COSE312: Compilers

Lecture 15 — Semantic Analysis (5)

Hakjoo Oh
2017 Spring

Announcement

- No class on next week (5/22, 5/24)
- Homework 3 is out (due 5/28)

Semantic Analysis (Static Analysis)

The goal is to prove the absence of certain types of semantic errors. For example, we aim to prove that

$$\text{If } x \text{ is a positive value, } f(x) \text{ is never } \mathbf{0}$$

for the following function:

```
int f(int x) {
  y := 1;
  while (x != 0) {
    y := y * x;
    x := x - 1;
  }
  return y;
}

x = f(5); y = 10 / x;
x = f(7); y = 10 / x;
...
x = f(2); y = 10 / x;
```

Semantic Analysis is Undecidable

For example, we cannot statically decide the possible values of x at the last statement:

$$\texttt{if } \cdots \texttt{ then } x := 1 \texttt{ else } (S; x := -1); y := x$$

The value of x is 1 if S does not terminate; otherwise, x can be either 1 or -1. Determining the value of x requires to solve the halting problem, which is undecidable in general.

Principle of Static Anaylsis

Static analysis aims to compute safe approximations of the program semantics.

$$\mathbf{12345 + 9873 * 5925 + (-5918) * (-881) = \ ?}$$

- Concrete semantics: 63,723,628
- Static analysis: [50,000,000, 100,000,000]
- Static analysis: a positive number
- Static analysis: an even number

"Abstract interpretation" of programs: e.g.,

$$\boldsymbol{p \mathbin{\hat{+}} p \mathbin{\hat{*}} p \mathbin{\hat{+}} n \mathbin{\hat{*}} n = p \mathbin{\hat{+}} p \mathbin{\hat{+}} p = p \mathbin{\hat{+}} p = p}$$

Example: Sign Analysis

```
int f(int x) {
  y := 1;
  while (x != 0) {
    y := y * x;
    x := x - 1;
  }
  return y;
}
```

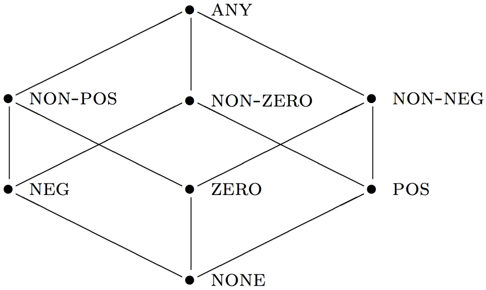
Abstract Domain and Semantics

Static analysis is defined with abstract domain and abstract semantics:

- abstract domain: abstract representation of program values
  - represented by a CPO
- abstract semantics: abstract interpretation of the concrete semantics of the program
  - represented by a monotone function $\boldsymbol{F}$

Abstract Domain of Sign Analysis

We abstract integers by the complete lattice $(\mathbf{Sign}, \sqsubseteq)$:



Abstract Domain of Sign Analysis

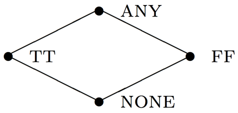
The meaning is defined by the abstraction and concretization functions:

$$
\begin{array}{rcl}
\alpha_{\mathbb{Z}} & : & \mathcal{P}(\mathbb{Z}) \rightarrow \textbf{Sign} \\
\alpha_{\mathbb{Z}}(Z) & = & \bigsqcup_{z \in Z} \alpha_1(z) \\
& & \text{where } \alpha_1(z) = \begin{cases} \text{NEG} & \cdots z < 0 \\ \text{ZERO} & \cdots z = 0 \\ \text{POS} & \cdots z > 0 \end{cases}
\end{array}
$$

$$
\begin{array}{rcl}
\gamma_{\mathbb{Z}} & : & \textbf{Sign} \rightarrow \mathcal{P}(\mathbb{Z}) \\
\gamma_{\mathbb{Z}}(\text{NONE}) & = & \emptyset \\
\gamma_{\mathbb{Z}}(\text{POS}) & = & \{z \mid z > 0\} \\
\gamma_{\mathbb{Z}}(\text{NEG}) & = & \{z \mid z < 0\} \\
\gamma_{\mathbb{Z}}(\text{ZERO}) & = & \{0\} \\
\gamma_{\mathbb{Z}}(\text{NON-POS}) & = & \{z \mid z \leq 0\} \\
\gamma_{\mathbb{Z}}(\text{NON-NEG}) & = & \{z \mid z \geq 0\} \\
\gamma_{\mathbb{Z}}(\text{NON-ZERO}) & = & \{z \mid z \neq 0\} \\
\gamma_{\mathbb{Z}}(\text{ANY}) & = & Z
\end{array}
$$

Abstract Domain of Sign Analysis

The truth values $\mathbf{T} = \{true, false\}$ are abstracted by the complete lattice $(\widehat{\mathsf{T}}, \sqsubseteq)$:



Exercise) Define the abstraction and concretization functions:

$$\alpha_{\mathbf{T}} : \mathcal{P}(\mathbf{T}) \to \widehat{\mathsf{T}}, \qquad \gamma_{\mathbf{T}} : \widehat{\mathsf{T}} \to \mathcal{P}(\mathbf{T})$$

Abstract Memory State

The complete lattice of abstract states:

$$\widehat{\textbf{State}} = \mathit{Var} \rightarrow \textbf{Sign}$$

with the pointwise ordering $\sqsubseteq$:

$$\hat{s}_1 \sqsubseteq \hat{s}_2 \iff \forall x \in \mathit{Var}.\ \hat{s}_1(x) \sqsubseteq \hat{s}_2(x).$$

The least upper bound: for $Y \subseteq \widehat{\textbf{State}}$,

$$\bigsqcup Y = \lambda x.\ \bigsqcup_{\hat{s} \in Y} \hat{s}(x)$$

Lemma

Let S be a non-empty set and $(D, \sqsubseteq)$ be a poset. Then, the poset $(S \rightarrow D, \sqsubseteq)$ with the ordering

$$f_1 \sqsubseteq f_2 \iff \forall s \in S.\ f_1(s) \sqsubseteq f_2(s)$$

is a complete lattice if D is a complete lattice, and it is a CPO if D is a CPO.

Abstract Memory State

The abstraction and concretization functions for the abstract states:

$$\alpha : \mathcal{P}(\mathrm{State}) \to \widehat{\textsf{State}}$$

$$\alpha(S) = \lambda x.\ \bigsqcup_{s \in S} \alpha_{\mathbb{Z}}(\{s(x)\})$$

$$\gamma : \widehat{\textsf{State}} \to \mathcal{P}(\mathrm{State})$$

$$\gamma(\hat{s}) = \{s \in \mathrm{State} \mid \forall x \in \mathit{Var}.\ s(x) \in \gamma_{\mathbb{Z}}(\hat{s}(x))\}$$

Abstract Semantics

The abstract semantics of arithmetic expressions:

$$
\begin{array}{rcl}
\widehat{\mathcal{A}}[\![\, a \,]\!] & : & \widehat{\textbf{State}} \to \textbf{Sign} \\
\widehat{\mathcal{A}}[\![\, n \,]\!](\hat{s}) & = & \alpha_{\mathbb{Z}}(\{n\}) \\
\widehat{\mathcal{A}}[\![\, x \,]\!](\hat{s}) & = & \hat{s}(x) \\
\widehat{\mathcal{A}}[\![\, a_1 + a_2 \,]\!](\hat{s}) & = & \widehat{\mathcal{A}}[\![\, a_1 \,]\!](\hat{s}) +_S \widehat{\mathcal{A}}[\![\, a_2 \,]\!](\hat{s}) \\
\widehat{\mathcal{A}}[\![\, a_1 \star a_2 \,]\!](\hat{s}) & = & \widehat{\mathcal{A}}[\![\, a_1 \,]\!](\hat{s}) \star_S \widehat{\mathcal{A}}[\![\, a_2 \,]\!](\hat{s}) \\
\widehat{\mathcal{A}}[\![\, a_1 - a_2 \,]\!](\hat{s}) & = & \widehat{\mathcal{A}}[\![\, a_1 \,]\!](\hat{s}) -_S \widehat{\mathcal{A}}[\![\, a_2 \,]\!](\hat{s})
\end{array}
$$

Abstract Semantics

$+_S$	NONE	NEG	ZERO	POS	NON-POS	NON-ZERO	NON-NEG	ANY
NONE	NONE	NONE	NONE	NONE	NONE	NONE	NONE	NONE
NEG	NONE	NEG	NEG	ANY	NEG	ANY	ANY	ANY
ZERO	NONE	POS	ZERO	POS	NON-POS	NON-ZERO	NON-NEG	ANY
POS	NONE	ANY	POS	POS	ANY	ANY	POS	ANY
NON-POS	NONE	NEG	NON-POS	ANY	NON-POS	ANY	ANY	ANY
NON-ZERO	NONE	ANY	NON-ZERO	ANY	ANY	ANY	ANY	ANY
NON-NEG	NONE	ANY	NON-NEG	POS	ANY	ANY	NON-NEG	ANY
ANY	NONE	ANY	ANY	ANY	ANY	ANY	ANY	ANY

$\star_S$	NEG	ZERO	POS
NEG	POS	ZERO	NEG
ZERO	ZERO	ZERO	ZERO
POS	NEG	ZERO	POS

$-_S$	NEG	ZERO	POS
NEG	ANY	NEG	NEG
ZERO	POS	ZERO	NEG
POS	POS	POS	ANY

Abstract Semantics

The abstract semantics of boolean expressions:

$$
\begin{array}{rcl}
\widehat{\mathcal{B}}[\![b]\!] & : & \widehat{\textbf{State}} \to \widehat{\textbf{T}} \\
\widehat{\mathcal{B}}[\![\texttt{true}]\!](\hat{s}) & = & \mathrm{TT} \\
\widehat{\mathcal{B}}[\![\texttt{false}]\!](\hat{s}) & = & \mathrm{FF} \\
\widehat{\mathcal{B}}[\![a_1 = a_2]\!](\hat{s}) & = & \widehat{\mathcal{B}}[\![a_1]\!](\hat{s}) =_S \widehat{\mathcal{B}}[\![a_2]\!](\hat{s}) \\
\widehat{\mathcal{B}}[\![a_1 \leq a_2]\!](\hat{s}) & = & \widehat{\mathcal{B}}[\![a_1]\!](\hat{s}) \leq_S \widehat{\mathcal{B}}[\![a_2]\!](\hat{s}) \\
\widehat{\mathcal{B}}[\![\neg b]\!](\hat{s}) & = & \neg_S \widehat{\mathcal{B}}[\![b]\!](\hat{s}) \\
\widehat{\mathcal{B}}[\![b_1 \wedge b_2]\!](\hat{s}) & = & \widehat{\mathcal{B}}[\![b_1]\!](\hat{s}) \wedge_S \widehat{\mathcal{B}}[\![b_2]\!](\hat{s})
\end{array}
$$

Abstract Semantics

$=_S$	NEG	ZERO	POS
NEG	ANY	FF	FF
ZERO	FF	TT	FF
POS	FF	FF	ANY

$\leq_S$	NEG	ZERO	POS
NEG	ANY	TT	TT
ZERO	FF	TT	TT
POS	FF	FF	ANY

$\neg_T$	
NONE	NONE
TT	FF
FF	TT
ANY	ANY

$\wedge_T$	NONE	TT	FF	ANY
NONE	NONE	NONE	NONE	NONE
TT	NONE	TT	FF	ANY
FF	NONE	FF	FF	FF
ANY	NONE	ANY	FF	ANY

Abstract Semantics

$$
\begin{array}{rcl}
\widehat{\mathcal{C}}[\![c]\!] & : & \widehat{\textsf{State}} \rightarrow \widehat{\textsf{State}} \\
\widehat{\mathcal{C}}[\![x := a]\!] & = & \lambda \hat{s}.\hat{s}[x \mapsto \widehat{\mathcal{A}}[\![a]\!](\hat{s})] \\
\widehat{\mathcal{C}}[\![\texttt{skip}]\!] & = & \textsf{id} \\
\widehat{\mathcal{C}}[\![c_1; c_2]\!] & = & \widehat{\mathcal{C}}[\![c_2]\!] \circ \widehat{\mathcal{C}}[\![c_1]\!] \\
\widehat{\mathcal{C}}[\![\texttt{if}\ b\ c_1\ c_2]\!] & = & \widehat{\textsf{cond}}(\widehat{\mathcal{B}}[\![b]\!], \widehat{\mathcal{C}}[\![c_1]\!], \widehat{\mathcal{C}}[\![c_2]\!]) \\
\widehat{\mathcal{C}}[\![\texttt{while}\ b\ c]\!] & = & \mathit{fix} \widehat{F} \\
 & & \text{where } \widehat{F}(g) = \widehat{\textsf{cond}}(\widehat{\mathcal{B}}[\![b]\!], g \circ \widehat{\mathcal{C}}[\![c]\!], \textsf{id})
\end{array}
$$

$$
\widehat{\textsf{cond}}(f, g, h)(\hat{s}) = \begin{cases} \bot & \cdots f(\hat{s}) = \mathrm{NONE} \\ f(\hat{s}) & \cdots f(\hat{s}) = \mathrm{TT} \\ g(\hat{s}) & \cdots f(\hat{s}) = \mathrm{FF} \\ f(\hat{s}) \sqcup g(\hat{s}) & \cdots f(\hat{s}) = \mathrm{ANY} \end{cases}
$$

Examples

- ```
  x := 0;
  y := 1;
  if (x == y)
    z := 1
  else
    z := -1
  ```
- ```
  x := 0;
  y := -1;
  while (x < 10) {
    x := x + 1;
    y := y + 1;
  }
  ```

cf) Other Abstract Domains

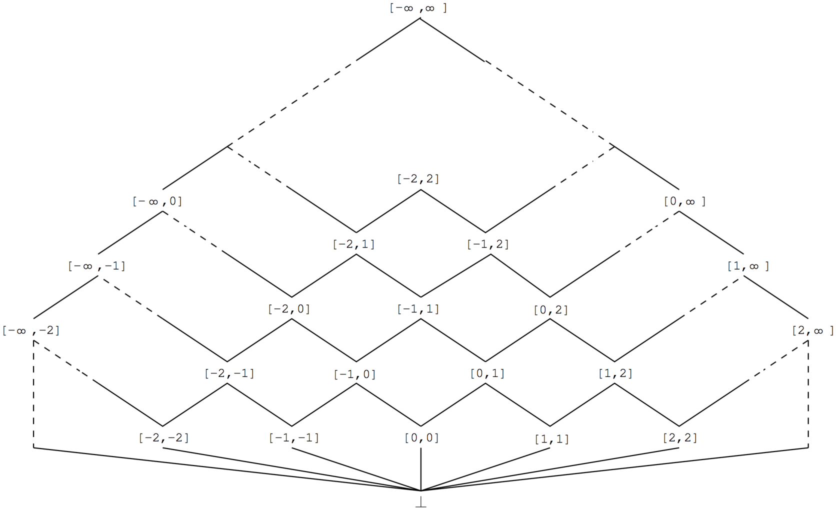
Motivating example:

```
char a[10], b[10];
int x = input();
if (x > 0)
  if (x < 10)
    memcpy(a, b, x);
```

cf) Other Abstract Domains

The interval complete lattice:

$$\mathbb{I} = \{\bot\} \cup \{[l, u] \mid l, u \in \mathbb{Z} \cup \{-\infty, +\infty\} \ \wedge \ l \leq u\}$$



cf) Other Abstract Domains

The interval domain cannot infer the relationships between variables:

```
i = 0;
p = 0;

while (i < 12) {
    i = i + 1;
    p = p + 1;
}
assert(i==p)
```

Interval analysis

i	[12,12]
p	[0,+oo]

Octagon analysis

i	[12,12]
p	[12,12]
p-i	[0,0]
p+i	[24,24]

Summary

- Approaches to specifying semantics of programs
  - Big-step operational semantics
  - Small-step operational semantics
  - Denotational semantics
- Semantic analysis by safely approximating the program semantics
  - Sign analysis, interval analysis, octagon analysis, etc