Online seminar series on Verification beyond 2020

Is Static Analysis Successful?

Patrick Cousot

IMDEA Software, Madrid & NYU, New York pcousot@cs.nyu.edu cs.nyu.edu/~pcousot

Tuesday, July $7^{\rm th}$, 2020

🕈 "Is Static Analysis Successful?"

- 1/37 -

 $^{\odot}$ P. Cousot, IMDEA Software, Madrid & NYU, New York, Tuesday, July 7 $^{\rm th}$, 2020

Abstract

When I was invited for the online seminar, I first thought of a technical subject (more precisely, *symbolic terms, substitutions, and systems of equations* which abstract interpretation clarifies by giving them various ground semantics).

Assume $\langle \mathcal{C}, \sqsubseteq \rangle \xleftarrow{\gamma} \langle \mathcal{A}, \preccurlyeq \rangle$ is a Galois connection between posets and α is surjective. If $\langle \mathcal{C}, \sqsubseteq \rangle$ is a complete lattice then so is $\langle \mathcal{A}, \preccurlyeq \rangle$.

So symbolic terms, substitutions, and equations are complete lattices $\langle A, \preccurlyeq \rangle$ where $\langle C, \sqsubseteq \rangle$ is a powerset with ground terms (already well-known but proved "in abstracto" in the literature).

The problem is that substitutions do not have a unique interpretation! This is the origin of great difficulties, misunderstandings, and lot of confusion about substitutions in the literature.

🕈 "Is Static Analysis Successful?"

Abstract

But then, the conversation Andreas and I had on the mainstream success of deductive methods while static analysis stays in the shade, led me to another idea:

discuss whether static analysis is successful, or not, and what are the conditions to make it mainstream, and, why not, immensely popular?

The successes of deductive methods

♥ "Is Static Analysis Successful?"

- 4/37 -

Deductive methods

- Theorem provers
- Proof assistants
- SMT solvers

♥ "Is Static Analysis Successful?"



🕈 "Is Static Analysis Successful?"

- 6/37 -

Theorem provers

- First-order theorem proving started with Jacques Herbrand (invents unification in 1930), John Robinson (invents resolution in 1965), ...;
- Great academic tools: ACL2, B prover of Atelier B, iProver, Prover9, PVS, etc;
- Industrial successes: e.g.
 - B method used to prove the security software of the new Paris driverless metro line 14 (not the control/command software);
 - Verification of all elementary floating-point arithmetics on the AMD Athlon by ACL2¹;
- Not simple, slow, requires specialists, proofs must be changed after each program modification, etc;
- Punctual successes not easily replicated (e.g. B not used for renovation of existing line A of Paris RER).

¹J. Strother Moore, Marijn J. H. Heule: Industrial Use of ACL2: Applications, Achievements, Challenges, and Directions. ARCADE@CADE 2017: 42-45

Theorem provers

- Full automation is hopeless²;
- Real successes mostly came from reducing the initial ambitions:
 - Restrict automation: Proof assistants;
 - Prove less: SMT solvers.

²Henry Gordon Rice, Classes of Recursively Enumerable Sets and Their Decision Problems, Trans. Amer. Math. Soc., 74:1, 1953, 358–366.

"Is Static Analysis Successful?"



🕈 "Is Static Analysis Successful?"

- 9/37 -

Proof assistants

- Interact or program the proof;
- Great academic tools: Isabelle³, Coq⁴, etc;
- Great successes, e.g. for Coq:
 - Four color theorem, Feit-Thompson theorem by Georges Gonthier,
 - CompCert by Xavier Leroy;
- Industrialization (of software proved by Coq):
 - CompCert sold by AbsInt (used by Airbus France);

⁴Calculus of constructions

↑ "Is Static Analysis Successful?"

- 10/37 -

 $^{^3 {\}rm first-order}$ logic (FOL), higher-order logic (HOL) or Zermelo–Fraenkel set theory (ZFC)

Proof assistants

- Despite proving already known theorems, proof assistants are not used
 - by mathematicians (who favor creation over verification),
 - by production engineers;
- Real compilers (LLVM, GCC, etc) are 10 to 50 times larger than CompCert, proofs become inhuman.
- The hope is more for small complex algorithms (e.g. EasyCrypt ⁵ in cryptography).

⁵Gilles Barthe, François Dupressoir, Benjamin Grégoire, César Kunz, Benedikt Schmidt, Pierre-Yves Strub: EasyCrypt: A Tutorial. FOSAD 2013: 146-166



♥ "Is Static Analysis Successful?"

- 12/37 -

SMT solvers

- Restrict what you can prove;
- Fixed reduced product of abstract domains⁶;
- An unexpected formalization by abstract interpretation⁷;
- Great academic tools: CVC3, CVC4, Z3, and many, many, others;
- Some industrial successes e.g.:
 - In R&D on very specific subjects (most often small but complex algorithms)⁸
 - Used by AdaCore to check SPARK contracts;
- Rather unstable over time in competitions;
- At the limit one SMT solver per application⁹.

⁸e.g. Z3 and SMT in Industrial R&D, Nicolaj Bjørner, 2018

⁹could be done by incorporating the adequate abstract domains in the product, presently customization is done by hand.

* "Is Static Analysis Successful?"

- 13/37 -

⁶Patrick Cousot, Radhia Cousot, Laurent Mauborgne: Theories, solvers and static analysis by abstract interpretation. J. ACM 59(6): 31:1-31:56 (2012)

⁷Vijay D'Silva, Leopold Haller, Daniel Kroening: Abstract satisfaction. POPL 2014: 139-150.

[○] P. Cousot, IMDEA Software, Madrid & NYU, New York, Tuesday, July 7th, 2020

ls static analysis successful?

↑ "Is Static Analysis Successful?"

- 14/37 -

Static analysis is hard to understand

Contrary to deductive methods (check the work of mathematicians), static analysis (check the work of programmers) is more difficult to perceive

- by the general public;
- by programmers (who learn to prove theorems at school but not to prove and, even less, to analyze a program with a tool¹⁰).

¹⁰Compilers are the only universally used static analyzers.

Static analysis is harder than verification (by deductive methods)

Program Analysis Is Harder Than Verification: A Computability Perspective

Patrick Cousot¹⁽[®]), Roberto Giacobazzi^{2,3}^{(®}), and Francesco Ranzato⁴^(⊠)^(®)

 ¹ New York University, New York City, USA
² University of Verona, Verona, Italy
³ IMDEA Software Institute, Madrid, Spain
⁴ University of Padova, Padova, Italy ranzato@math.unipd.it

Abstract. We study from a computability perspective static program analysis, namely detecting sound program assertions, and verification, namely sound checking of program assertions. We first design a general computability model for domains of program assertions and corresponding program analysers and verifiers. Next, we formalize and prove an instantiation of Rice's theorem for static program analysis and verification. Then, within this general model, we provide and show a precise statement of the popular belief that program analysis is a harder problem than program verification: we prove that for finite domains of program assertions, program analysis and verification are equivalent problems, while for infinite domains, program analysis is strictly harder than verification.

"Is Static Analysis Successful?"

- 16/37 -

(ロト・イラト・イミト・モート) © P. Cousot, IMDEA Software, Madrid & NYU, New York, Tuesday, July 7th, 2020

Two additional difficulties for static analysis

- induction (no inductive invariants are given to static analysers);
- no universal interface (to many programming languages, libraries, and systems).

The context of static analysis is quite diverse

- Academic and industrial tools;
- Academic and industrial users;
- Explicit versus implicit specifications;
- Small models to large programs;
- Sound and unsound tools;
- Bug finding versus verification;
- Terminology is confusing (static analysis versus software checking);
- Many different levels of ambitions (from compilers, linters, unsound commercial tools, semantic-based, sound, and precise tools, to verifiers);

 \rightarrow hard for non-specialists to have a clear understanding of the field;

 \rightarrow a lot of space for unscrupulous charlatans.



- 18/37 -

© P. Cousot, IMDEA Software, Madrid & NYU, New York, Tuesday, July 7th, 2020



Keys to static analysis successes

♥ "Is Static Analysis Successful?"

- 19/37 -

Specifications are inexistent

- Formal semantics of languages;
- Requirements for programs;
- Few exceptions: e.g. MISRA C for automobile, DO-178C for the avionics.

```
DOI:10.1145/2736348Lestie LamportViewpointWho Builds a HouseWho Builds a Housewithout DrawingBlueprints?
```

¹¹Static analysis of blueprints is certainly also useful!

Program certification is not mandatory

- No regulation on software quality;
- A facility rather than an obligation;
- Obligation of means rather than results;
- Software engineering is empiricism but no science;

Benchmarks are biased

- Academic benchmarks do not reflect industrial needs;
- Industrial benchmarks are not publicly available.

Independent evaluations are rare

Few independent comparative evaluations;

 \rightarrow choosing the appropriate static analysis tool is very difficult for engineers and managers.

Example 1 of benchmarks

Benchmarking Software Model Checkers on Automotive Code

Lukas Westhofen¹, Philipp Berger², and Joost-Pieter Katoen²

¹ OFFIS e.V., Oldenburg, Germany lukas.westhofen@offis.de ² RWTH Aachen University, Aachen, Germany {berger, katoen}@cs.rwth-aachen.de

Metric	DSR	ECC
Complexity		
Source lines of code	1,354	2,517
Cyclomatic complexity	213	268

Requirement Characteristics.

Invariant properties are assertions that are supposed to hold for all reachable states. Bounded-response properties request that a certain assertion holds within a given number of computational steps whenever a given, second assertion holds.

Coverage. Fig. 2 shows the verification results of running the open-source verifiers on the two case studies, omitting the results of the witness validation.



- 24/37 -

Example 2 of benchmarks

NISTIR 8304 SATE VI Ockham Sound Analysis Criteria

Paul E. Black Kanwardeep Singh Walia https://doi.org/10.6028/NIST.IR.8304

The criteria were:

- 1. The tool is claimed to be sound.
- 2. For at least one weakness class and one test case the tool produces findings for a minimum of 75 % of appropriate sites.
- 3. Even one incorrect finding disqualifies a tool for this SATE.

Our conclusion is that Astrée and Frama-C with Eva satisfied the SATE VI Ockham Sound Analysis Criteria.

Astrée produces 36316 alarms including all 18954 known bugs in the test suite (plus a number that where unknown but real). Note: Astrée is not a general purpose static analyzer, which is not reflected in the Juliet 1.3 (<u>https://samate.nist.gov/SRD/testsuite.php</u>) test suite.

Frama-C with Eva produces 42056 alarms including all 18954 known bugs in the test suite.

The analyzers that fail the test are not mentioned in the politically-correct report.



Of course not everybody agrees with the significance of the test suite:



🕈 "Is Static Analysis Successful?"

- 26/37 -

Tool qualification is inexistent

- Almost no requirements on the static analysis tools used in industry; \rightarrow you can run any tool to have a clear conscience!
- An exception: DO-333 (Formal Methods) for avionics (Astrée is DO-333-qualified, meaning it can replace unit tests for runtime errors¹²).

¹²Astrée 20.4 supports Annex J of ISO/IEC 9899:1999 (E) and ISO/IEC TS 17961:2013 C guidelines, the Common Weakness Enumeration (CWE) catalog, the SEI CERT C Coding Standard, the MISRA-C:2004, MISRA-C:2012 (including Amendments 1 and 2), MISRA-C++:2008 rules.

Education is parcimonious

- Static analysis theory and practice is (almost) not taught;
- The use of static analysis is not even mentioned in programming courses.

Human resources are scarce

- Very few high-priced specialists for designing static analyzers;
- But, after one month engineers can use static analyzers autonomously;
- Not true for deductive methods.



Job Description:

The DexOpto regularizations is looking for Software Engineers passionate about research and development of Google-scale deep regramm analysis tools, based on adment interpretation. We are starting a new team that will develop cutting edge deep Go static analysis tools and deploy them in production. We are looking for individuals who enjoy collaborative teamwork (hiero on computationality) hand problema, are motivated by impact, deeply can about software correctness and security, and have a strong academic background in program analysis.

- 29/37 -

(© P. Cousot, IMDEA Software, Madrid & NYU, New York, Tuesday, July 7th, 2020

Many bad designs

- Control: graph, SSA, etc versus structural induction (e.g. Astrée, Zoncolan at FB);
- Data; unique representation of properties (e.g. Infer) versus structured in abstract domains (e.g. Astrée);
- Extensibility: most often not considered in the original design (e.g. Infer), with exceptions (Astrée, Frama-C with Eva plugin);



🕈 "Is Static Analysis Successful?"

- 30/37 -

Long term support

- Academic: require permanent self-funded institutions (e.g. INRIA for Ocaml/Coq);
- Free-software: too complicated (e.g. Clousot is public domain makes no significant progress since Fähndrich & Logozzo left MS research for FB);
- Industrial: indispensable but few competent and not extremely profitable (academic Astrée: 60.000 lines of Ocaml, AbsInt: 265.000, not counting the much larger user interface).



- 31/37 -



 $\label{eq:product} \begin{array}{cccc} & < & \square \\ & & & \blacksquare \\ \end{array} \\ & & & \blacksquare \\ \\ & & \blacksquare \\ \end{array} \\ & & & \blacksquare \\ \\ & & \blacksquare \\$

Irresponsibility

- Programmers are never held responsible for their errors, even when the human and economic consequences are huge¹³;
- Software engineers are guaranteed qualified immunity under the pretext that verification is beyond best practice;
- If best practice would include the mandatory use of standards and qualified tools, programmers and their hierarchy could be held accountable at least for definite bugs automatically found be static analysis tools.



DOI:10.1145/2631185

Vinton G. Cerf

Responsible Programming

¹³e.g. 2009–11 Toyota vehicle recalls, Boeing 737 MAX groundings.

"Is Static Analysis Successful?"

- 32/37 -

Successes remain unknown

- AbsInt has sold thousands of industrial licenses of Astrée;
- Who knows that?
- Which formal methods can make such a claim?

Towards an Industrial Use of Sound Static Analysis for the Verification of Concurrent Embedded Avionics Software

Antoine Miné École Normale Supérieure 45, rue d'Ulm F-75230 Paris Cedes 05, France mine@di.ens.fr David Delmas Airbus Operations S.A.S. 316, route de Bayonne 31060 Toulouse Cedex 9, France david.delmas@airbus.com

Google scholar citations: 21



Technical Note

EVALUATION OF STATIC ANALYSIS TOOLS USED TO ASSESS SOFTWARE IMPORTANT TO NUCLEAR POWER PLANT SAFETY

ALAIN OURGHANLIAN

EDF Lab CHATOU, Simulation and Information Technologies for Power Generation Systems Department, EDF R&D, 6 quai Watier, BP 49, 78401 Chatou Cedex, France

- 33/37 -

৫ P. Cousot, IMDEA Software, Madrid & NYU, New York, Tuesday, July 7th, 2020



♥ "Is Static Analysis Successful?"

- 34/37 -

(ロ > (⑦ > 〈 豆 > 〈 豆 > 〈 豆 > ´ 豆 > ´ 豆 ^ つ Q (^ © P. Cousot, IMDEA Software, Madrid & NYU, New York, Tuesday, July 7th, 2020

Necessary conditions to make static analyzers mainstream

- Specify before programming;
- Program certification is mandatory;
- Benchmarks are publicly available;
- Independent evaluations are fair and public;
- Tools qualification is mandatory;
- Education on static analyzers starts with programming
- Human resources are considerably developed;
- Tool designs are principled;
- Tools are supported in the long term;
- Programmers are made responsible for their errors;
- Industrial successes are glorified.

Necessary conditions to make static analyzers mainstream

- Specify before programming;
- Program certification is mandatory;
- Benchmarks are publicly available;
- Independent evaluations are fair and public;
- Tools qualification is mandatory;
- Education on static analyzers starts with programming
- Human resources are considerably developed;
- Tool designs are principled;
- Tools are supported in the long term;
- Programmers are made responsible for their errors;
- Industrial successes are glorified.

Hopefully, sufficient! 😂

The End, Thank you

🕈 "Is Static Analysis Successful?"

- 36/37 -