Abstract

Abstract interpretation is a semantic approximation theory which has mainly been used for the design of static program analyzers. Our objective is to explain and illustrate the notion of abstraction/concretization and its numerous variants which are commonly used in abstract interpretation to formalize the loss of information. We also explain how the concrete model can be transformed into an abstract semantic model, and inversely for refinement.

Several examples are given for the design of programming language semantics as well as model-checking and program analysis algorithms. To illustrate the notions of relative completeness and of existence of a best abstraction, we show that transitional, demonic, natural and angelic denotational, predicate transformer and axiomatic semantics are all relatively complete, best abstractions of a maximal trace semantics (or equivalently that the maximal trace semantics is a refinement of all these semantics). To illustrate incompleteness, we consider model-checking of finite transition systems for a temporal logic, both with maximal trace semantics. The logic can be restricted to ensure relative completeness at the expense of expressiveness. To illustrate inexistence of best approximations, we consider several abstract domains for the abstraction of sets of vectors of numbers and sets of graphs (for so-called set-based analysis).

References

Abstraction in abstract interpretation

• Abstraction is understood as an approximation:

Abstraction in abstract interpretation, ©P. Cousot, Nov 16, 1999

Properties as sets

• A property is the set of objects which have this property;
• Example (properties of integers):
  - Positive: \{1, 2, 3, 4, \ldots\}
  - Odd: \{1, 3, 5, 7, \ldots\}
• There is often a confusion on the fact that abstract interpretation does not deal with abstract objects but with abstract properties of objects;
• This is because the two notions sometime coincide;
• The view of abstract interpretation as abstraction of properties is more powerful that pseudo-evaluation on abstract objects.

Abstraction in abstract interpretation, ©P. Cousot, Nov 16, 1999

Example: rule of signs

Standard semantics

• Standard semantics:
  - Operational: what are the steps of evaluation of the expression when knowing an assignment of values to the free variables;
  - Example (\(\rho = [x : 5, y : -3]\)):
    \[
    \llbracket x \times x + y \times y \rrbracket_\rho \\
    \rightarrow (\llbracket x \rrbracket_\rho \times \llbracket x \rrbracket_\rho) + (\llbracket y \rrbracket_\rho \times \llbracket y \rrbracket_\rho) \\
    \rightarrow (5 \times 5) + (-3 \times -3) \\
    \rightarrow 25 + 9 \\
    \rightarrow 34
    \]

Abstraction in abstract interpretation, ©P. Cousot, Nov 16, 1999
• **Denotational:** what is the value of the expression when knowing an assignment of values to the free variables:

\[
\llbracket e \rrbracket \in (X \mapsto \mathbb{Z}) \mapsto \mathbb{Z}
\]

\[
\llbracket n \rrbracket \rho = n
\]

\[
\llbracket x \rrbracket \rho = \rho(x)
\]

\[
\llbracket e_1 \times e_2 \rrbracket \rho = \llbracket e_1 \rrbracket \times \llbracket e_1 \rrbracket
\]

\[
\llbracket e_1 + e_2 \rrbracket \rho = \llbracket e_1 \rrbracket + \llbracket e_1 \rrbracket
\]

Example (\(\rho = [x : +1, y : -1]\)):

\[
\llbracket x \times x + y \times y \rrbracket \rho
\]

\[
\rightarrow (\llbracket x \rrbracket \rho \times \llbracket y \rrbracket \rho) + (\llbracket y \rrbracket \rho \times \llbracket y \rrbracket \rho)
\]

\[
\rightarrow (+1 \times +1) + (-1 \times -1)
\]

\[
\rightarrow +1 + +1
\]

\[
\rightarrow +1
\]

**Correctness:** the rule of signs is a step by step simulation of the standard semantics (inconclusive when no rule applies e.g. \(+1 + -1 = ?\)

• Same idea in “subject reduction” of type theory.

---

**Example: rule of signs**

1 — the abstract object point of view

• **Objective:** determine the sign of an expression;

• **Pseudo-evaluation** method:
  - replace values by their signs;
  - interpret arithmetic operators on signs:
    \(+1 + +1 = +1,\)
    \(+1 \times -1 = -1,\) etc.

• **Abstraction:**
  - concrete object \(\rightarrow\) abstract object
    - integer \(\rightarrow\) sign
    - integer \(\times\) integer \(\rightarrow\) integer \(\rightarrow\) sign \(\times\) sign \(\rightarrow\) sign

**Example: rule of signs**

2 — the abstract property point of view

• **Property** of an expression: set of its possible semantics;

• **Collecting semantics:** the strongest program property:

\[
\llbracket e \rrbracket \in \rho((X \mapsto \mathbb{Z}) \mapsto \mathbb{Z})
\]

\[
\llbracket e \rrbracket \triangleq \{ \llbracket e \rrbracket \} \tag{1}
\]

• **Abstract semantics:** a computable approximation of the collecting semantics.
Approximation

• Two alternatives:
  - Universal/from above: consider a superset of the possible cases,
  - Existential/from below: consider a subset of the possible cases;
• By duality, only universal approximation need to be formally studied;
• The rule of signs is a universal approximation (i.e. $+1 + +1 = +1$ is valid whether $3 + 2 = 5$ or $3 + 2 = 1789!$) since more cases are considered than possible.

Approximation for the rule of signs

- $\alpha_0 : \mathbb{Z} \leftrightarrow S$ where $S \doteq \{+1, 0, -1\}$
  $\alpha_0(n) = -1$ iff $n < 0$
  $\alpha_0(n) = 0$ iff $n = 0$
  $\alpha_0(n) = +1$ iff $n > 0$

- $\alpha_1 \in \wp(\mathbb{Z}) \mapsto \wp(S)$
  $\alpha_1(N) \doteq \{\alpha_0(n) \mid n \in N\}$
  $\gamma_1 \in \wp(S) \mapsto \wp(\mathbb{Z})$
  $\gamma_1(S) \doteq \{n \mid \alpha_0(n) \in S\}$

Example:
$\{0, 17\} \xrightarrow{\alpha_1} \{0, +1\} \xrightarrow{\gamma_1} \{0, 1, \ldots, 17, \ldots\}$

The lattice of signs [4]:

\[
\begin{array}{c}
\{1\} \\
\{0\} \\
\{-1\} \\
\{-1,0,1\}
\end{array}
\]

Example:
\[
\begin{array}{c}
\{0, 17\} \xrightarrow{\alpha_1} \{0, +1\} \xrightarrow{\gamma_1} \{0, 1, \ldots, 17, \ldots\}
\end{array}
\]

Reference


Abstraction in abstract interpretation, © P. Cousot, Nov 16, 1999
Abstraction in abstract interpretation.

- $\alpha_3 : (X \mapsto \wp(Z)) \mapsto (X \mapsto \wp(S))$
  - $\alpha_3(\rho) = \lambda X \cdot \alpha_1(\rho(X))$

$\gamma_3 : (X \mapsto \wp(S)) \mapsto (X \mapsto \wp(Z))$
  - $\gamma_3(\rho) = \lambda X \cdot \gamma_1(\rho(X))$

Example:
- $\alpha_3 \mapsto [X : \{0, 5\}, Y : \{0, 5\}]$
- $\gamma_3 \mapsto [X : \{0, +1\}, Y : \{0, +1\}]$
- $\gamma_1 \mapsto [X : \mathbb{N}, Y : \mathbb{N}]$

Example:
- $\{[X : 0, Y : 0], [X : 5, Y : 5]\}$
- $\alpha_3 \mapsto [X : \{0, +1\}, Y : \{0, +1\}]$
- $\gamma_3 \mapsto \{[X : n, Y : m] \mid n \in \mathbb{N} \land m \in \mathbb{N}\}$

Intuition:
- $\wp(X \mapsto Z) \xrightarrow{s} \wp(Z)$
- $\alpha_4 \xrightarrow{\gamma_4} \alpha_1 \xrightarrow{\gamma_1} \wp(S) \xrightarrow{S} \wp(S)$

- $\alpha_5 \in \wp(X \mapsto Z) \mapsto \wp(Z) \mapsto (\wp(X \mapsto Z) \mapsto \wp(S))$
  - $\alpha_5(s) = \alpha_1 \circ s \circ \gamma_4$

$\gamma_5 \in (\wp(X \mapsto Z) \mapsto \wp(S)) \mapsto (\wp(X \mapsto Z) \mapsto \wp(Z))$
  - $\gamma_5(S) = \gamma_1 \circ S \circ \alpha_4$

- $\alpha_6 \in \wp((X \mapsto Z) \mapsto Z) \mapsto (\wp(X \mapsto Z) \mapsto \wp(Z))$
  - $\alpha_6(S) = \lambda R \cdot \{s(\rho) \mid s \in S \land \rho \in R\}$

$\gamma_6 \in (\wp(X \mapsto Z) \mapsto \wp(Z)) \mapsto (\wp(X \mapsto Z) \mapsto Z)$
  - $\gamma_6(S) = \{s \mid \forall \rho \in X \mapsto Z : s(\rho) \in S(\{\rho\})\}$

Intuition:
- $\gamma_6(\alpha_6(\{s\}))$
  - $= \{s' \mid \forall \rho' \in X \mapsto Z : s'(\rho') \in \{s''(\rho'') \mid s'' \in \{s\} \land \rho'' \in \{\rho'\}\}\}$
  - $= \{s' \mid \forall \rho' \in X \mapsto Z : s'(\rho') \in \{s(\rho')\}\}$
  - $= \{s' \mid \forall \rho' \in X \mapsto Z : s'(\rho') = s(\rho')\}$
  - $= \{s' \mid s' = s\}$
  - $= \{s\}$

Abstraction in abstract interpretation.

© P. Cousot, Nov 16, 1999
• $\alpha_7 \in \wp((X \mapsto Z) \mapsto Z) \longmapsto ((X \mapsto \wp(S)) \mapsto \wp(S)) \quad (9)$

\[ \alpha_7 \doteq \alpha_5 \circ \alpha_6 \]

\[ \gamma_7 \in ((X \mapsto \wp(S)) \mapsto \wp(S)) \mapsto \wp((X \mapsto Z) \mapsto Z) \]

\[ \gamma_7 \doteq \gamma_6 \circ \gamma_5 \]

Intuition:

\[ \wp((X \mapsto Z) \mapsto Z) \]

\[ \alpha_6 \quad \gamma_6 \]

\[ \wp(X \mapsto Z) \mapsto \wp(Z) \]

\[ \alpha_5 \quad \gamma_5 \]

\[ (X \mapsto \wp(S)) \mapsto \wp(S) \]

---

**Calculational design of the abstract semantics**

\[ \llbracket e \rrbracket = \alpha_7(\llbracket e \rrbracket) \]

\[ = \alpha_5 \circ \alpha_6(\llbracket e \rrbracket) \quad \text{by def. (9) of } \alpha_7 \]

\[ = \alpha_1 \circ \alpha_6(\llbracket e \rrbracket) \circ \gamma_4 \quad \text{by def. (7) of } \alpha_5 \]

\[ = \lambda R \cdot \alpha_1(\{ s(\rho) \mid s \in \llbracket e \rrbracket \land \rho \in \gamma_4(R) \}) \quad \text{by def. (8) of } \alpha_5 \]

\[ = \lambda R \cdot \alpha_1(\{ \llbracket e \rrbracket \rho \mid \rho \in \gamma_4(R) \}) \quad \text{by def. (1) of } \llbracket e \rrbracket \]

\[ = \lambda R \cdot \alpha_1(\{ \llbracket e \rrbracket \rho \mid \rho \in \gamma_2 \circ \gamma_3(R) \}) \quad \text{by def. (6) of } \alpha_5 \]

\[ = \lambda R \cdot \alpha_1(\{ \llbracket e \rrbracket \rho \mid \rho \in \gamma_2(\lambda Y \cdot \gamma_1(R(Y))) \}) \quad \text{by def. (5) of } \gamma_3 \]

\[ = \lambda R \cdot \alpha_1(\{ \llbracket e \rrbracket \rho \mid \rho \in \{ \rho \mid \forall Y \in \wp(X) : \rho(X) \in \gamma_1(R(Y)) \} \}) \quad \text{by def. (4) of } \gamma_2 \]

We go on by structural induction on $e$. 

---

**Specification of the rule of signs abstract semantics**

- **Standard semantics:** $\llbracket e \rrbracket$
- **Collecting semantics:** $\{ e \} \doteq \llbracket e \rrbracket$
- **Abstract semantics:** $\llbracket e \rrbracket \doteq \alpha_7(\{ e \})$ best approximation
  $\llbracket e \rrbracket \supseteq \alpha_7(\{ e \})$ suboptimal approximation

**Example of suboptimal abstract semantics:**

- $\alpha_7(\llbracket X - X \rrbracket)(X : \{+1\})$
  $= \alpha_7(\llbracket 0 \rrbracket)(X : \{+1\})$
  $= \{0\}$

- $\llbracket X - X \rrbracket(X : \{+1\})$
  $= \llbracket X \rrbracket(X : \{+1\}) - \llbracket X \rrbracket(X : \{+1\})$
  $= \{+1\} - \{+1\}$
  $= \{-1,0,+1\}$

- $e \equiv n$:
  $\lambda R \cdot \alpha_1(\{ \llbracket n \rrbracket \rho \mid \forall Y \in \wp(Y) \in \gamma_1(R(Y)) \})$
  $= \lambda R \cdot \alpha_1(\{ n \})$ by def. $\llbracket n \rrbracket \rho \doteq n$
  $= \lambda R \cdot \{ \alpha_0(n) \}$ by def. (3) of $\alpha_1$

- $\lambda R \cdot \{ -1 \}$ if $n < 0$
- $\lambda R \cdot \{ 0 \}$ if $n = 0$
- $\lambda R \cdot \{ +1 \}$ if $n > 0$

- $e \equiv X$:
  $\lambda R \cdot \alpha_1(\{ \llbracket X \rrbracket \rho \mid \forall Y \in \wp(Y) \in \gamma_1(R(Y)) \})$
  $= \lambda R \cdot \alpha_1(\{ \rho(X) \mid \forall Y \in \wp(Y) \in \gamma_1(R(Y)) \})$ by def. $\llbracket X \rrbracket \rho \doteq \rho(X)$

- $\lambda R \cdot \alpha_1(\{ \rho(X) \mid \rho(X) \in \gamma_1(R(X)) \})$
  $= \lambda R \cdot \alpha_1(\gamma_1(R(X)))$
  $= \lambda R \cdot R(X)$
• \( e \equiv e_1 + e_2 \):

\[
\lambda R \cdot \alpha_1(\{e_1 + e_2\} R | \rho \in \gamma_4(R)) \\
= \lambda R \cdot \alpha_1(\{e_1\} R + [e_2] R | \rho \in \gamma_4(R))
\]

by def. \([e_1 + e_2] R \triangleq [e_1] R + [e_2] R\)

\( \subseteq \lambda R \cdot \alpha_1(\{e_1\} R + [e_2] R | \rho \in \gamma_4(R) \land \rho' \in \gamma_4(R)) \)

\( = \lambda R \cdot \alpha_1(\{x + y \mid x \in \{e_1\} R | \rho \in \gamma_4(R) \land y \in \{e_2\} R | \rho' \in \gamma_4(R)) \land \rho\}

\( \subseteq \lambda R \cdot \alpha_1(\{x + y \mid x \in \gamma_1 \circ \alpha_1(\{e_1\} R | \rho \in \gamma_4(R)) \land y \in \gamma_1 \circ \alpha_1(\{e_2\} R | \rho' \in \gamma_4(R)) \land \rho\}

\( \subseteq \lambda R \cdot \alpha_1(\{x + y \mid x \in \gamma_1(\{e_1\} R) \land y \in \gamma_1(\{e_2\} R)) \) by ind. hyp.

\( = \lambda R \cdot (\{e_1\} R + \{e_2\} R) \) where + is calculated by cases:

\[\ldots\]

---

**Summary of the abstract semantics**

- \( \langle n \rangle R = \{-1\} \) if \( n < 0 \)
- \( \{0\} \) if \( n = 0 \)
- \( +\{1\} \) if \( n > 0 \)

- \( \langle X \rangle R = R(x) \)

- \( \langle e_1 + e_2 \rangle R = \langle e_1 \rangle R + \langle e_2 \rangle R \)

where the “rule of signs” for addition + is:

<table>
<thead>
<tr>
<th>+</th>
<th>0</th>
<th>-1</th>
<th>0</th>
<th>+1</th>
<th>-1</th>
<th>+1</th>
<th>-1</th>
<th>+1</th>
<th>-1</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>-1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>-1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>-1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>-1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

---

**2. Formalization of abstraction/concretization**

---

**Reference**

Galois connection — 1

The pair \( \langle \alpha, \gamma \rangle \) is a Galois connection:

- \( \alpha \) is \( \subseteq \)-monotone
  \[ \Rightarrow \text{abstraction preserve implication;} \]
- \( \gamma \) is \( \subseteq \)-monotone
  \[ \Rightarrow \text{concretization preserves implication;} \]
- \( \gamma \circ \alpha \) is \( \subseteq \)-extensive
  \[ \Rightarrow \text{an abstraction introduces a loss of information;} \]
- \( \alpha \circ \gamma \) is \( \subseteq \)-reductive
  \[ \Rightarrow \text{a concretization can only be more precise.} \]

Notation: \( \langle \wp((X \mapsto \mathbb{Z}) \mapsto \mathbb{Z}), \subseteq \rangle \xrightarrow{\gamma} \langle (X \mapsto \wp(S)) \mapsto \wp(S), \subseteq \rangle \)

Galois connection — 2

An equivalent definition of
\[ \langle L, \leq \rangle \leftarrowarrow \langle M, \sqsubseteq \rangle \]
is

- \( \langle L, \leq \rangle \) and \( \langle M, \sqsubseteq \rangle \) are posets;
- \( \forall x \in L : \forall y \in M : \alpha(x) \sqsubseteq y \Leftrightarrow x \leq \gamma(y) \).

Example of Galois connection based abstraction:

\[ \begin{array}{c}
\{\mathbf{-1},\mathbf{0},\mathbf{1}\} \\
\{\mathbf{-1},\mathbf{0}\} \\
\{\mathbf{-1}\} \\
\varnothing \\
\end{array} \]

A few properties of Galois connections

- One function uniquely determine the other:
  \[ \alpha(x) = \bigcap \{y \mid x \leq \gamma(y)\} \]
  \[ \gamma(y) = \bigcup \{x \mid \alpha(x) \subseteq y\} \]
- \( \alpha \) has an adjoint iff it preserves existing lubs;
- \( \gamma \) has an adjoint iff it preserves existing glbs;
Surjections/injections

- $\alpha$ is surjective $\iff$ $\gamma$ is injective $\iff \alpha \circ \gamma = 1$:

$$\langle L, \leq \rangle \xrightarrow{\gamma} \langle M, \sqsubseteq \rangle \xleftarrow{\alpha}$$

- Assuming $\alpha$ surjective simplifies the formal presentation (not always possible in practice);
- Dually, $\alpha$ is injective $\iff$ $\gamma$ is surjective $\iff \gamma \circ \alpha = 1$:

$$\langle L, \leq \rangle \xleftarrow{\alpha} \langle M, \sqsubseteq \rangle \xrightarrow{\gamma}$$

Closure operators

If

$$\langle L, \leq \rangle \xrightarrow{\gamma} \langle M, \sqsubseteq \rangle$$

- $\gamma \circ \alpha$ represents the approximation of concrete properties by a concrete representation of the abstract properties;
- $\gamma \circ \alpha$ is an upper closure operator:
  - monotone,
  - extensive,
  - idempotent.
- A formally equivalent formalization of abstraction [6, Sections 6.2].

Moore family

If

$$\langle L, \leq \rangle \xrightarrow{\gamma} \langle M, \sqsubseteq \rangle$$

- $\gamma \circ \alpha(M)$ represents the set of concrete representations of the abstract properties;
- $\gamma \circ \alpha(M)$ is a Moore family:
  - contains a top element (if $M$ has a supremum),
  - closed by arbitrary intersections.
- A formally equivalent formalization of abstraction [7, Sections 6.1].

Reference


• Example of Moore family based abstraction:

![Diagram: Moore family abstraction]

- Example of Moore family based abstraction:
  - \{1\}
  - \{0\}
  - \{-1\}
  - \{-1,0\}
  - \{-1,1\}
  - \{-1,0,1\}

Abstraction in abstract interpretation, © P. Cousot, Nov 16, 1999

Best approximation

- \(\gamma \circ \alpha(P)\) is the concrete representation of the best abstract approximation of \(P\):
  - Its an upper-approximation: \(P \leq \gamma \circ \alpha(P)\);
  - Its the best abstraction: if is another abstract approximation (i.e. \(Q \in \gamma \circ \alpha(L)\) and \(P \leq Q\)) then \(\gamma \circ \alpha(P)\) is more precise (in that \(\gamma \circ \alpha(P) \leq Q\)).
- The best approximation does exists and is unique (by antisymmetry).

In absence of best approximation?

The classical rule of signs has no \(\{0\}\):

![Diagram: In absence of best approximation]

In absence of best approximation (continued)

- The best choice must be determined during this analysis:
  - \(\alpha(\{0\}) = \{-1,0\}\) is a better choice in \(\{0 + -1\}\) (i.e. \(\{-1,0\}\) is a better choice in \(\{0 + 1\}\).
- In practice, one uses \(\gamma\) only and widening/narrowing operators [8] as e.g. in [9];
- Other alternatives (e.g. use a soundness relation) discussed in [10].

References

Compound abstraction

• The abstraction is designed by composition:
  - of primitive abstractions;
  - of abstraction composition operators.

In French we would use the term “compositional”, which in the context of denotational semantics is already used to mean “by structural induction on the abstract syntax”.

Abstraction in abstract interpretation,

---

Example of primitive abstractions

• Elementwise abstraction: if \( \Subset \in S \mapsto S^\Subset \) then:

\[
\langle \wp(S), \subseteq \rangle \xlongleftarrow{\alpha_{\Subset}} \langle \wp(S^\Subset), \subseteq \rangle
\]

where:

\[
\begin{align*}
\alpha_{\Subset}(X) &\doteq \{ \Subset(x) \mid x \in X \} \\
\gamma_{\Subset}(Y) &\doteq \{ x \mid \Subset(x) \in Y \}
\end{align*}
\]

---

Examples of abstraction composition operators

• Abstraction composition: if \( \langle L, \leq \rangle \xrightarrow{\alpha_1} \langle M, \subseteq \rangle \) and \( \langle M, \subseteq \rangle \xrightarrow{\alpha_2} \langle N, \leq \rangle \) then:

\[
\langle L, \leq \rangle \xrightarrow{\gamma_1 \circ \alpha_1} \langle N, \leq \rangle
\]

• Functional abstraction: if \( \langle L, \leq \rangle \xrightarrow{\alpha_1} \langle L^\gamma, \leq \rangle \) and \( \langle M, \subseteq \rangle \xrightarrow{\alpha_2} \langle M^\circ, \subseteq \rangle \) then:

\[
\langle L \xmapsto{\text{mon}} M, \subseteq \rangle \xrightarrow{\lambda f \cdot \alpha_2 \circ \gamma_1} \langle L^\gamma \xmapsto{\text{mon}} M^\circ, \subseteq \rangle
\]

---

Fixpoint transfer and approximation

[11, p. 309]: If

- \( \langle L, \leq, 0, \lor \rangle \) is a cpo,
- \( \langle L, \leq \rangle \xrightarrow{\alpha} \langle M, \subseteq \rangle \),
- \( F \in L \xrightarrow{\min} L \),
- \( G \in M \xrightarrow{\min} M \),
- \( \alpha \circ F = \subseteq G \circ \alpha \) local completeness/approximation

then

\[
\langle M, \subseteq, \bot, \sqcup \rangle \text{ is a cpo where } \bot = \alpha(0) \text{ and } \sqcup X = \alpha(\lor \gamma(X)) \),
\[
\alpha \circ F \circ \gamma = \subseteq G \,.
\]

\[
\alpha(\text{lfp} F) \subseteq \subseteq \text{lfp} \subseteq G.
\]

---

Reference

The lattice of abstract interpretations

- The abstract interpretations of a semantics are isomorphic to closure operators;
- So the complete lattice of abstract interpretations \([12]\) is isomorphic to the complete lattice of closure operators on a complete lattice/cpo.

3. Abstraction/concretization in program analysis

Soundness and completeness

- \(\llbracket P \rrbracket\): collecting semantics of \(P\)
- \(\langle L, \leq \rangle \xrightarrow{\gamma} (M, \sqsubseteq)\): abstraction
- \(\llbracket P \rrbracket\): abstract semantics of \(P\)
- Soundness:
  \[ \forall P : \alpha(\llbracket P \rrbracket) \sqsubseteq \llbracket P \rrbracket \]
- (Global) completeness:
  \[ \forall P : \alpha(\llbracket P \rrbracket) = \llbracket P \rrbracket \]

Data flow analysis is an abstract interpretation

From [13, section 7.2.0.6.3]:

- \(\tau = (\mathcal{S}, \mathcal{A}, t)\): transition system (\(\mathcal{S} \triangleq C \times M\) control \times memory states, actions \(\mathcal{A}\) are assignments and tests)
- \(\mathcal{T}\): traces \(\langle s_0, a_0, s_1 \rangle \langle s_1, a_1, s_2 \rangle \ldots \langle s_{n-1}, a_{n-1}, s_n \rangle\), \(s_i \in \mathcal{S}\), \(a_i \in \mathcal{A}\)
- \(\mathcal{M}_\tau \subseteq \mathcal{T}\): program semantics (set of prefix closed finite traces generated by \(\tau\));
- \(\mathcal{E}\): set of expressions appearing in actions \(\mathcal{A}\);

References


Abstraction in abstract interpretation,

- **Static partitioning abstraction** [14]:
  \[ \alpha_p(M) \triangleq \prod_{\ell \in C} \{ \sigma'(s, a, \langle \ell, m \rangle) \mid \sigma'(s, a, \langle \ell, m \rangle) \in M \} \]
  such that:
  \[ \langle \varphi(T), \subseteq \rangle \xrightarrow{\gamma_p/\alpha_p} \langle C \mapsto \varphi(T), \subseteq \rangle \]

- **Pointwise abstraction**: If \( \langle \varphi(T), \subseteq \rangle \xrightarrow{\gamma} \langle L, \subseteq \rangle \) then:
  \[ \langle C \mapsto \varphi(T), \subseteq \rangle \xrightarrow{\gamma/\alpha} \langle C \mapsto L, \subseteq \rangle \]
  where:
  \[ \hat{\alpha}(M) \triangleq \lambda \ell \cdot \alpha(M(\ell)) \]

---

**Program static analysis is an abstract interpretation**

- **States** (finite control states \( \mathbb{C} \))
  \[ S \triangleq \mathbb{C} \times \mathbb{Z}^n \]
- **Properties**
  \[ P \triangleq \varnothing(S) \]
- **Interval abstract domains**
  \[ I \triangleq \{ [a, b] \mid a \leq b \} \cup \{ \bot \} \]
- **Interval abstraction**
  \[ \langle \varphi(Z), \subseteq \rangle \xrightarrow{\gamma_1/\alpha_1} \langle I, \subseteq \rangle \]

\[ \alpha_1(Z) \triangleq (Z = \emptyset \lor \bot : [\min Z, \max Z]) \quad (\min Z = -\infty, \max Z = \infty) \]

- **State abstraction**
  \[ \langle P, \subseteq \rangle \xrightarrow{\gamma_2/\alpha_2} \langle C \mapsto ([1, n] \mapsto I), \subseteq \rangle \]

\[ \alpha_2(S) \triangleq \prod_{\ell \in C} \prod_{i=1}^n \alpha_1(z \mid \exists x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n : (\ell, \langle x_1, \ldots, x_{i-1}, z, x_{i+1}, \ldots, x_n \rangle) \in S) \]

---

\[ \cup \]

**References**


\[ \top \]
Grammar analysis is an abstract interpretation

- $G = \langle N, T, P, A \rangle$ context free grammar
- The semantics $\llbracket G \rrbracket$ of $G$ is the terminal language generated by $G$;
- The \textbf{FIRST} algorithm is an abstract interpretation of the grammar fixpoint semantics \cite{18} by:
  \[ \langle T^*, \subseteq \rangle \xrightarrow{\gamma} \langle \varphi(T \cup \{e^1\}), \subseteq \rangle \]
  where:
  \[ \alpha(L) \triangleq \{ \varnothing(\sigma) \mid \sigma \in L \} \]
  \[ \emptyset(\epsilon) \triangleq \epsilon \]
  \[ \emptyset(x\sigma) \triangleq x \]

\textbf{Reference}

* $e$ is the empty string.

\textit{Type inference is an abstract interpretation}

- No best approximation ($f \in \mathbb{F}^3$, $a, b, c \in \mathbb{F}^1$, $n \in \mathbb{F}^0$):
  \[ X_0 \triangleq \emptyset \]
  \[ X_1 \triangleq \{ f(a(n), b(n), c(n)) \} \]
  \[ X_2 \triangleq \{ f(a(n), b(n), c(n)), f(a^2(n), b^2(n), c^2(n)) \} \]
  \[ \cdots \]
  \[ X_k \triangleq \{ f(a(n), b(n), c(n)), \ldots, f(a^k(n), b^k(n), c^k(n)) \} \]
  \[ \cdots \]
  \[ X_\omega \triangleq \bigcup_{k \geq 0} X_k = \{ f(a^k(n), b^k(n), c^k(n)) \mid k \geq 0 \} \]
  is not context free

- The $X_k$, $k \geq 0$ can all be described by a regular tree grammar but not $X_\omega$. So $X_\omega$ is approximated by the regular language:
  \[ \{ f(a^k(n), b^\ell(n), c^m(n)) \mid k, \ell, m \geq 0 \}. \]

\textbf{Reference}

\textit{Set based analysis is an abstract interpretation}

- Concrete domain: set $T$ of tree on a finite signature $\mathbb{F} = \bigcup_{n \geq 0} \mathbb{F}^n$;
- Abstract domain: regular tree grammars $G$ in Greibach normal form:
  \[ \{ X := f^n(Y_1, \ldots, Y_n) \mid f^n \in \mathbb{F}^n \} \]
  \[ \{ X := f^0 \mid f^0 \in \mathbb{F}^0 \} \]
  where non-terminals $X, \ldots$ correspond to program elements (variables, etc...)
- Concretization: $\gamma(G)$ is the set of finite trees generated by the grammar $G$;

\textbf{Reference}
Concrete collecting domain:
\[ P \triangleq \varphi(S) \]

Abstract domain:
\[ m \in M \quad m ::= \text{int} \mid m_1 > m_2 \]
Church/Curry/Hindley monotype
\[ H \in H \triangleq X \mapsto M \]
type environments
\[ \theta \in I \triangleq H \times M \]
типings
\[ T \in T \triangleq \varphi(I) \]
program types

In general a \( \lambda \)-expression has (infinitely) many types:
\[ \lambda x. x \text{ has types } \{ \langle H, m > m \rangle \mid H \in H \land m \in M \} \]

Galois connection:
\[ \langle P, \subseteq \rangle \xrightarrow{\gamma} \langle T, \supseteq \rangle \]

Typable programs cannot go wrong: if a \( \lambda \)-expression \( e \) has type \( T \) and \( T \neq \emptyset \) then \( \forall R \in \mathbb{R} : [e] R \neq \omega \) (since \( [e] \in \gamma(T) \));

Model checking is an abstract interpretation

\( \langle S, t \rangle \): transition system \( t \subseteq S^2 \);
\( \Sigma_S \): set of traces on states \( S \);
\( M_t \subseteq \Sigma_S \): model (set of maximal traces) generated by \( t \);
\( X_s \triangleq \{ s \sigma \mid s \sigma \in X \} \) subset of traces of \( X \subseteq \Sigma_S \) starting with state \( s \);
\( \varphi \in L \): formulae of temporal logic \( L \);
\( [\varphi] \subseteq \Sigma_S \): semantics of the closed temporal formula (set of maximal traces);

Reference

• Boolean universal model checking is $\alpha_t^\forall(\llbracket \varphi \rrbracket)$

\[
\langle \varphi(\Sigma), \supseteq \rangle \xrightarrow{\gamma_t^\forall} \langle \varphi(S), \supseteq \rangle
\]

$\alpha_t^\forall(\Phi) \doteq \{ s \in S \mid \mathcal{M}_{t,s} \subseteq \Phi \}$

• Boolean existential model checking is dual $(\alpha_t^\exists(\Phi) = \neg \alpha_t^\forall(\neg \Phi))$ so:

\[
\langle \varphi(\Sigma), \subseteq \rangle \xrightarrow{\gamma_t^\exists} \langle \varphi(S), \subseteq \rangle
\]

$\alpha_t^\exists(\Phi) \doteq \{ s \in S \mid (\mathcal{M}_{t,s} \cap \Phi) \neq \emptyset \}$

4. Abstraction/concretization in semantics

Abstract model checking is the composition of abstract interpretations

• State abstraction:

\[
\langle \varphi(S), \subseteq \rangle \xrightarrow{\gamma_S} \langle \varphi(S^\sharp), \subseteq \rangle
\]

• Trace-based model abstraction:

\[
\langle \varphi(\Sigma), \subseteq \rangle \xrightarrow{\gamma_m} \langle \varphi(\Sigma^\sharp), \subseteq \rangle
\]

• Abstract model checking:

$\alpha_t^\forall \doteq \alpha_S \circ \alpha_t^\forall \circ \gamma_m$

Semantics are abstract interpretations

• The various semantics of programming languages can be understood as abstract interpretations of a maximal trace semantics;

• (and the fixpoint characterizations of these semantics can all be constructively derived from the maximal trace semantics generated by a transition system (i.e. small-step operational semantics), see [22]).

Reference

Maximal trace semantics

- The basic maximal trace semantics is:

\[ \alpha \text{ is a Galois connection:} \]

\[ \begin{align*}
\text{blocking state} & : & T^\ddagger &= \{ \bullet \rightarrow \ldots \rightarrow \bullet \} & \leftrightarrow \text{finite traces} \\
\text{infinite traces} & : & \bigcup \{ \bullet \rightarrow \ldots \rightarrow \bullet \} & \leftrightarrow \text{infinite traces}
\end{align*} \]

\[ T^\ddagger \in \wp(T), \ T \text{ is the set of traces over states } S. \]

Natural, demoniac & angelic semantics

- Natural trace semantics: \( T^\ddagger; \)

- Angelic abstraction \(^6\):

\[ \alpha(T^\ddagger) = \{ \bullet \rightarrow \ldots \rightarrow \bullet \in T^\ddagger \}; \]

- Demoniac abstraction \(^6\):

\[ \alpha(T^\ddagger) = T^\ddagger \bigcup \{ \bullet \rightarrow \ldots \rightarrow \bullet \in T^\ddagger \}. \]

The \( \alpha \)'s are Galois connections.

Transition semantics

- Transition semantics:

\[ \alpha(T^\ddagger) = \{ \langle a, b \rangle | \bullet \rightarrow \ldots \rightarrow \bullet \in T^\ddagger \} \bigcup \{ \langle a', b' \rangle | \bullet \rightarrow \ldots \rightarrow \bullet \in T^\ddagger \} \]

\[ \alpha \text{ is a Galois connection:} \  \wp(T), \subseteq \xrightarrow{\alpha} (\wp(S \times S), \subseteq) \]

Relational semantics

\[ \alpha \in \wp(T) \longmapsto \wp(S \times S), \quad S_\bot = S \cup \{ \bot \} \]

\[ R = \alpha(T) \]

\[ = \{ \langle a, b \rangle | \bullet \rightarrow \ldots \rightarrow \bullet \in T \} \bigcup \{ \langle a, \bot \rangle | \bullet \rightarrow \ldots \rightarrow \bullet \in T \} \]

\[ \alpha \text{ is a Galois connection.} \]

\(^6\) Eliminate all infinite traces.

\(^6\) Introduce all arbitrary finite traces for states possibly starting an infinite trace.
Non-deterministic denotational semantics

\[ \alpha \in \wp(\mathbb{S} \times \mathbb{S}_\perp) \mapsto (\mathbb{S} \mapsto \wp(\mathbb{S}_\perp)) \]

\[ \mathcal{D} = \alpha(\mathcal{R}) \]
\[ = \lambda s \cdot \{ s' \in \mathbb{S}_\perp \mid \langle s, s' \rangle \in \mathcal{R} \} \]

\[ \alpha \text{ is a Galois isomorphism.} \]

Axiomatic semantics

\[ \alpha \in (\wp(\mathbb{S}) \mapsto \wp(\mathbb{S}_\perp)) \mapsto \wp(\wp(\mathbb{S} \times \wp(\mathbb{S}_\perp))) \]

\[ \mathcal{H} = \alpha(\mathcal{W}) \]
\[ = \{ (P, Q) \mid P \subseteq \mathcal{W}(Q) \} \]

\[ \alpha \text{ is a Galois injection.} \]

Predicate transformer semantics

\[ \alpha \in (\mathbb{S} \mapsto \wp(\mathbb{S}_\perp)) \mapsto (\wp(\mathbb{S}_\perp) \mapsto \wp(\mathbb{S})) \]

\[ \mathcal{W} = \alpha(\mathcal{D}) \]
\[ = \lambda Q \cdot \{ s \in \mathbb{S} \mid \forall s' \in \mathbb{S}_\perp : s' \in \mathcal{D}(s) \Rightarrow s' \in Q \} \]

\[ \alpha \text{ is a Galois injection.} \]

The hierarchy of semantics

Abstraction in abstract interpretation, © P. Cousot, Nov 16, 1999
5. Conclusion

What is abstract interpretation?

- Semantics as well as program analysis algorithms approximate the incomputable collection of all possible behaviors of programs on computers [24];
- Abstract interpretation is a theory of abstraction understood as a discrete approximation of computer system behavior specifications [23];

Reference


On the coincidence of abstraction/refinement in program verification and abstraction/concretization in abstract interpretation (tentative)

- Abstraction/refinement in program verification:
  \[ \alpha: \text{concrete object} \rightarrow \text{abstract object} \]
- Abstraction/concretization in abstract interpretation:
  \[ \alpha: \mathcal{P}(\text{concrete object}) \rightarrow \mathcal{P}(\text{abstract object}) \]
- Coincidence:
  - Lift the reasoning on objects to reasoning on object properties (e.g. using predicate transformers) ???
  - Use category theory (various attempts that we made did not bring any new practical idea) ???

Abstraction in abstract interpretation,  © P. Cousot, Nov 16, 1999

• What I know about refinement \(^1\):
  - \( c: C \rightarrow C \) concrete operation \( a: A \rightarrow A \) abstract operation
  - \( \alpha: C \rightarrow A \) abstraction
  - \( \alpha \circ c = a \circ \alpha \) refinement condition (10)
- Lifting to sets:
  - \( \alpha^\sharp (X) \overset{\gamma^\sharp}{\rightarrow} \{ \alpha(x) \mid x \in X \} \)
  - \( c^\sharp (X) \overset{\gamma^\sharp}{\rightarrow} \{ c(x) \mid x \in X \} \)
  - \( a^\sharp (Y) \overset{\gamma^\sharp}{\rightarrow} \{ a(y) \mid x \in Y \} \)
  so that (10) implies: \( \alpha^\sharp \circ c^\sharp = a^\sharp \circ \alpha^\sharp \). If \( \alpha \) is surjective then \( a^\sharp = \alpha^\sharp \circ c^\sharp \circ \gamma^\sharp \) (i.e. \( a^\sharp \) is the abstract interpretation of \( c^\sharp \)).
- So (?) reasoning on objects in program development by refinement is “equivalent to” reasoning on their local properties (i.e. topos theory is relevant)?

\(^1\) nothing, so to what I guess about refinement!
Contents

Abstraction in abstract interpretation ........................................ 1
Abstract ................................................................................. 2

1. Introductive example ............................................................ 3
   Abstract interpretation .......................................................... 4
   Abstraction in abstract interpretation ........................................ 5
   Objects and their properties ....................................................... 6
   Properties as sets .................................................................. 7
   Example: rule of signs standard semantics ................................. 8
   Example: rule of signs — 1 — the abstract object point of view .... 10
   Example: rule of signs — 2 — the abstract property point of view .. 12
   Approximation ...................................................................... 13
   Approximation for the rule of signs .......................................... 14
   Specification of the rule of signs abstract semantics .................... 22
   Calculational design of the abstract semantics ........................... 23

Abstraction in abstract interpretation, © P. Cousot, Nov 16, 1999

2. Formalization of abstraction/concretization .................................... 28
   Galois connection — 1 ............................................................ 29
   Galois connection — 2 ............................................................ 31
   A few properties of Galois connections ...................................... 32
   Surjections/injections ............................................................. 33
   Closure operators .................................................................. 34
   Moore family ........................................................................ 36
   Best approximation .................................................................. 38
   In absence of best approximation? ............................................ 39
   In absence of best approximation (continued) ........................... 40
   Compound abstraction ............................................................. 41
   Example of primitive abstractions .............................................. 42
   Examples of abstraction composition operators .......................... 43
   Fixpoint transfer and approximation ......................................... 44
   The lattice of abstract interpretations ......................................... 45

Abstraction in abstract interpretation, © P. Cousot, Nov 16, 1999

3. Abstraction/concretization in program analysis .............................. 47
   Data flow analysis is an abstract interpretation .......................... 48
   Program static analysis is an abstract interpretation .................... 51
   Grammar analysis is an abstract interpretation ........................... 53
   Set based analysis is an abstract interpretation ........................... 54
   Type inference is an abstract interpretation ................................ 56
   Model checking is an abstract interpretation ............................... 60
   Abstract model checking is the composition of abstract interpretations 62

Abstraction in abstract interpretation, © P. Cousot, Nov 16, 1999

4. Abstraction/concretization in semantics ........................................ 63
   Semantics are abstract interpretations ...................................... 64
   Maximal trace semantics ........................................................ 65
   Transition semantics ................................................................ 66
   Natural, demoniac & angelic semantics ..................................... 67
   Relational semantics ................................................................ 68
   Non-deterministic denotational semantics ................................... 69

Abstraction in abstract interpretation, © P. Cousot, Nov 16, 1999

Summary of the abstract semantics ........................................... 27

Soundness and completeness ..................................................... 46

Abstraction in abstract interpretation, © P. Cousot, Nov 16, 1999

Abstraction in abstract interpretation, © P. Cousot, Nov 16, 1999

THE END

Abstraction in abstract interpretation, © P. Cousot, Nov 16, 1999
Predicate transformer semantics ........................................... 70
Axiomatic semantics ......................................................... 71
The hierarchy of semantics .................................................. 72

5. Conclusion ................................................................... 73
   What is abstract interpretation? ........................................... 74

Electronic bibliography