Abstract Interpretation, cont.

Announcements

- HW3?
- HW4 is optional
- HW5 (proofs Abstract Interpretation and Haskell) coming up on Monday
  - Please start early 😊

Outline

- Abstract interpretation
- Semantics
- Notion of abstraction (i.e., abstraction relation)
- Concretization and abstraction functions
- Galois Connections
- Applications of abstract interpretation
- Termination

Towards Concretization and Abstraction Functions

- Abstraction relation is consistent with order!
  - Concrete order:
    - If \( c_0 \preceq c_1 \) and \( c_1 \) is represented by \( a \), then \( c_0 \) is represented by \( a \)
  - Abstract order:
    - If \( a_s \leq a_t \) and \( c \) is represented by \( a_s \), then \( c \) is represented by \( a_t \)
    - Advantage: reasoning in abstract space proceeds by going up the abstract lattice; property preserves the abstraction relation

Abstraction Relation is Consistent with Partial Orders!

Concrete lattice:  

Abstract lattice:
Abstraction Relation is Consistent with Partial Orders!

Concrete lattice:

Abstract lattice:

Which abstraction is “best”?

Towards Concretization and Abstraction Functions

- Previous slides, more formally
- Concrete lattice $C$, $\subseteq$ and abstract lattice $A, \leq$
- Abstraction relation is consistent with ordering:
  - For every $c_0, c_1 \in C$ and every $a \in A$, $c_0 \subseteq c_1$ and $c_1 \Rightarrow a \Rightarrow c_0 \Rightarrow a$
  - For every $a_0, a_1 \in A$ and every $c \in C$, $a_0 \leq a_1$ and $c \rightleftharpoons a_0 \Rightarrow c \rightleftharpoons a_1$
- The abstraction relation makes sense but is inconvenient. We need functions!
  - Concretization function: $A \Rightarrow C$
  - Abstraction function: $C \Rightarrow A$

Concretization Function

- Definition:
  - Concretization function $\gamma : A \Rightarrow C$ (if it exists) maps $a \in A$ to the largest (most general) element $c \in C$ such that $c \leftarrow a$
  - Note: $\gamma(a)$ “covers” all concrete elements that are represented by $a$
  - $\gamma(a)$ returns the most general element $c$ such that $c$ is represented by $a$. This is called concretization

Gamma Examples

Concrete lattice:

Abstract lattice:

Abstraction Function

- Definition:
  - Abstraction function $\alpha : C \Rightarrow A$ (if it exists) maps $c \in C$ to the smallest (most precise) element $a \in A$ such that $c \leftarrow a$
  - $\alpha$ maps $c$ to the most precise $a$ such that $a$ represents $c$. This is called best abstraction

Alpha Examples

Concrete lattice:

Abstract lattice:
Concretization Function Examples

- Concretization of lattice of signs
  - $\gamma_S(T) \rightarrow \mathbb{Z}$
  - $\gamma_S(\pm) \rightarrow \{1,2,3,\ldots\}$
  - $\gamma_S(\bot) \rightarrow \{-\infty,-3,-2,-1\}$
  - $\gamma_S(0) \rightarrow \{0\}$
  - $\gamma_S(\top) \rightarrow \{\}$

- Concretization of lattice of intervals
  - $\gamma_I([a,b]) \rightarrow \{a,a+1,\ldots,b-1,b\}$
  - $\gamma_I(T)$ etc.

Abstraction Function Examples

- Signs abstraction
  - $\alpha_S(c) \rightarrow \begin{cases} 
  \bot & \text{if } c = \{\}\text{ or } c = T \\
  0 & \text{if } c = \{0\} \\
  + & \text{if } \forall n \in c, n > 0 \\
  - & \text{if } \forall n \in c, n < 0 \\
  T & \text{otherwise}
  \end{cases}$

- Constants abstraction
  - $\alpha_C(c) \rightarrow \begin{cases} 
  \bot & \text{if } c = \{\} \\
  n & \text{if } c = \{n\} \\
  T & \text{otherwise}
  \end{cases}$

Outline

- Abstract interpretation
- Semantics
- The abstraction relation
- Concretization and abstraction functions
- Galois Connections
- Applications of abstract interpretation
- Termination

Galois Connection

A Galois Connection links $\alpha$ and $\gamma$ so that they refer to the same abstraction relation $\text{abstraction}\Rightarrow$!

Definition

A Galois connection is defined by concrete lattice $(C, \subseteq)$, abstract lattice $(A, \leq)$, an abstraction function $\alpha : C \rightarrow A$ and concretization function $\gamma : A \rightarrow C$ such that

- for every $a \in A$ and every $c \in C$, $c \subseteq \gamma(a)$ if and only if $\alpha(c) \leq a$

Galois Connection Example

Concrete lattice: $\gamma$

Abstract lattice: $\alpha$

- $\alpha_C(c) \rightarrow \bot$ if $c = \{\}$
- $\alpha_C(c) \rightarrow n$ if $c = \{n\}$
- $\alpha_C(c) \rightarrow T$ otherwise

- $\gamma_C(T) \rightarrow Z$
- $\gamma_C(n) \rightarrow \{n\}$
- $\gamma_C(\bot) \rightarrow \{\}$

- If $c = \{\}$, then for every $a$, $\{\} \subseteq \gamma_C(a)$ and $\alpha_C(\{\}) \leq a$

- If $c = \{n\}$, then $c \subseteq \gamma_C(a)$ or $\alpha_C(c) \leq a$ for $a = n$ or $a = T$. In either case equivalence holds

- If $c$ is any other set, then $c \subseteq \gamma_C(a)$ or $\alpha_C(c) \leq a$ only for $a = T$ (Since $c$ has at least 2 elements.) Then $c \subseteq \gamma_C(T) = Z$ and $\alpha_C(c) \leq T$ hold.
Galois Connection Example

- Signs lattice
  
  \[ a_d(c) \to + \text{ if for every } n \in c, n > 0 \]
  
  \[ a_d(c) \to - \text{ if for every } n \in c, n < 0 \]
  
  \[ a_d(c) \to 0 \text{ if } c = \{0\} \]
  
  \[ a_d(c) \to 1 \text{ if } c = \emptyset \]
  
  \[ a_c(c) \to T \text{ otherwise} \]

\[ \gamma_c(T) \to \mathbb{Z} \]

\[ \gamma_c(+) \to \{1, 2, \ldots\} \]

\[ \gamma_c(0) \to \{0\} \]

\[ \gamma_c(-) \to \{-\ldots, -2, -1\} \]

\[ \gamma_c(\cup) \to \emptyset \]

Galois Connection Properties

- For every \( a \in A \) and every \( c \in C \)
  
  \[ c \subseteq \gamma(a) \text{ if and only if } a(c) \leq a \]

- **Monotonicity:**
  
  \( a \) is monotone

  \( \gamma \) is monotone

Proof: homework

Galois Connection Properties

- For every \( a \in A \) and every \( c \in C \)
  
  \[ c \subseteq \gamma(a) \text{ if and only if } a(c) \leq a \]

- **Repetition:**
  
  \( a \circ \gamma \circ a = a \)

  \( \gamma \circ a \circ \gamma = \gamma \)

Proof: exercise

Other Properties

- **Distributivity (preservation of upper bounds):**
  
  For every \( c_0, c_1 \) in \( C \),
  
  \[ a(c_0 \lor c_1) = a(c_0) \lor a(c_1) \]

- **Uniqueness**
  
  For a given \( \gamma : A \to C \), there exists at most one
  
  \( \alpha : C \to A \) such that \( \gamma \) and \( \alpha \) form a Galois connection

  For a given \( \alpha : C \to A \), there exists at most one
  
  \( \gamma : A \to C \) such that \( \gamma \) and \( \alpha \) form a Galois connection

Outline

- Abstract interpretation
- Semantics
- The abstraction relation
- Concretization and abstraction functions
- Galois Connections
- Applications of abstract interpretation
- Termination
Applications of Abstract Interpretation
- Deriving and reasoning about static analysis!
- Deriving a static analysis
- The hard (ad-hoc) way...
  - Starting from some concrete space and semantics
  - Define abstract space and abstract semantics
  - Guess an invariant
  - Make an inductive argument that each transfer function preserves the invariant
  - Invariant must imply correctness (soundness) conclusion

An Example: The Semantics
- We will derive a static analysis that computes signs \((+,-,0)\) from a collecting semantics and an abstraction
- Semantics (partial)
  - Assume fixed set of integer variables: \(x,y,z\)
  - \(\sigma = (n_x,n_y,n_z)\)
    - \(\sigma\) is a mapping from integer variables to values in \(\mathbb{Z}\)
  - concrete state \(S\) is a set of \(\sigma\)’s
    - \([x=n] \Rightarrow (\sigma[x\leftarrow n] \mid \sigma \in S)\)
    - \([x=y + z] \Rightarrow (\sigma[x\leftarrow y]|\sigma[z]|\sigma) \mid \sigma \in S)\)
  - \([x=n] \Rightarrow (\sigma[x\leftarrow n]|\sigma) \mid \sigma \in S)\)

The Abstraction
- \(\alpha\) is a composition of two abstractions
  - Collecting abstraction: collects the values of a variable across all \(\sigma\)’s into one set
  - Signs abstraction: our running example
    - \(\alpha : S \rightarrow (\{n_x | (n_{x_{\alpha}} \in S)\},\{n_y | (\ldots) \in S\},\{n_z | (\ldots) \in S\})\)
    - E.g. \(S = \{(1,2,0),(-1,3,0)\}\)
    - \(\alpha(S) = (\alpha(1,-1),\alpha(2,3),\alpha(0)) = (T,+,0)\)
      - i.e., in the abstract, \(x\) is \(T\), \(y\) is \(+\), \(z\) is \(0\)

The Abstraction
- Abstract state \(\hat{S}\) is a tuple \((a_x,a_y,a_z)\)
  - \(\gamma : \hat{S} \rightarrow \{(n_x,n_y,n_z) | n_x \in \gamma_S(a_x), n_y \in \gamma_S(a_y), n_z \in \gamma_S(a_z)\}\)
  - E.g., from previous slide:
    - \(S = \{(1,2,0),(-1,3,0)\}\)
    - \(\alpha(S) = (\alpha(1,-1),\alpha(2,3),\alpha(0)) = (T,+,0)\)
    - \(\gamma(\hat{S}) = ?\)
      - \(\hat{S} = (T,+,0)\)
      - \(\gamma(\hat{S}) = \{(1,2,0),(-1,2,0),(-2,3,0)\}\)
  - Exercise: Prove \(\alpha, \gamma\) form Galois connection

Deriving the Abstract Semantics
- We can derive an abstract semantics by “mechanical” transformation:
  - Applying \(\alpha\) on the result of a concrete transfer function \([\text{Stmt}]\) (i.e., \(\alpha([\text{Stmt}][S])\) derives an abstract transfer function \([\text{Stmt}][\hat{S}]\)
  - Theorem: \(\alpha([\text{Stmt}][S]) = [\text{Stmt}][\alpha(S)]\), written \(\alpha \circ [\text{Stmt}] = [\text{Stmt}] \circ \alpha\) implies static analysis is sound (i.e., computes an over-approximation)
  - The exact alpha-based transformation does not always work

Exercise: Prove \(\alpha, \gamma\) form Galois connection

An "easier" way...
- Start from concrete space \((C,\subseteq)\) and semantics
- Define abstract space \((A,\subseteq)\)
- Define a Galois Connection by setting \(\alpha\) and \(\gamma\)
- Derive abstract semantics
  - "Mechanical" transformation:
    - Use \(\alpha\) when it exists
    - Use \(\gamma\) otherwise (more advanced abstract interpretation)
- Correctness (soundness) holds!

\[ S\hat{=} (T,+,0), \gamma(S) = \{(1,2,0),(-1,2,0),(-2,3,0)\} \]
The Abstract Semantics

- Deriving the abstract semantics:
- Abstracting result of concrete transfer function:
  \[ a^{\alpha}[[\text{Stmt}]](S) = (a_\alpha \left( \sum_{n} \frac{1}{n_i} \in \|\text{Stmt}\|]/(S)) \],
  \[ a_\alpha \left( \sum_{n} \frac{1}{n_i} \in \|\text{Stmt}\|]/(S) \]
  \[ a_\alpha \left( \sum_{n} \frac{1}{n_i} \in \|\text{Stmt}\|]/(S) \]

Read: apply concrete transfer function \[\|\text{Stmt}\|]/(S)\]
resulting in a new set of \(\alpha(\mu_n\mu_n\mu_n)\)'s. Then
1. apply collecting abstraction: collect all values of \(x, y, \)
and \(z\) into respective sets,
2. apply sign abstraction on each set.

The Abstract Semantics

\[ ([x=n]) = \hat{S}(x \leftarrow a_\alpha((n))) \]
\[ ([x+y+z]) = \hat{S}(x \leftarrow a_\alpha((y))) \]
\[ ([x+y+z]) = \hat{S}(x \leftarrow a_\alpha((y))) \]

Prove: applying abstract transfer function on abstract value \(a(S)\) yields same result as applying concrete transfer function on \(S\), then applying \(a\) on the result: \([[\text{Stmt}]]^x a \circ [[\text{Stmt}]]^y\)
Guarantees soundness!

The Abstract Semantics

<table>
<thead>
<tr>
<th>Operation</th>
<th>T</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>(x=n)</td>
<td>T</td>
<td>F</td>
</tr>
<tr>
<td>(x+y)</td>
<td>T</td>
<td>T</td>
</tr>
</tbody>
</table>

Dataflow Analysis

For each concrete transfer function we derive the corresponding abstract transfer function:

\[ a^\ast([x=n]) =\]

\[ (a_\alpha((n))) \]

\[ a_\alpha((n)) =\]

3. if \((x>0)\)

4. \(y=x-y\)

Define transfer functions, then
apply fixpoint iteration.

Abstract Interpretation Does Better

\[ ([x+y+z]) = \hat{S}(x \leftarrow a_\alpha((n))) \]

E.g., applies \(f_4\) on \((\ast, \ast, \ast)\)
\(\text{in}(S) = (T, T)\)

3. if \((x>0)\)

4. \(y=x-y\)

\(\text{E.g., applies } f_4 \text{ on } (\ast, \ast, \ast)\)
\(\text{in}(S) = (T, T)\)

5. \(\text{Out}(4) = (T, T)\)
Abstract Interpretation is Even More General!

- \[|\text{Stmt}| \circ \alpha \ast |\text{Stmt}| = |\text{Stmt}| \circ \alpha\] may not exist for some \(|\text{Stmt}|\)
- For some abstract domains, \(\alpha\) does not exist!
- For some abstract domains, the lattice does not have finite height!
- Abstract Interpretation allows building analyses even in cases like these. Uses \(\gamma\):
  - \[|\text{Stmt}| \circ \gamma(\mathcal{S}) \subseteq \gamma(|\text{Stmt}|(\mathcal{S}))\]
  - Sound (over-approximation) but less precise

For some abstract domains, the lattice does not have finite height!

Abstract Interpretation allows building analyses even in cases like these. Uses \(\gamma\):

- \[|\text{Stmt}| \circ \gamma(\mathcal{S}) \subseteq \gamma(|\text{Stmt}|(\mathcal{S}))\]
- Sound (over-approximation) but less precise

Widening

- What if the abstract lattice does not have finite height?
- E.g., lattice of intervals, a popular abstract domain
  - \[\cdots [-3,-1] [-2,-1] [-1,-1] [0,1] [1,3] \cdots \]

Widening: Over-approximates join \(\vee\), for correctness

- Guarantees termination, or faster convergence on programs with loops

\[
x = 0; \\
\text{while (true) \{ if (x < 9999) x = x+1; else x = -x; }
\]

Typical widening:

\[
[a,b] \vee [a,b] = [a, +\infty) \text{ if } b < b_i \\
[a,b] \vee [a,b] = [a,b] \text{ otherwise}
\]

(Read: if interval grows with next iteration through the loop, widen to infinity.)

Abstract Interpretation, Conclusion

- A general framework
  - Building static analyses!
  - Reasoning about correctness of static analysis
  - Comparing static analyses

Active area of research

- New applications of abstract interpretation
- Proof assistants
- New abstract domains
- Faster analysis techniques

Next Time

- Types and Type Based Analysis (plan for next two weeks or so)
  - Pure Lambda calculus
  - Typed Lambda calculus
  - Type soundness: progress and preservation
  - Simple type inference
  - Hindley-Milner type inference and polymorphism