A casual introduction to Abstract Interpretation

NYU, 29–30 March 2012

Patrick Cousot

cs.nyu.edu/~pcousot
di.ens.fr/~cousot

Examples of Abstractions

Abstractions of Dora Maar by Picasso

Pixelation of a photo by Jay Maisel
Abstractions of a man / crowd

The concrete is not always well-known!

Numerical abstractions in Astrée

Collecting semantics:
- partial traces

Intervals:
- $x \in [a, b]$

Simple congruences:
- $x \equiv a[b]$

Octagons:
- $\pm x \pm y \leq a$

Ellipses:
- $x^2 + by^2 - axy \leq d$

Exponentials:
- $-a^{bt} \leq y(t) \leq a^{bt}$

A slightly more detailed example
Set of functions

How to approximate \{ f_1, f_2, f_3, f_4 \} ?

A less precise abstraction

Concrete questions on the \( f_i \)

\[ \exists i, t \in [l, h]: f_i(t) > M \]

\[ \exists i, t \in [l, h]: f_i(t) < m \]

Min/max questions on the \( f_i \)
Concrete questions answered in the abstract

\[ \exists i, t \in [l, h] : f_i(t) > M ? \quad \text{I don't know} \]

\[ \exists i, t \in [l, h] : f_i(t) < m ? \quad \text{No} \]

Min/max questions on the \( f_i \)

Soundness of the abstraction

• No concrete case is ever forgotten:

A more precise/refined abstraction

An even more precise/refined abstraction
Passing to the limit

Sound and *complete* abstraction for min/max questions on the $f_i$

\[ f(t) \]

\[ t \]

The hierarchy of abstractions

A non-comparable abstraction

Sound and *incomplete* abstraction for min/max questions on the $f_i$

Elements of Abstract Interpretation Theory Explained with ...
Elements of Abstract Interpretation Theory Explained with ... Flowers

The concrete world

A mini graphical language

- **Objects** $o \in O$
- **Operations** on objects $O^n \to O$, $n \geq 0$
- **Logical operations** on objects $O^n \to \text{Booleans}$, $n \geq 0$

Objects

- An object $o \in O$ is defined by
  - An origin (a reference point $\times$)
  - A set of (infinitely small) black pixels (on a white background)

Example I of object:
An example II of object: a flower

Constant objects

- A petal is an example of constant object

Logical operations on objects

- Inclusion $\subseteq$
- Examples:

Operation on objects: rotation

- rotation $r[a](o)$
  rotates the object $o$ clockwise by angle $a$ degrees around its origin
Example I of rotation

\[ \text{petal} = r[45](\text{petal}) = \]

Example II of rotation

\[ \text{flower} = r[-45](\text{flower}) = \]

Operation on objects: add a stem

- Add a stem
  \[ \text{stem}(o) \]
  adds a stem to object \( o \) (up to the origin of object \( o \), with new origin at the root of the stem)

\[ o = \text{stem}(o) = \]

Operation on objects: union

- The union \( o_1 U o_2 \) of objects \( o_1 \) and \( o_2 \) is the superposition of the pixels of \( o_1 \) and \( o_2 \) at their origins

- Example:
  \[ o_1 = \]
  \[ o_2 = \]
  \[ o_1 U o_2 = \]
Example: corolla

- corolla = petal \cup r[45](petal) \cup r[90](petal) \cup r[135](petal) \cup r[180](petal) \cup r[225](petal) \cup r[270](petal) \cup r[315](petal)

Building a corolla iteratively

- corolla = flower
- flower = stem(corolla)

Corolla transformer

- F(X) = r[45]X \cup petal
- Example:
  - X =
    - r[45]X =
    - r[45]X \cup petal =
Iterates of a transformer to a fixpoint

- The iterates $F^n$, $n \geq 0$, of $F$ from the empty set $\emptyset$ are:
  
  $F^0 = \emptyset$
  
  $F^1 = F(F^0)$
  
  $F^2 = F(F^1)$
  
  $\vdots$
  
  $F^{n+1} = F(F^n)$
  
  $\vdots$
  
  $F^\omega = \bigcup_{n \geq 0} F^n = \text{lfp } F$

- Least fixpoint: $F(\text{lfp } F) = \text{lfp } F$, and
  
  $F(x) = x$ implies $\text{lfp } F \subseteq x$

---

Concrete bouquet

- bouquet = $r[-45](\text{flower}) \cup \text{flower} \cup r[45](\text{flower})$

---

Fixpoint corolla

- $F(X) = r[45]X \cup \text{petal}$

- corolla = $\text{lfp } F = \text{petal}$

- Proof: the iterates are:
  
  $F^0$
  
  $F^1$
  
  $F^2$
  
  $F^3$
  
  $F^4$
  
  $F^5$
  
  $F^6$
  
  $F^7$
  
  $F^8$

---

The abstract world
**Over-approximation**

- An over-approximation of an object \( o \) is an object \( \tilde{o} \) with
  - same origin
  - more pixels

- The dual is an under-approximation, with less pixels

---

**Abstraction**

- An abstraction of an object \( o \) is a mathematical/computer representation of an over-approximation of this object \( o \)

- The abstraction is sometimes exact else is a strict over-approximation

  - Examples abstraction by plain squares
    - ![exact abstraction](image)
    - ![strict over-approximation](image)

---

**Examples of abstractions of flowers**

- Encode a concrete over-approximation by its outline

  - ![concrete](image)
  - ![abstract](image)
  - ![more abstract](image)

---

**Examples of over-approximations of flowers**

- ![examples](image)
A Touch of Abstract Interpretation Theory

Abstract domain

- An abstract domain is
  - a set of abstract objects $\overline{O}$ (abstracting concrete objects)
  - a set of abstract operations (abstracting the concrete operations) $\overline{O}^n \rightarrow \overline{O}, n \geq 0$
  - a set of logical abstract operations $\overline{O}^n \rightarrow \text{Booleans}, n \geq 0$

Abstraction function

- The abstraction function $\alpha \in O \rightarrow \overline{O}$ maps concrete objects $o \in O$ to their approximation by an abstract object $\alpha(o) \in \overline{O}$

Example I of abstraction function by plain squares:

- Example II of abstraction function
Example III of abstraction function

Outlining brush of infinite diameter

The hierarchy of abstractions

• Larger brush diameter: more abstract
• Different brush shapes: may be non-comparable abstractions

Concretization function

• A concretization function $\gamma \in \overline{O} \rightarrow O$ maps an abstract object $\bar{o} \in \overline{O}$ to the concrete objects $\gamma(\bar{o}) \in O$ that is represents/approximates

• $\gamma(\bar{o})$ is the concrete meaning/semantics of $\bar{o}$

Example of concretization
Abstract logical operation: abstract inclusion

- The abstract flower inclusion is defined as

\[ \overline{o}_1 \subseteq \overline{o}_2 \text{ if and only if } \gamma(\overline{o}_1) \subseteq \gamma(\overline{o}_2) \]

- Example:

Galois connection 1/4

- \( \alpha \) is increasing

Galois connection 2/4

- \( \gamma \) is increasing

- Proof: by definition of \( \subseteq \), \( \overline{o}_1 \subseteq \overline{o}_2 \) implies \( \gamma(\overline{o}_1) \subseteq \gamma(\overline{o}_2) \)

Galois connection 3/4

- For all concrete objects \( x \in O \), \( x \subseteq \gamma \circ \alpha(x) \)

- Intuition: soundness (over-approximation)

The larger the concrete, the larger the abstract
Galois connection 4/4

- For all abstract objects \( y \in \bar{O} \), \( \alpha \circ \gamma(y) = y \)

- Intuition: \( \alpha \) returns the most precise abstraction

\[
\begin{align*}
\text{flower} & \quad \gamma(\text{flower}) & \quad \alpha(\gamma(\text{flower}))
\end{align*}
\]

Example of biological abstraction

- Let \( \text{Species} \) be the set of all chemical species \((\mathcal{C}, c_1, c_1', \ldots, c_k, c_k', \ldots \in \text{Species})\).
- Let \( \text{Local\_view} \) be the set of all local views
- Let \( \alpha \in \varphi(\text{Species}) \rightarrow \varphi(\text{Local\_view}) \) be the function that maps any set of complexes into the set of their local views.

\[
\alpha(\{R(Y1 \sim u, l!1), E(r!1)\}) = \{R(Y1 \sim u, l!r.E); E(r!l.R)\}
\]

- The function \( \alpha \) defines a Galois connexion:

\[
\varphi(\text{Species}) \xrightarrow{\gamma} \varphi(\text{Local\_view})
\]

- (The function \( \gamma \) maps a set of local views into the set of complexes that can be built with these local views).

Galois connection: all in one

- Notation:

\[
\begin{align*}
<0, \subseteq> & \quad \gamma & \quad \alpha & \quad <\bar{O}, \subseteq>
\end{align*}
\]

- Equivalent definition

\[
\forall o \in O, \bar{o} \in \bar{O}: \alpha(o) \subseteq \bar{o} \quad \text{iff} \quad o \subseteq \gamma(\bar{o})
\]

and

\[
\alpha \text{ surjective} \quad \text{(otherwise } \alpha \circ \gamma(y) \not\subseteq y)\]

Specification of abstract operations

- \( \text{cte} \triangleq \alpha(\text{cte}) \)
- \( \text{op}_1(x) \triangleq \alpha(\text{op}_1(\gamma(x))) \)
- \( \text{op}_2(x, y) \triangleq \alpha(\text{op}_2(\gamma(x), \gamma(y))) \)
- \( \ldots \)
- \( \text{op}_n(x_1, \ldots, x_n) \triangleq \alpha(\text{op}_n(\gamma(x_1), \ldots, \gamma(x_n))) \)

- Can be less precise

\[
\alpha(\text{op}_n(\gamma(x_1), \ldots, \gamma(x_n))) \subseteq \text{op}_n(x_1, \ldots, x_n)
\]
Abstract constants

- Abstract petal

\[ \alpha(\cdot) \triangleq \cdot \]

A commutation theorem on rotation

- \[ \alpha(r[a](y)) = r[a](\alpha(y)) \quad \forall y \in \bar{D} \]
- Proof:
  \[
  \begin{align*}
  \alpha(r[a](y)) &= \alpha(\gamma(\alpha(r[a](y)))) \\
  &= \alpha(\gamma(r[a](\alpha(y)))) \\
  &= \alpha(r[a](\gamma(\alpha(y)))) \\
  \triangleq r[a](\alpha(y)) & \quad \text{definition abstract rotation}
  \end{align*}
  \]

Abstract rotation

- Abstract rotation

\[
\bar{r}[a](\bar{\delta}) \triangleq \alpha(r[a](\gamma(\bar{\delta}))) \\
\begin{align*}
\alpha(r[a](\gamma(\bar{\delta}))) &= \alpha(\gamma(r[a](\bar{\delta}))) & \text{rotation preserves shape} \\
&= r[a](\bar{\delta}) & \text{identity}
\end{align*}
\]
- Example:

Abstract stems

- \[ \overline{\text{stem}}(y) \triangleq \alpha(\text{stem}(\gamma(y))) \]

- Example:
Abstract union

- \( x \cup y \triangleq \alpha(\gamma(x) \cup \gamma(y)) \)

- Join abstraction theorem:
  \[ \alpha(x) \cup \alpha(y) = \alpha(x \cup y) \]

Abstract bouquet (cont’d)

- bouquet = \( \overline{r}[-45](\text{flower}) \cup \text{flower} \cup \overline{r}[45](\text{flower}) \)

- \[ \alpha \left( \gamma \left( \begin{array}{c}
\overline{r}[-45](\text{flower}) \\
\text{flower}
\end{array} \right) \cup \gamma \left( \begin{array}{c}
\overline{r}[45](\text{flower})
\end{array} \right) \right) \]

- \( \alpha(\text{bouquet}) \)

Abstract bouquet (cont’d)

- \( \alpha(\overline{r}[-45](\text{flower}) \cup \text{flower} \cup \overline{r}[45](\text{flower})) \)

- rotation commutation theorem
  \[ \overline{r}[-45](\alpha(\text{flower})) \cup \alpha(\text{flower}) \cup \overline{r}[45](\alpha(\text{flower})) \]

- join abstraction theorem
  \[ \alpha(\overline{r}[-45](\text{flower}) \cup \alpha(\text{flower}) \cup \alpha(\overline{r}[45](\text{flower})) \]

- definition concrete bouquet
  \[ \alpha(\text{flower}) \]

A theorem on abstract bouquets

- bouquet
  \[ \overline{r}[-45](\text{flower}) \cup \text{flower} \cup \overline{r}[45](\text{flower}) \]

- rotation commutation theorem
  \[ \overline{r}[-45](\alpha(\text{flower})) \cup \alpha(\text{flower}) \cup \overline{r}[45](\alpha(\text{flower})) \]

- join abstraction theorem
  \[ \alpha(\overline{r}[-45](\text{flower}) \cup \alpha(\text{flower}) \cup \alpha(\overline{r}[45](\text{flower})) \]

- definition concrete bouquet
  \[ \alpha(\text{flower}) \]
Abstract corolla transformer

- Corolla transformer commutation theorem:

\[ \alpha(F(x)) = \alpha(\text{petal} \cup r[45](x)) \]

- Join abstraction theorem:

\[ \alpha(\text{petal}) \cup \alpha(r[45](x)) = \text{petal} \cup \alpha(r[45](x)) \]

- Definition abstract petal:

\[ \text{petal} \cup \alpha(r[45](x)) = \overline{\text{F}}(\alpha(x)) \]

by defining \( \overline{F}(y) = \text{petal} \cup \overline{r}[45](y) \)

Example of biological transformer

- Concrete rule:

- Abstract rule:

Abstract transformer

- An abstract transformer \( \overline{F} \) is

\[ \forall P \in \mathcal{P} : \alpha \circ F(P) \subseteq \overline{F} \circ \alpha(P) \]

- Sound \iff

- Complete \iff

\[ \forall P \in \mathcal{P} : \alpha \circ F(P) = \overline{F} \circ \alpha(P) \]

Fixpoint abstraction

- For an increasing and sound abstract transformer, we have a fixpoint approximation

\[ \alpha(\text{lfp} \leq F) \subseteq \text{lfp} \leq \overline{F} \]

- For an increasing, sound, and complete abstract transformer, we have an exact fixpoint abstraction

\[ \alpha(\text{lfp} \leq F) = \text{lfp} \leq \overline{F} \]
Abstract corolla

- corolla = α(corolla) = α(lfp F) = lfp F

since F(x) = petal ∪ r[45](x)
and F(y) = petal ∪ r[45](y)
do commute: α(F(x)) = F(α(x))

Iterates for the abstract corolla

Example of biological fixpoint

- Concrete reachability transformer:

\[
\begin{align*}
&\varphi(\text{Species}) \rightarrow \varphi(\text{Species}) \\
&F : \{ \begin{array}{ll} \\
X & \rightarrow X \cup \{ c' \mid \exists R_k \in \mathcal{R}, c_1, \ldots, c_m \in X, c_1, \ldots, c_m \rightarrow R_k c'_1, \ldots, c'_n \} \\
\end{array} \}
\end{align*}
\]

- Reachable species from Species₀

\[
lfp \lambda. X. \text{Species₀ } \cup F(X)
\]

- Abstract reachability transformer:

\[
\begin{align*}
&\varphi(\text{Local}_\text{view}) \rightarrow \varphi(\text{Local}_\text{view}) \\
&F^\dagger : \{ \begin{array}{ll} \\
X & \rightarrow X \cup \{ l' \mid \exists R_k \in \mathcal{R}, l_1, \ldots, l_m \in X, l_1, \ldots, l_m \rightarrow R_k l'_1, \ldots, l'_n \} \\
\end{array} \}
\end{align*}
\]

Convergence acceleration with widening

Infinite iteration

Accelerated iteration with widening
(e.g. with a widening based on the derivative as in Newton-Raphson method)
Abstraction of the graphical language

• Any graphical program can be abstracted by replacing the concrete objects/operations by abstract ones

• The soundness follows by induction on the syntax of programs

Applications of Abstract Interpretation in Computer Science

See Software Horror Stories (www.cs.tau.ac.il/~nachumd/horror.html)

Software

• Ait: static analysis of the worst-case execution time of control/command software (www.absint.com/ait/)

• Astrée: proof of absence of runtime errors in embedded synchronous real time control/command software (www.absint.com/astree/). AstreéA for asynchronous programs (www.astreea.ens.fr/)

• C Global Surveyor, NASA, static analyzer for flight software of NASA missions (www.cmu.edu/silicon-valley/faculty-staff/venet-arnaud.html)

• Checkmate: static analyzer of multi-threaded Java programs (www.pietro.ferrara.name/checkmate/)

• CodeContracts Static Checker, Microsoft (msdn.microsoft.com/en-us/devlabs/dd491992.aspx)

• Fluctuat: static analysis of the precision of numerical computations (www-list.cea.fr/labos/gb/LSL/fluctuat/index.html)

• Infer: static analyzer for C/C++ (monoidics.com/)

• Julia: static analyzer for Java and Android programs (www.juliasoft.com/juliasoft-android-java-verification.aspx?Id=201177234649)

• Predator: static analyzer of C dynamic data structures using separation logic (www.fit.vutbr.cz/research/groups/verifit/tools/predator/)

• Terminator: termination proof (www.cs.ucl.ac.uk/staff/p.ohearn/Invader/Invader/Invader_Home.html)

etc

Libraries:

• Apron numerical domains library (apron.cri.ensmp.fr/library/)

• Parma Polyhedral Library (bugseng.com/products/ppl/)

etc
Hardware

- (Generalized) symbolic trajectory evaluation (Intel)

Example of ternary simulation
If some inputs are undefined, the output often is too, but not always:

```
X X X
0 1 X
```

7-input AND gate

Example of quaternary simulation
It's theoretically convenient to generalize ternary to quaternary simulation, introducing an 'overconstrained' value $T$.

We can think of each quaternary value as standing for a set of possible values:

- $T = \emptyset$
- $0 = \{0\}$
- $1 = \{1\}$
- $X = \{0, 1\}$

This is essentially a simple case of an abstraction mapping, and we can think of the abstract values partially ordered by information.

Intel's Successes with Formal Methods

John Harrison
Intel Corporation
15 March 2012

Conclusion

If the simulation/analysis/checking of your model does not scale up, consider using (sound (and complete)) abstractions

System biology

- See SBFM'2012!