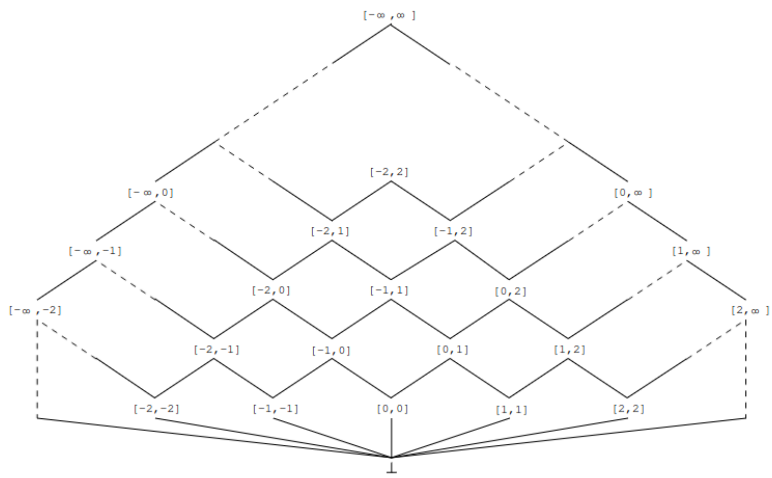

Figure 1: Interval lattice [1]

For Galois connections, concrete domain is

$$D = (2^{\mathbb{Z}}, \sqsubseteq) \tag{7}$$

abstract domain is

$$D^{\#} = (Int, \sqsubseteq) \tag{8}$$

where

$$Int = (\mathbb{Z} \cup \{-\infty\}) \times (\mathbb{Z} \cup \{\infty\}) \cup \{\varnothing\} \tag{9}$$

The lattice for the abstract domain is the one defined above. Let α be the abstraction function and γ be

[1] Møller, A. and Schwartzbach, M.I., 2012. Static program analysis. Notes. Feb.

the concretization function, where

$$\alpha : 2^{\mathbb{Z}} \to Int \tag{10}$$

$$\alpha(Z) = \begin{cases} \varnothing & if\ Z = \varnothing \\ [\sqcap Z, \sqcup Z] & otherwise \end{cases} \tag{11}$$

$$\gamma : Int \to 2^{\mathbb{Z}} \tag{12}$$

$$\gamma(I) = \begin{cases} \varnothing & if\ I = \varnothing \\ \{z \in \mathbb{Z} \mid z_1 \le z \le z_2\} & if\ I = [z_1, z_2] \end{cases} \tag{13}$$

The pair of α and γ defines the Galois connections.

In concrete domain, define $\mathcal{C}[\![c]\!]$ to be the transfer function for command c . Let $\mathcal{X}_l$ be the states of the program at the program point $l \in L$, specifically, $\mathcal{X}_e$ are the states of the program at entry. Then

$$\mathcal{X}_{l \neq e} = \bigcup_{l'} \mathcal{C}[\![c]\!]\mathcal{X}_{l'} \tag{14}$$

where l' is a program point that can directly lead to l.

Furthermore, let the superscript # represent the entity's counterpart in abstract domain, then the abstract equation system can be defined as

$$\mathcal{X} : L \to D^{\#} any\ solution\ of \begin{cases} \mathcal{X}_e^{\#} such\ that\ \mathcal{X}_e \subseteq \gamma(\mathcal{X}_e^{\#}) \\ \mathcal{X}_{l \neq e}^{\#} \supseteq^{\#} \bigcup_{l'}^{\#} \mathcal{C}^{\#}[\![c]\!]\mathcal{X}_{l'}^{\#} \end{cases} \tag{15}$$

For interval analysis, the abstract arithmetic operators can be defined as below. Let $f^{\#} : D^{\#n} \to D^{\#}$ be a safe approximation in abstract domain of function $f : D^n \to D$ in concrete domain with rank n, then

$$z^{\#} = [z, z] \tag{16}$$

$$-^{\#}[l, h] = [-h, -l] \tag{17}$$

$$[l_1, h_1] +^{\#} [l_2, h_2] = [l_1 + l_2, h_1 + h_2] \tag{18}$$

$$[l_1, h_1] -^{\#} [l_2, h_2] = [l_1 - h_2, h_1 - l_2] \tag{19}$$

$$[l_1, h_1] \times^{\#} [l_2, h_2] = [min(l_1 l_2, l_1 h_2, h_1 l_2, h_2 h_2), max(l_1 l_2, l_1 h_2, h_1 l_2, h_2 h_2)] \tag{20}$$

$$[l_1, h_1]/^{\#}[l_2, h_2] = \begin{cases} \perp & if\ l_2 = h_2 = 0 \\ [min(l_1/l_2, l_1/h_2, h_1/l_1, h_1/h_2), & if\ 0 \le l_2 \\ max(l_1/l_2, l_1/h_2, h_1/l_1, h_1/h_2)] & \\ [-h_1, -l_1]/^{\#}[-h_2, -l_2] & if\ h_2 \le 0 \\ ([l_1, h_1]/^{\#}[l_2, 0]) \cup^{\#} ([l_1, h_1]/^{\#}[0, h_2]) & otherwise \end{cases} \tag{21}$$

where $\pm\infty \times 0 = 0$, $\forall x : x/\pm\infty = 0$, $\forall x : x/0 = \perp$, $\infty - \infty = 0$, $\infty + \infty = \infty$, $\forall x < 0 : x \times \infty = -\infty$

For modulo, let $m = max(|l_2| - 1, |h_2| - 1)$ then

$$[l_1, h_1]\%^{\#}[l_2, h_2] = \begin{cases} [0, min(h_1, m)] & if\ l_1 \ge 0 \\ [-min(-l_1, m), 0] & if\ h_1 \le 0 \\ [-min(-l_1, m), min(h_1, m)] & if\ l_1 < 0 \wedge h_1 > 0 \end{cases} \tag{22}$$

In addition, any operation with $\perp$ yields $\perp$.

For boolean operation

$$[l_1, h_1] <^{\#} [l_2, h_2] = \begin{cases} true & if\ l_1 < h_2 \\ false & otherwise \end{cases} \tag{23}$$

$$[l_1, h_1] \leq^{\#} [l_2, h_2] = \begin{cases} true & if\ l_1 \leq h_2 \\ false & otherwise \end{cases} \tag{24}$$

$$[l_1, h_1] >^{\#} [l_2, h_2] = \begin{cases} true & if\ h_1 > l_2 \\ false & otherwise \end{cases} \tag{25}$$

$$[l_1, h_1] \geq^{\#} [l_2, h_2] = \begin{cases} true & if\ h_1 \geq l_2 \\ false & otherwise \end{cases} \tag{26}$$

$$[l_1, h_1] ==^{\#} [l_2, h_2] = \begin{cases} false & if\ l_1 > h_2 \vee l_2 > h_1 \\ true & otherwise \end{cases} \tag{27}$$

$$[l_1, h_1] \neq^{\#} [l_2, h_2] = \begin{cases} false & if\ l_1 == h_1 \wedge l_2 == h_2 \wedge l_1 == l_2 \\ true & otherwise \end{cases} \tag{28}$$

Interestingly, in the above definitions, $[l_1, h_1] > [l_2, h_2]$ and $[l_1, h_2] < [l_2, h_2]$ can be true at the same time as long as there is an overlap between them.

For path sensitivity analysis, we restrict our analysis to $X \blacktriangle Y$ and $X \blacktriangle c$ where $\blacktriangle$ is one of $<$, $\leq$, $>$, $\geq$, $==$ or $\neq$. Their corresponding transfer function $\mathcal{C}^{\#}$ are defined below. (Generic backward arithmetic and comparison operators which refine their argument are more complex to implement.)

Assume $X = [l_1, h_1]$ and $Y = [l_2, h_2]$. For brevity, we only stated the transfer function for $X \blacktriangle Y$. That for $X \blacktriangle c$ is similar without the state of c being updated as it is a constant.

$$\mathcal{C}^{\#}[\![X > Y]\!]\mathcal{X}^{\#} = \begin{cases} \perp^{\#} & if\ l_2 \geq h_1 \\ \mathcal{X}^{\#}[X \to [max(l_1, l_2 + 1), h_1], Y \to [l_1, min(h_1 - 1, h_2)]] & otherwise \end{cases} \tag{29}$$

$$\mathcal{C}^{\#}[\![X \geq Y]\!]\mathcal{X}^{\#} = \begin{cases} \perp^{\#} & if\ l_2 > h_1 \\ \mathcal{X}^{\#}[X \to [max(l_1, l_2), h_1], Y \to [l_1, min(h_1, h_2)]] & otherwise \end{cases} \tag{30}$$

$$\mathcal{C}^{\#}[\![X < Y]\!]\mathcal{X}^{\#} = \begin{cases} \perp^{\#} & if\ l_1 \geq h_2 \\ \mathcal{X}^{\#}[X \to [l_1, min(h_1, h_2 - 1)], Y \to [max(l_1 + 1, l_2), h_2]] & otherwise \end{cases} \tag{31}$$

$$\mathcal{C}^{\#}[\![X \leq Y]\!]\mathcal{X}^{\#} = \begin{cases} \perp^{\#} & if\ l_1 > h_2 \\ \mathcal{X}^{\#}[X \to [l_1, min(h_1, h_2)], Y \to [max(l_1, l_2), h_2]] & otherwise \end{cases} \tag{32}$$

$$\mathcal{C}^{\#}[\![X == Y]\!]\mathcal{X}^{\#} = \begin{cases} \perp^{\#} & if\ l_1 > h_2 \vee l_2 > h_1 \\ \mathcal{X}^{\#}[X \to [max(l_1, l_2), min(h_1, h_2)], & otherwise \\ Y \to [max(l_1, l_2), min(h_1, h_2)]] & \end{cases} \tag{33}$$

$$\mathcal{C}^{\#}[\![X \neq Y]\!]\mathcal{X}^{\#} = \begin{cases} \perp^{\#} & if\; l_1 = h_1 \wedge l_2 = h_2 \wedge l_1 = l_2 \\ \mathcal{X}^{\#} & otherwise \end{cases} \tag{34}$$

1.2 Difference Analysis

In this task, we use interval as an abstraction. So the lattice, Galois connection and abstract semantics are the same as the for interval analysis above.

At each program point, the max separation between two variables can be computed as $max(|l_1 - h_2|, |l_2 - h_1|)$