Static Program Analysis

Lecture 8: Dataflow Analysis VII (Interval Analysis & Widening)

Thomas Noll

Lehrstuhl für Informatik 2
(Software Modeling and Verification)

RWTH Aachen University

noll@cs.rwth-aachen.de

http://www-i2.informatik.rwth-aachen.de/i2/spa11/

Summer Semester 2011

Outline

Dataflow Analysis with Non-ACC Domains

- **Reminder:** $(D, \sqsubseteq)$ satisfies ACC if each ascending chain $d_1 \sqsubseteq d_2 \sqsubseteq \ldots$ eventually stabilizes, i.e., there exists $n \in \mathbb{N}$ such that $d_n = d_{n+1} = \ldots$
- If height (= maximal chain length) of $(D, \sqsubseteq)$ is m, then fixpoint computation terminates after $\leq |L| \cdot m$ iterations
- **But:** if $(D, \sqsubseteq)$ has infinite ascending chains
  $\Longrightarrow$ algorithm may not terminate
- **Solution:** use widening operators to enforce termination

Widening Operators

Definition (Widening operator)

Let $(D, \sqsubseteq)$ be a complete lattice. A mapping $\nabla : D \times D \to D$ is called widening operator if

- for every $d_1, d_2 \in D$,
$$d_1 \sqcup d_2 \sqsubseteq d_1 \nabla d_2$$
and
- for all ascending chains $d_0 \sqsubseteq d_1 \sqsubseteq \ldots$, the ascending chain $d_0^\nabla \sqsubseteq d_1^\nabla \sqsubseteq \ldots$ eventually stabilizes where
$$d_0^\nabla := d_0 \text{ and } d_{i+1}^\nabla := d_i^\nabla \nabla d_{i+1} \text{ for } i \in \mathbb{N}$$

Remarks:

- $(d_i^\nabla)_{i \in \mathbb{N}}$ is clearly an ascending chain as
$$d_{i+1}^\nabla = d_i^\nabla \nabla d_{i+1} \sqsupseteq d_i^\nabla \sqcup d_{i+1} \sqsupseteq d_i^\nabla$$
- In contrast to $\sqcup$, ∇ does not have to be commutative, associative, monotonic, nor absorptive ($d \nabla d = d$)
- The requirement $d_1 \sqcup d_2 \sqsubseteq d_1 \nabla d_2$ guarantees soundness of widening

The Domain of Interval Analysis

- The domain $(Int, \subseteq)$ of intervals over $\mathbb{Z}$ is defined by

$$Int := \{[z_1, z_2] \mid z_1 \in \mathbb{Z} \cup \{-\infty\}, z_2 \in \mathbb{Z} \cup \{+\infty\}\}, z_1 \leq z_2\} \cup \{\emptyset\}$$

  where
  - $-\infty \leq z$, $z \leq +\infty$, and $-\infty \leq +\infty$ (for all $z \in \mathbb{Z}$)
  - $\emptyset \subseteq I$ (for all $I \in Int$)
  - $[y_1, y_2] \subseteq [z_1, z_2]$ iff $z_1 \leq y_1$ and $y_2 \leq z_2$
- $(Int, \subseteq)$ is a complete lattice with (for every $\mathcal{I} \subseteq Int$)

$$\bigsqcup \mathcal{I} = \begin{cases} \emptyset & \text{if } \mathcal{I} = \emptyset \text{ or } \mathcal{I} = \{\emptyset\} \\ [Z_1, Z_2] & \text{otherwise} \end{cases}$$

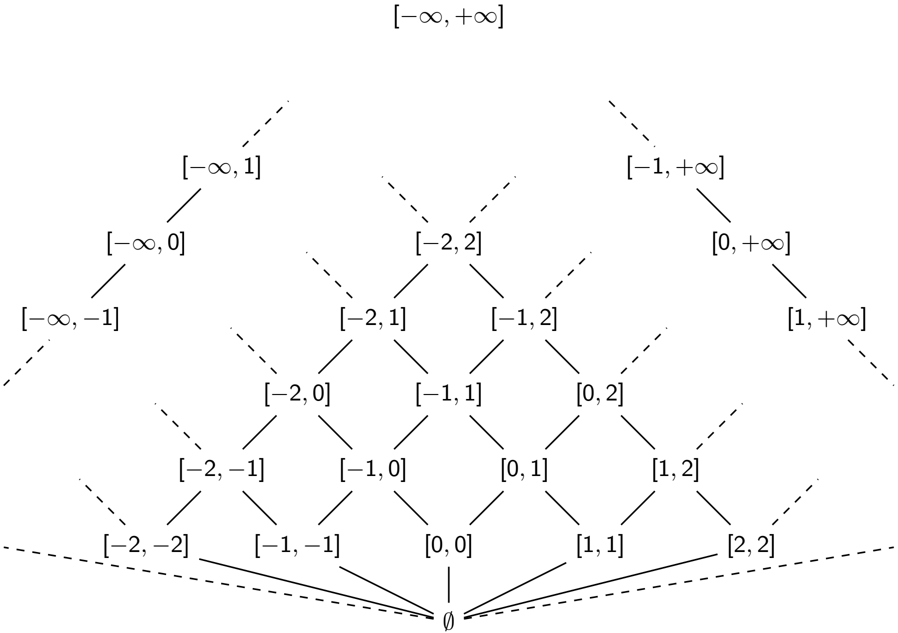
  where

$$Z_1 := \textstyle\bigsqcap_{\mathbb{Z} \cup \{-\infty\}} \{z_1 \mid [z_1, z_2] \in \mathcal{I}\}$$
$$Z_2 := \textstyle\bigsqcup_{\mathbb{Z} \cup \{+\infty\}} \{z_2 \mid [z_1, z_2] \in \mathcal{I}\}$$

  (and thus $\bot = \emptyset$, $\top = [-\infty, +\infty]$)
- Clearly $(Int, \subseteq)$ has infinite ascending chains, such as

$$\emptyset \subseteq [1, 1] \subseteq [1, 2] \subseteq [1, 3] \subseteq \ldots$$

The Complete Lattice of Interval Analysis



Outline

Formalizing Interval Analysis I

The dataflow system $S = (L, E, F, (D, \sqsubseteq), \iota, \varphi)$ is given by

- set of labels $L := L_c$,
- extremal labels $E := \{\mathrm{init}(c)\}$ (forward problem),
- flow relation $F := \mathrm{flow}(c)$ (forward problem),
- complete lattice $(D, \sqsubseteq)$ where
  - $D := \{\delta \mid \delta : Var_c \to Int\}$
  - $\delta_1 \sqsubseteq \delta_2$ iff $\delta_1(x) \subseteq \delta_2(x)$ for every $x \in Var_c$
- $\iota := \top_D : Var_c \to Int : x \mapsto \top_{Int} = [-\infty, +\infty]$
- φ: see next slide

Formalizing Interval Analysis II

Transfer functions $\{\varphi_l \mid l \in L\}$ are defined by

$$\varphi_l(\delta) := \begin{cases} \delta & \text{if } B^l = \texttt{skip} \text{ or } B^l \in BExp \\ \delta[x \mapsto val_\delta(a)] & \text{if } B^l = (x \texttt{ := } a) \end{cases}$$

where

$$val_\delta(x) := \delta(x) \qquad val_\delta(a_1\texttt{+}a_2) := val_\delta(a_1) \oplus val_\delta(a_2)$$
$$val_\delta(z) := [z, z] \qquad val_\delta(a_1\texttt{-}a_2) := val_\delta(a_1) \ominus val_\delta(a_2)$$
$$val_\delta(a_1\texttt{*}a_2) := val_\delta(a_1) \odot val_\delta(a_2)$$

with

$$\emptyset \oplus I := I \oplus \emptyset := \emptyset \ominus I := \ldots := \emptyset$$
$$[y_1, y_2] \oplus [z_1, z_2] := [y_1 + z_1, y_2 + z_2]$$
$$[y_1, y_2] \ominus [z_1, z_2] := [y_1 - z_2, y_2 - z_1]$$
$$[y_1, y_2] \odot [z_1, z_2] := [\min_{y \in [y_1,y_2], z \in [z_1,z_2]} y \cdot z, \max_{y \in [y_1,y_2], z \in [z_1,z_2]} y \cdot z]$$

Remarks:

- Possible refinement of DFA to take conditional blocks b^l into account
  - essentially: b as edge label, $\varphi_l(\delta)(x) = \delta(x) \setminus \{z \in \mathbb{Z} \mid x = z \implies \neg b\}$ (cf. "Conditions and Assertions" later)
- Important: soundness and optimality of abstract operations
  - soundness: $z_1 \in I_1, z_2 \in I_2 \implies z_1 + z_2 \in I_1 \oplus I_2$
  - optimality: $I_1 \oplus I_2$ as small as possible

Outline

Applying Widening to Interval Analysis

- A widening operator: $\nabla : Int \times Int \to Int$ with

$$\begin{aligned}
\emptyset \nabla I &:= I \nabla \emptyset := I \\
[x_1, x_2] \nabla [y_1, y_2] &:= [z_1, z_2] \quad \text{where} \\
z_1 &:= \begin{cases} x_1 & \text{if } x_1 \leq y_1 \\ -\infty & \text{otherwise} \end{cases} \\
z_2 &:= \begin{cases} x_2 & \text{if } y_2 \leq x_2 \\ +\infty & \text{otherwise} \end{cases}
\end{aligned}$$

- Widening turns infinite ascending chain

$$I_0 = \emptyset \subseteq I_1 = [1,1] \subseteq I_2 = [1,2] \subseteq I_3 = [1,3] \subseteq \ldots$$

into a finite one:

$$\begin{aligned}
I_0^\nabla &= I_0 = \emptyset \\
I_1^\nabla &= I_0^\nabla \nabla I_1 = \emptyset \nabla [1,1] = [1,1] \\
I_2^\nabla &= I_1^\nabla \nabla I_2 = [1,1] \nabla [1,2] = [1,+\infty] \\
I_3^\nabla &= I_2^\nabla \nabla I_3 = [1,+\infty] \nabla [1,3] = [1,+\infty]
\end{aligned}$$

- In fact, the maximal chain length arising with this operator is 4:

$$\emptyset \subseteq [3,7] \subseteq [3,+\infty] \subseteq [-\infty,+\infty]$$

Worklist Algorithm with Widening I

Goal: extend Algorithm 5.2 by widening to ensure termination

Algorithm 8.1 (Worklist algorithm with widening)

Input: *dataflow system* $S = (L, E, F, (D, \sqsubseteq), \iota, \varphi)$

Variables: $W \in (L \times L)^*$, $\{\mathsf{AI}_l \in D \mid l \in L\}$

Procedure:

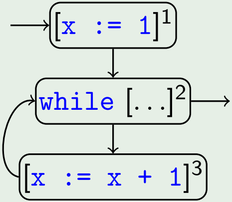
$W := \varepsilon$; **for** $(l, l') \in F$ **do** $W := W \cdot (l, l')$; *% Initialize* W
for $l \in L$ **do** *% Initialize* AI
  if $l \in E$ **then** $\mathsf{AI}_l := \iota$ **else** $\mathsf{AI}_l := \bot_D$;
while $W \neq \varepsilon$ **do**
  $(l, l') :=$ **head**(W); $W :=$ **tail**(W);
  if $\varphi_l(\mathsf{AI}_l) \not\sqsubseteq \mathsf{AI}_{l'}$ **then** *% Fixpoint not yet reached*
    $\mathsf{AI}_{l'} := \mathsf{AI}_{l'} \nabla \varphi_l(\mathsf{AI}_l)$;
    for $(l', l'') \in F$ **do**
      if (l', l'') *not in* W **then** $W := (l', l'') \cdot W$;

Output: $\{\mathsf{AI}_l \mid l \in L\}$, *denoted by* $\mathsf{fix}^\nabla(\Phi_S)$

Remark: due to widening, only $\mathsf{fix}(\Phi_S) \sqsubseteq \mathsf{fix}^\nabla(\Phi_S)$ is guaranteed (cf. Thm. 5.4)

Worklist Algorithm with Widening II

Example 8.2



Transfer functions (for $\delta(\texttt{x}) = I$):

$$\begin{aligned}
\varphi_1(I) &= [1,1] \\
\varphi_2(I) &= I \\
\varphi_3(\emptyset) &= \emptyset \\
\varphi_3([x_1,x_2]) &= [x_1+1, x_2+1]
\end{aligned}$$

Application of worklist algorithm (on the board)

1. without widening: does not terminate
2. with widening: terminates with expected result for AI_2 ($[1,+\infty]$)