

AAA616: Program Analysis

Lecture 13 — Abstract Interpretation Example (2)

Hakjoo Oh
2026 Fall

Programs

Language:

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

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

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

Examples

- `x = 1;`
`p = &x;`
`*p = *p + 1`
- `p = alloc;`
`q = &p;`
`**q = 1`
- `x = 1;`
`while (x < 10) {`
 `p = alloc;`
 `*p = *p + x;`
 `x = x + 1`
`}`
- `if (...) { p = alloc; *p = 1; }`
 `else { p = alloc; *p = 2; }`
 `*p = 3`

Concrete Semantics

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

- $\llbracket lv \rrbracket : \textit{Mem} \rightarrow \textit{Loc}$:

$$\begin{aligned} \llbracket x \rrbracket(m) &= x \\ \llbracket *x \rrbracket(m) &= m(x) \end{aligned}$$

- $\llbracket e \rrbracket : \textit{Mem} \rightarrow \textit{Val}$:

$$\begin{aligned} \llbracket n \rrbracket(m) &= n \\ \llbracket lv \rrbracket(m) &= m(\llbracket lv \rrbracket(m)) \\ \llbracket \&lv \rrbracket(m) &= \llbracket lv \rrbracket(m) \\ \llbracket e_1 + e_2 \rrbracket(m) &= \llbracket e_1 \rrbracket(m) +_{\textit{Int}} \llbracket e_2 \rrbracket(m) \end{aligned}$$

- $\llbracket b \rrbracket : Mem \rightarrow Bool$:

$$\begin{aligned}
 \llbracket true \rrbracket(m) &= true \\
 \llbracket false \rrbracket(m) &= false \\
 \llbracket e_1 = e_2 \rrbracket(m) &= \llbracket e_1 \rrbracket(m) =_{Int} \llbracket e_2 \rrbracket(m) \\
 \llbracket e_1 \leq e_2 \rrbracket(m) &= \llbracket e_1 \rrbracket(m) \leq_{Int} \llbracket e_2 \rrbracket(m) \\
 \llbracket \neg b \rrbracket(m) &= \neg \llbracket b \rrbracket(m) \\
 \llbracket b_1 \wedge b_2 \rrbracket(m) &= \llbracket b_1 \rrbracket(m) \wedge \llbracket b_2 \rrbracket(m)
 \end{aligned}$$

- $f_n : \wp(Mem) \rightarrow \wp(Mem)$:

$$\begin{aligned}
 f_n(M) &= \{m \mid \llbracket lw \rrbracket(m) \mapsto \llbracket e \rrbracket(m) \mid m \in M\} \cdots \text{cmd}(n) = lw := e \\
 f_n(M) &= \{m \mid \llbracket lw \rrbracket(m) \mapsto l, l \mapsto 0 \mid m \in M\} \cdots \text{cmd}(n) = lw := \text{alloc}, \\
 &\quad l \text{ is new} \\
 f_n(M) &= \{m \in M \mid \llbracket b \rrbracket(m) = true\} \cdots \text{cmd}(n) = \text{assume}(b)
 \end{aligned}$$

- $F : (N \rightarrow \wp(Mem)) \rightarrow (N \rightarrow \wp(Mem))$:

$$F(X) = \lambda n. f_n\left(\bigcup_{n' \rightarrow n} X(n')\right)$$

- Collecting semantics:

$$\text{fix } F \in N \rightarrow \wp(Mem)$$

Abstract Domain

$$\begin{array}{ll}
 Mem & = Loc \rightarrow Val & \widehat{Mem} & = \widehat{Loc} \rightarrow \widehat{Val} \\
 Loc & = Var + HeapAddr & \widehat{Loc} & = Var + AllocSite \\
 Val & = Int + Loc & \widehat{Val} & = Interval \times \wp(\widehat{Loc})
 \end{array}$$

$$\bullet \wp(HeapAddr) \xleftrightarrow[\alpha_{HeapAddr}]{\gamma_{HeapAddr}} \wp(AllocSite)$$

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

$$\bullet \wp(Loc) \xleftrightarrow[\alpha_{Loc}]{\gamma_{Loc}} \wp(\widehat{Loc})$$

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

$$\bullet \wp(Int) \xleftrightarrow[\alpha_{Int}]{\gamma_{Int}} Interval$$

$$\bullet \wp(Val) \xleftrightarrow[\alpha_{Val}]{\gamma_{Val}} \widehat{Val}$$

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

$$\bullet \wp(Mem) \xleftrightarrow[\alpha_{Mem}]{\gamma_{Mem}} \widehat{Mem}$$

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

$$\bullet N \rightarrow \wp(Mem) \xleftrightarrow[\alpha]{\gamma} N \rightarrow \widehat{Mem}: \alpha(X) = \lambda n. \alpha_{Mem}(X(n))$$

Abstract Semantics

- $\llbracket lv \rrbracket : \widehat{Mem} \rightarrow \wp(Loc)$

$$\begin{aligned}\llbracket x \rrbracket(m) &= \{x\} \\ \llbracket *x \rrbracket(m) &= m(x).2\end{aligned}$$

- $\llbracket e \rrbracket : \widehat{Mem} \rightarrow Val$

$$\begin{aligned}\llbracket n \rrbracket(m) &= \langle [n, n], \emptyset \rangle \\ \llbracket lv \rrbracket(m) &= \bigsqcup_{l \in \llbracket lv \rrbracket(m)} m(l) \\ \llbracket \&lv \rrbracket(m) &= \langle \perp, \llbracket lv \rrbracket(m) \rangle \\ \llbracket e_1 + e_2 \rrbracket(m) &= \llbracket e_1 \rrbracket(m) \hat{+} \llbracket e_2 \rrbracket(m)\end{aligned}$$

- $\llbracket b \rrbracket : \widehat{Mem} \rightarrow Bool$

$$\begin{aligned}\llbracket true \rrbracket(m) &= true \\ \llbracket false \rrbracket(m) &= false \\ \llbracket e_1 = e_2 \rrbracket(m) &= \llbracket e_1 \rrbracket(m) \hat{=} \llbracket e_2 \rrbracket(m) \\ \llbracket e_1 \leq e_2 \rrbracket(m) &= \llbracket e_1 \rrbracket(m) \hat{\leq} \llbracket e_2 \rrbracket(m) \\ \llbracket \neg b \rrbracket(m) &= \neg \llbracket b \rrbracket(m) \\ \llbracket b_1 \wedge b_2 \rrbracket(m) &= \llbracket b_1 \rrbracket(m) \wedge \llbracket b_2 \rrbracket(m)\end{aligned}$$

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

$$\hat{f}_n(m) = m[x \mapsto \llbracket e \rrbracket(m)]$$

$$\hat{f}_n(m) = m[x \mapsto \llbracket e \rrbracket(m)]$$

$$\hat{f}_n(m) = \bigsqcup_{l \in \llbracket lv \rrbracket(m)} m[l \mapsto m(l) \sqcup \llbracket e \rrbracket(m)]$$

$$\hat{f}_n(m) = m[x \mapsto (\perp, \{n\}), n \mapsto ([0, 0], \emptyset)]$$

$$\hat{f}_n(m) = m[x \mapsto (\perp, \{n\}), n \mapsto ([0, 0], \emptyset)]$$

$$\hat{f}_n(m) = \bigsqcup_{l \in \llbracket lv \rrbracket(m)} m'[l \mapsto m(l) \sqcup (\perp, \{n\})]$$

$$\hat{f}_n(m) = \bigsqcup \{m' \sqsubseteq m \mid true \sqsubseteq \llbracket b \rrbracket(m')\}$$

$$\dots \text{cmd}(n) = x := e$$

$$\dots \text{cmd}(n) = lv := e, \\ \llbracket lv \rrbracket(m) = \{x\}$$

$$\dots \text{cmd}(n) = lv := e$$

$$\dots \text{cmd}(n) = x := \text{alloc}$$

$$\dots \text{cmd}(n) = lv := \text{alloc}, \\ \llbracket lv \rrbracket(m) = \{x\}$$

$$\dots \text{cmd}(n) = lv := \text{alloc} \\ m' = m[n \mapsto ([0, 0], \emptyset)]$$

$$\dots \text{cmd}(n) = \text{assume}(b)$$

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

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

- Abstract semantics:

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