Homework 2
AAA616, Fall 2022

Hakjoo Oh

Due: 11/30, 23:59

The goal of this assignment is to design and implement a static analyzer based on the abstract interpretation framework. Consider the following language:

$$\begin{array}{rcl} lv & \rightarrow & x \mid *x \\ e & \rightarrow & n \mid lv \mid \&lv \mid e_1 + e_2 \mid e_1 \star e_2 \mid e_1 - e_2 \\ b & \rightarrow & \texttt{true} \mid \texttt{false} \mid e_1 = e_2 \mid e_1 \leq e_2 \mid \neg b \mid b_1 \wedge b_2 \\ c & \rightarrow & lv := e \mid lv := \texttt{alloc} \mid \texttt{skip} \mid c_1; c_2 \mid \texttt{if}\ b\ c_1\ c_2 \mid \texttt{while}\ b\ c \end{array}$$

Assume programs are represented by control flow graphs. Let $(N, \rightarrow)$ be a control-flow graph and $\mathsf{cmd}(n)$ be the command associated with node n:

$$lv := e \mid lv := \texttt{alloc} \mid assume(b)$$

1 Concrete Semantics

The concrete domain and semantics are defined as follows (details to be explained in class).

Concrete Domain

$$\begin{array}{rcl} Mem & = & Loc \rightarrow Val \\ Loc & = & Var + HeapAddr \\ Val & = & Int + Loc \end{array}$$

Concrete Semantics

- $[\![lv]\!] : Mem \rightarrow Loc$:

$$\begin{array}{rcl} [\![x]\!](m) & = & x \\ [\![*x]\!](m) & = & m(x) \end{array}$$

- $[\![e]\!] : Mem \rightarrow Val$:

$$\begin{array}{rcl} [\![n]\!](m) & = & n \\ [\![lv]\!](m) & = & m([\![lv]\!](m)) \\ [\![\&lv]\!](m) & = & [\![lv]\!](m) \\ [\![e_1 + e_2]\!](m) & = & [\![e_1]\!](m) +_{Int} [\![e_2]\!](m) \end{array}$$

- $[\![b]\!] : Mem \rightarrow Bool$:

$$\begin{array}{rcl} [\![\texttt{true}]\!](m) & = & true \\ [\![\texttt{false}]\!](m) & = & false \\ [\![e_1 = e_2]\!](m) & = & [\![e_1]\!](m) =_{Int} [\![e_2]\!](m) \\ [\![e_1 \leq e_2]\!](m) & = & [\![e_1]\!](m) \leq_{Int} [\![e_2]\!](m) \\ [\![\neg b]\!](m) & = & \neg [\![b]\!](m) \\ [\![b_1 \wedge b_2]\!](m) & = & [\![b_1]\!](m) \wedge [\![b_2]\!](m) \end{array}$$

- $f_n : \wp(\mathit{Mem}) \to \wp(\mathit{Mem})$:

$$\begin{array}{rcll}
f_n(M) & = & \{m[[\![lv]\!](m) \mapsto [\![e]\!](m)] \mid m \in M\} & \cdots\ \mathsf{cmd}(n) = lv := e \\
f_n(M) & = & \{m[[\![lv]\!](m) \mapsto l, l \mapsto 0] \mid m \in M\} & \cdots\ \mathsf{cmd}(n) = lv := \mathtt{alloc}, \\
 & & & \quad l \text{ is new} \\
f_n(M) & = & \{m \in M \mid [\![b]\!](m) = \mathit{true}\} & \cdots\ \mathsf{cmd}(n) = \mathit{assume}(b)
\end{array}$$

- $F : (N \to \wp(\mathit{Mem})) \to (N \to \wp(\mathit{Mem}))$:

$$F(X) = \lambda n.\ f_n\Big(\bigcup_{n' \to n} X(n')\Big)$$

- Collecting semantics:

$$\mathit{fix} F \in N \to \wp(\mathit{Mem})$$

2 Abstract Semantics

The abstract domain and semantics are defined as follows (details to be explained in class).

Abstract Domain

$$\begin{array}{rcl}
\widehat{\mathit{Mem}} & = & \widehat{\mathit{Loc}} \to \widehat{\mathit{Val}} \\
\widehat{\mathit{Loc}} & = & \mathit{Var} + \mathit{AllocSite} \\
\widehat{\mathit{Val}} & = & \mathit{Interval} \times \wp(\widehat{\mathit{Loc}})
\end{array}$$

- $\wp(\mathit{HeapAddr}) \underset{\alpha_{\mathit{HeapAddr}}}{\overset{\gamma_{\mathit{HeapAddr}}}{\leftrightarrows}} \wp(\mathit{AllocSite})$

$$\alpha_{\mathit{HeapAddr}}(H) = \{\mathsf{allocsite}(h) \mid h \in H\}$$

- $\wp(\mathit{Loc}) \underset{\alpha_{\mathit{Loc}}}{\overset{\gamma_{\mathit{Loc}}}{\leftrightarrows}} \wp(\widehat{\mathit{Loc}})$

$$\alpha_{\mathit{Loc}}(L) = \{x \mid x \in L\} \uplus \alpha_{\mathit{HeapAddr}}(\{h \mid h \in L\})$$

- $\wp(\mathit{Int}) \underset{\alpha_{\mathit{Int}}}{\overset{\gamma_{\mathit{Int}}}{\leftrightarrows}} \mathit{Interval}$

$$\alpha_{\mathit{Int}}(\emptyset) = \bot, \quad \alpha_{\mathit{Int}}(Z) = [\mathsf{min}(Z), \mathsf{max}(Z)]$$

- $\wp(\mathit{Val}) \underset{\alpha_{\mathit{Val}}}{\overset{\gamma_{\mathit{Val}}}{\leftrightarrows}} \widehat{\mathit{Val}}$

$$\alpha_{\mathit{Val}}(V) = \langle \alpha_{\mathit{Int}}(\{z \mid z \in V\}), \alpha_{\mathit{Loc}}(\{l \mid l \in V\}) \rangle$$

- $\wp(\mathit{Mem}) \underset{\alpha_{\mathit{Mem}}}{\overset{\gamma_{\mathit{Mem}}}{\leftrightarrows}} \widehat{\mathit{Mem}}$

$$\alpha_{\mathit{Mem}}(M) = \lambda l. \begin{cases} \bigsqcup\{m(l) \mid m \in M\} & \cdots\ l \in \mathit{Var} \\ \bigsqcup\{m(a) \mid m \in M, a \in \gamma_{\mathit{HeapAddr}}(l)\} & \cdots\ l \in \mathit{AllocSite} \end{cases}$$

- $N \to \wp(\mathit{Mem}) \underset{\alpha}{\overset{\gamma}{\leftrightarrows}} N \to \widehat{\mathit{Mem}}$

$$\alpha(X) = \lambda n. \alpha_{\mathit{Mem}}(X(n))$$

Abstract Semantics

- $[\![lv]\!] : \widehat{Mem} \to \wp(Loc)$

$$\begin{array}{rcl} [\![x]\!](m) & = & \{x\} \\ [\![*x]\!](m) & = & m(x).2 \end{array}$$

- $[\![e]\!] : \widehat{Mem} \to Val$

$$\begin{array}{rcl} [\![n]\!](m) & = & \langle [n,n], \emptyset \rangle \\ [\![lv]\!](m) & = & \bigsqcup_{l \in [\![lv]\!](m)} m(l) \\ [\![\&lv]\!](m) & = & \langle \bot, [\![lv]\!](m) \rangle \\ [\![e_1 + e_2]\!](m) & = & [\![e_1]\!](m) \hat{+} [\![e_2]\!](m) \end{array}$$

- $[\![b]\!] : \widehat{Mem} \to Bool$

$$\begin{array}{rcl} [\![\texttt{true}]\!](m) & = & true \\ [\![\texttt{false}]\!](m) & = & false \\ [\![e_1 = e_2]\!](m) & = & [\![e_1]\!](m) \hat{=} [\![e_2]\!](m) \\ [\![e_1 \leq e_2]\!](m) & = & [\![e_1]\!](m) \hat{\leq} [\![e_2]\!](m) \\ [\![\neg b]\!](m) & = & \neg [\![b]\!](m) \\ [\![b_1 \wedge b_2]\!](m) & = & [\![b_1]\!](m) \wedge [\![b_2]\!](m) \end{array}$$

- $\hat{f}_n : \widehat{Mem} \to \widehat{Mem}$:

$$\begin{array}{rcll} \hat{f}_n(m) & = & m[x \mapsto [\![e]\!](m)] & \cdots\ \mathsf{cmd}(n) = x := e \\ \hat{f}_n(m) & = & m[x \mapsto [\![e]\!](m)] & \cdots\ \mathsf{cmd}(n) = lv := e, \\ & & & \quad [\![lv]\!](m) = \{x\} \\ \hat{f}_n(m) & = & \bigsqcup_{l \in [\![lv]\!](m)} m[l \mapsto m(l) \sqcup [\![e]\!](m)] & \cdots\ \mathsf{cmd}(n) = lv := e \\ \hat{f}_n(m) & = & m[x \mapsto (\bot, \{n\}), n \mapsto ([0,0], \emptyset)] & \cdots\ \mathsf{cmd}(n) = x := \texttt{alloc} \\ \hat{f}_n(m) & = & m[x \mapsto (\bot, \{n\}), n \mapsto ([0,0], \emptyset)] & \cdots\ \mathsf{cmd}(n) = lv := \texttt{alloc}, \\ & & & \quad [\![lv]\!](m) = \{x\} \\ \hat{f}_n(m) & = & \bigsqcup_{l \in [\![lv]\!](m)} m'[l \mapsto m(l) \sqcup (\bot, \{n\})] & \cdots\ \mathsf{cmd}(n) = lv := \texttt{alloc} \\ & & & \quad m' = m[n \mapsto ([0,0], \emptyset)] \\ \hat{f}_n(m) & = & \bigsqcup \{m' \sqsubseteq m \mid true \sqsubseteq [\![b]\!](m')\} & \cdots\ \mathsf{cmd}(n) = assume(b) \end{array}$$

- $\hat{F} : (N \to \widehat{Mem}) \to (N \to \widehat{Mem})$:

$$\hat{F}(X) = \lambda n.\ \hat{f}_n\Big(\bigsqcup_{n' \to n} X(n')\Big)$$

- Abstract semantics:

$$\bigsqcup_{i \geq 0} \hat{F}^i(\bot) \in N \to \widehat{Mem}$$

3 Problems

1. Prove that the static analysis designed above is sound: i.e.,

$$\alpha(\mathit{fix} F) \sqsubseteq \bigsqcup_{i \geq 0} \hat{F}^i(\bot).$$

   (To formally prove the soundness, you may need to define the abstraction and semantics more precisely — you are allowed to modify them.)

2. Implement the static analyzer in OCaml (or in your preferred language).