Abstract Interpretation for Automatic Differentiation, Runtime Error detection and Security Analysis

Christèle Faure

AG60
Contents

• One framework: abstract interpretation of program

• Three applications:
  • Automatic Differentiation
  • Runtime Error detection
  • Automated Security analysis

• Conclusion

• Andreas
Abstract interpretation: practical view

“Executing” programs according to a particular semantic

- Concrete value $\rightarrow$ abstract value
- Concrete execution $\rightarrow$ abstract execution

So that the abstract execution gives correct information about all possible concrete executions
Abstract interpretation: theoretical view

Way of computing

upper (resp. lower) approximations

of the

least (resp. greatest) fixpoint

of a monotonic function

from a complete lattice to itself
Theoretical framework

- **Galois connection: concrete <-> abstract**

- **Abstract domains**
  - Finite / infinite
  - Relational / non relational

- **Abstraction**
  - Value -> set of values
  - Loop in the control -> Fixpoint iteration

- **Approximation**
  - Sources: Loop, compound object, union
  - Solutions: widening and narrowing operators
Example: Manual evaluation

\[ x := 2; \quad x = 2 \]
\[ y := 20; \quad y = 20 \]
\[ \text{while } (x < y) \quad 2 < 20, 4 < 19, 8 < 18, 16 < 17, 32 < 16 \]
\[ \{ x := 2 \times x; \quad x = 4, 8, 16, 32 \} \]
\[ y := y - 1; \quad y = 19, 18, 17, 16 \]
Example: Abstract interpretation with Sign

Nearly no information

\[ x := 2; \quad x > 0 \]
\[ y := 20; \quad y > 0 \]
\[ \text{while } (x < y) \]
\[ \{ x := 2 \times x; \quad x > 0 \]
\[ y := y - 1; \} \quad y \rightarrow T \]
Example: Abstract interpretation with interval

\[ x := 2; \quad x \in [2, 2] \]
\[ y := 20; \quad y \in [20, 20] \]

while \( x < y \)

\[ [2, 2] < [20, 20] \quad [2, 4] < [19, 20] \quad [2, 8] < [18, 20] \]
\[ [2, 16] < [17, 20] \quad [2, 32] < [16, 20] \]

\[ \{ x := 2 \times x; \quad x \in [4, 4] \quad [4, 8] \quad [4, 16] \quad [4, 32] \} \quad y \in [19, 19] \quad [18, 19] \quad [17, 19] \quad [16, 19] \]

\[ x \in [4, +\infty[ \quad x \in 2\mathbb{Z} \cap [4..32] \]
\[ y \in ]-\infty, 19] \quad y \in [16..19] \]
Automatic Differentiation

- **Static analysis**
  - Information
    - Variable aliases
    - Variable activity
      - depend on active inputs
      - Impact active outputs
    - Variable « to be stored » status
      - Modified
      - Overwritten
  - Very simple abstract domain

- **Code transformation**
  - Over approximation => generated program time / memory consuming
  - Static / dynamic => automatic generation of operator overloading

- **Tools:** Odyssée -> Tapenade
Runtime Error detection

- **Information**
  - Alias analysis
  - Value analysis

- **Property check**

  \[
  x := 2; \quad x \in [2,2] \\
  y := 20; \quad y \in [20,20] \\
  \text{while } (x < y) \\
  \{ x := 2 \times x; \quad x \in [4,32] \\
  y := y - 1; \quad y \in [16,19] \\
  \} \quad (x \neq 0 \text{ && } y \neq 0) = \text{True} \\
  z := 1 / (x \times y); \quad \text{Division by zero}
  \]
Runtime Error detection (2)

- **Complex abstract domain**
  - Intervals
  - Congruencies
  - Polyhedrons

- **Static analysis**
  - Over approximation => properties (checks) not “proven”
  - Static / dynamic => automatic generation of unproven checks

- **PolySpace, Astrée, Frama C, Penjili, Fluctuat (numerical errors)**
Software security

• **Actual state**
  - > 200 static or static/dynamic tools
  - Find flaws/bugs/vulnerabilities in piece of software
  - Security analysis
    - Manual
    - Highly skilled people

• **Objective**
  - Computer aided software security analysis
  - Evaluating intrinsic exploitability of flaws
    - Potential attacks
    - Accessibility from inputs
    - Impact on output
    - ...
  - Evaluate effectiveness of protections
Conclusions

• **Static analysis**
  - Same problems (approximations => sometimes useless results)
  - Combine static / dynamic tools

• **Software security**
  - New field
  - Not yet structured (vocabulary / objectives / methods)
  - Exciting subject !
Andreas

- We met at Santa Fe with Nicole Rostaing (1996)
- Gave me a flavor of AD
  - Interesting / open field
  - Established community
- Came at INRIA for one sabbatical year
  - Lot of discussions on forward / backward ...
  - Merge static and dynamic AD
  - Checkpointing
  - Graphical description
  - First AD book
- Nice AD Workshop (2000)
- AD project at INRIA
I remember

- Apples and pieces of bred kept for later on
- Questions about french grammar
  - Why do you say “this” and not “that”?
  - What do you use « le présent du subjonctif » for?
  - I could not answer !!!!

Discovery of the Piggy-Back

- Industrial experiment with Odyssée on Alenia code
- One observation on a gnuplot figure of derivatives for me
- A whole theory for him
  - Convergence acceleration of fix point iteration
  - Piggy-back optimisation