Abstract Interpretation: a methodology for the rapid development of provably correct static analyzers

Arnaud Venet
Kestrel Technology
NASA Ames Research Center

arnaud@email.arc.nasa.gov
Static analysis in real life

- Undecidable problem: automatic program verification $\Rightarrow$ loops
- Approximation for decidability: false positives
- Tradeoff precision/efficiency
- The approximation should be tunable:

```
Choosing an approximation → Analysis → Evaluating the results

Tuning the approximation
```
Abstract Interpretation

+ A general methodology for building static analyzers
+ Provides generic algorithms
+ Approximation and resolution are separated: the analyzers are tunable by construction
+ The soundness proof goes along with the analyzer design

- Scalability is difficult to achieve
Methodology

Concrete Semantics

Collecting Semantics

Partitioning

Abstract Semantics

Iterative Resolution Algorithms

Abstract Domain

Abstract Domain

Tuners
Methodology

Concrete Semantics

Collecting Semantics

Partitioning

Abstract Semantics

Iterative Resolution Algorithms

Tuners

Abstract Domain

Abstract Domain
Concrete semantics

Small-step operational semantics: $(\Sigma, \rightarrow)$

Example:

```
1:   n = 0;
2:   while n < 1000 do
3:     n = n + 1;
4:   end
5:   exit
```

$\langle 1, n \Rightarrow \Omega \rangle \rightarrow \langle 2, n \Rightarrow 0 \rangle \rightarrow \langle 3, n \Rightarrow 0 \rangle \rightarrow \langle 4, n \Rightarrow 1 \rangle$

$\rightarrow \langle 2, n \Rightarrow 1 \rangle \rightarrow \ldots \rightarrow \langle 5, n \Rightarrow 1000 \rangle$
Methodology

Abstract Semantics

Concrete Semantics

Collecting Semantics

Partitioning

Abstract Semantics

Abstract Domain

Abstract Domain

Iterative Resolution Algorithms

Tuners
Collecting semantics

The first abstraction step. It defines the observable behaviors of programs:

- Sets of states (e.g. range of variables)
- Sets of finite traces (e.g. computational dependencies)
- Sets of finite and infinite traces (e.g. termination properties)
State properties

The set of descendants of the initial state $s_0$:

$$S = \{ s \mid s_0 \rightarrow ... \rightarrow s \}$$

Theorem: $F : (\mathcal{P}(\Sigma), \subseteq) \rightarrow (\mathcal{P}(\Sigma), \subseteq)$

$$F(S) = \{ s_0 \} \cup \{ s' \mid \exists s \in S: s \rightarrow s' \}$$

$$S = \text{lfp} F$$
Example

\[ S = \{ \langle 1, n \Rightarrow \Omega \rangle, \langle 2, n \Rightarrow 0 \rangle, \langle 3, n \Rightarrow 0 \rangle, \langle 4, n \Rightarrow 1 \rangle, \langle 2, n \Rightarrow 1 \rangle, \ldots, \langle 5, n \Rightarrow 1000 \rangle \} \]

1: \quad n = 0;
2: \quad while n < 1000 do
3: \quad \quad n = n + 1;
4: \quad end
5: \quad exit
Methodology

Abstract Semantics

Concrete Semantics

Collecting Semantics

Partitioning

Abstract Semantics

Iterative Resolution Algorithms

Tuners

Abstract Domain

Abstract Domain
Partitioning

We partition the set $\Sigma$ of states w.r.t. program points:

- $\Sigma = \Sigma_1 \oplus \Sigma_2 \oplus \ldots \oplus \Sigma_n$

- $F(S_1, \ldots, S_n)_i = \{ s' \in S_i | \exists j \exists s \in S_j: s \rightarrow s' \}$

- Control-flow graph: $(P, \rightarrow)$

- $F(S_1, \ldots, S_n)_i = \{ \langle i, \varepsilon' \rangle | \exists j \rightarrow i : \langle j, \varepsilon \rangle \rightarrow \langle i, \varepsilon' \rangle \}$
Semantic equations

- $i \rightarrow j : \text{operation } op$
- **Notation:** $E_i = \text{set of environments at program point } i$
- $[\text{op}]\varepsilon = \text{semantics of } op$
- System of semantic equations:

$$E_i = \bigcup \{ [\text{op}]E_j \mid j \rightarrow i : \text{op} \}$$

- Solution of the system $= S = \text{lfp } F$
Example

1: n = 0;
2: while n < 1000 do
3:    n = n + 1;
4: end
5: exit

\[ E_1 = \{ n \Rightarrow \Omega \} \]
\[ E_2 = [n = 0]E_1 \cup E_4 \]
\[ E_3 = E_2 \cap ]-\infty, 999] \]
\[ E_4 = [n = n + 1]E_3 \]
\[ E_5 = E_2 \cap [1000, +\infty[ \]
Other kinds of partitioning

In the case of collecting semantics of traces:

• Partitioning w.r.t. procedure calls: context sensitivity
• Partitioning w.r.t. executions paths in a procedure: path sensitivity
• Dynamic partitioning (Bourdoncle)
Methodology

Concrete Semantics

Collecting Semantics

Partitioning

Abstract Semantics

Tuners

Abstract Domain

Abstract Domain

Iterative Resolution Algorithms
Problem: Compute a sound approximation $s^#$ of $s$

Solution: Galois connections
Galois connection

$L_1, L_2$ two lattices

$(L_1, \subseteq) \leftrightarrow (L_2, \leq)$

• $\forall x \forall y : \alpha(x) \leq y \iff x \subseteq \gamma(y)$

• $\forall x \forall y : x \subseteq \gamma \circ \alpha(x) \& \alpha \circ \gamma(y) \leq y$
Theorem:
\[ \text{lfp } F \subseteq \gamma (\text{lfp } \alpha \circ F \circ \gamma) \]
Abstracting the collecting semantics

- Find a Galois connection:

\[ (\emptyset (\Sigma), \subseteq) \xleftrightarrow{\gamma} (\Sigma^#, \leq) \]

\[ \xrightarrow{\alpha} \]

- Find a function:

\[ \alpha \circ F \circ \gamma \leq F^# \]

Partitioning \(\Rightarrow\) Abstract sets of environments
Abstract algebra

- **Notation:** $E$ the set of all environments
- **Galois connection:**

\[
(\emptyset (E), \subseteq) \quad \xleftrightarrow{\gamma} \quad (E^#, \leq)
\]

- $\cup, \cap$ approximated by $\cup^#, \cap^#
- [\text{op}]$ approximated by $[\text{op}]^#

\[
\alpha \circ [\text{op}] \circ \gamma \leq [\text{op}]^#
\]
Abstract semantic equations

1:    n = 0;
2:    while n < 1000 do
3:      n = n + 1;
4:    end
5:    exit

\[ E_1^\# = \alpha (\{n \Rightarrow \Omega\}) \]
\[ E_2^\# = [n = 0]^\#E_1^\# \cup^\# E_4^\# \]
\[ E_3^\# = E_2^\# \cap^\# \alpha ([-\infty, 999]) \]
\[ E_4^\# = [n = n + 1]^\#E_3^\# \]
\[ E_5^\# = E_2^\# \cap^\# \alpha ([1000, +\infty[) \]
Methodology

Concrete Semantics

Collecting Semantics

Partitioning

Abstract Semantics

Abstract Domain

Iterative Resolution Algorithms

Abstract Domain

Tuners
Abstract domains

Environment: \( x \Rightarrow v, \ y \Rightarrow w, \ldots \)

Various kinds of approximations:

- Intervals (nonrelational):
  \( x \Rightarrow [a, b], \ y \Rightarrow [a', b'], \ldots \)

- Polyhedra (relational):
  \[ x + y - 2z \leq 10, \ldots \]

- Difference-bound matrices (weakly relational):
  \[ y - x \leq 5, \ z - y \leq 10, \ldots \]
Example: intervals

1: n = 0;
2: while n < 1000 do
3:   n = n + 1;
4: end
5: exit

- Iteration 1: $E_2^# = [0, 0]$
- Iteration 2: $E_2^# = [0, 1]$
- Iteration 3: $E_2^# = [0, 2]$
- Iteration 4: $E_2^# = [0, 3]$
- ...

...
Methodology

- Concrete Semantics
  - Collecting Semantics
    - Partitioning
      - Abstract Semantics
  - Abstract Domain
    - Abstract Domain

Tuners

- Iterative Resolution Algorithms
Widening operator

Lattice \((L, \leq)\): \(\nabla : L \times L \rightarrow L\)

- Abstract union operator:
  \[ \forall x \forall y : x \leq x \nabla y \land y \leq x \nabla y \]

- Enforces convergence: \((x_n)_{n \geq 0}\)

\[
\begin{aligned}
  y_0 &= x_0 \\
  y_{n+1} &= y_n \nabla x_{n+1}
\end{aligned}
\]

\((y_n)_{n \geq 0}\) is ultimately stationary
Widening of intervals

\[ [a, b] \triangledown [a', b'] \]

- If \( a \leq a' \) then \( a \) else \(-\infty\)
- If \( b' \leq b \) then \( b \) else \(+\infty\)

⇒ Open unstable bounds (jump over the fixpoint)
Iteration with widening

1:  n = 0;
2:  while n < 1000 do
3:    n = n + 1;
4:  end
5:  exit

\[(E_2^#)_{n+1} = (E_2^#)_n \bigtriangleup ([n = 0]^#(E_1^#)_n \cup^# (E_4^#)_n)\]

Iteration 1 (union):  \(E_2^# = [0, 0]\)
Iteration 2 (union):  \(E_2^# = [0, 1]\)
Iteration 3 (widening):  \(E_2^# = [0, +\infty] \Rightarrow \text{stable}\)
Imprecision at loop exit

1: n = 0;
2: while n < 1000 do
3: n = n + 1;
4: end
5: exit; t[n] = 0;

• $E^#_5 = [1000, +\infty[$

• The information is present in the equations
Narrowing operator

Lattice \((L, \leq)\): \(\Delta : L \times L \rightarrow L\)

- Abstract intersection operator:
  \[
  \forall x \forall y : x \cap y \leq x \Delta y
  \]
- Enforces convergence: \((x_n)_{n \geq 0}\)
  \[
  \begin{cases}
    y_0 &= x_0 \\
    y_{n+1} &= y_n \Delta x_{n+1}
  \end{cases}
  \]

\((y_n)_{n \geq 0}\) is ultimately stationary
Narrowing of intervals

\[ [a, b] \Delta [a', b'] \]

- If \( a = -\infty \) then \( a' \) else \( a \)
- If \( b = +\infty \) then \( b' \) else \( b \)

⇒ Refine open bounds
Iteration with narrowing

1: n = 0;
2: while n < 1000 do
3:   n = n + 1;
4: end
5: exit; \( t[n] = 0; \)

\[
\begin{align*}
(E_2^#)_{n+1} &= (E_2^#)_n \Delta ([n = 0]^#(E_1^#)_n \cup^# (E_4^#)_n) \\
\end{align*}
\]

Beginning of iteration: \( E_2^# = [0, +\infty[ \)

Iteration 1: \( E_2^# = [0, 1000] \implies \text{stable} \)

Consequence: \( E_5^# = [1000, 1000] \)
Methodology

- Concrete Semantics
- Collecting Semantics
- Partitioning
- Abstract Semantics

Tuners

- Abstract Domain
- Abstract Domain

Iterative Resolution Algorithms
Tuning the abstract domains

1: n = 0;
2: k = 0;
3: while n < 1000 do
4: n = n + 1;
5: k = k + 1;
6: end
7: exit

• Intervals:
  \[ E_4^\# = \langle n \Rightarrow [0, 1000], k \Rightarrow [0, +\infty] \rangle \]

• Convex polyhedra or DBMs:
  \[ E_4^\# = \langle 0 \leq n \leq 1000, 0 \leq k \leq 1000, n - k = 0 \rangle \]