Exploring interactions between automated program analysis and witnesses

Internship proposal for Spring 2026

Raphaël Monat
Informal enquiries are welcome by email (in English or French)
Other topics around static analysis can be designed, feel free to reach out.

1 Context – Static Program Analysis

One approach aiming at reducing the number of bugs is static program analysis through the framework of abstract interpretation created by Radhia and Patrick Cousot [2]. Contrary to dynamic analyses such as fuzzing [5], the program is not executed but its source code is analyzed. Thanks to this approach, the analysis conservatively considers all possible execution paths of the program during the analysis, ensuring the absence of false negatives. In addition, the analyses are automatic: they do not require any user interaction to complete their task and they will be completed in a guaranteed finite time. These analyses can be seen as "push-button" as no expert knowledge is required to run them. This approach has been particularly successful to certify the absence of runtime errors in critical embedded C software. Astrée [3] has proved the absence of runtime errors in software of Airbus planes. More recently, Frama-C's static analysis has been used on the code of nuclear power plants [1].

2 Goal – Exploring interactions between automated program analysis and witnesses

The goal of this project is to explore interactions between automated program analysis and witnesses used in the international Software-Verification Competition. These witnesses are used to check the results of another program analyzer.

Several directions can be pursued:

- In "Correctness Witness Validation by Abstract Interpretation" Saan et al. [4] describe a method to guide automated program analysis using correctness witnesses generated by other tools. It would be interesting to implement this approach within the Mopsa static analysis platform, in order to experimentally evaluate the benefits of the approach. Interactions with deductive program verifiers could also be investigated.
- Mopsa currently cannot export witnesses. It would be interesting to design a general, scalable and precise witness export.

3 Developed skills

The candidate will work in the overall field of formal methods and programming language theory. They will work on conservative static analyses, and in particular some rooted in the framework of abstract interpretation. We expect the successful candidate to be motivated to improve experimental research tools such as Mopsa, which is implemented in the OCaml functional programming language.

4 Logistics

The intern will be part of the SyCoMoRES team of Inria Lille & CRIStAL lab. Lille is a city close to Brussels, Paris & London, easily reachable by train, with a large student population and a number of cultural places & events. The lab has a very active equality and parity commission, which raises awareness on this topic to all staff (with specific events for newcomers), and provides outreach activities for high-schoolers. The advisor is an active member of this commission.

We plan to hold weekly in-person research meetings. In addition, the student will be able to attend monthly meetings with other Mopsa practitioners. This research project is part of ANR JCJC RAISIN.

References

[1] Patrick Baudin, François Bobot, David Bühler, Loïc Correnson, Florent Kirchner, Nikolai Kosmatov, André Maroneze, Valentin Perrelle, Virgile Prevosto, Julien Signoles, and Nicky Williams. The dogged pursuit of bug-free C programs: the Frama-C software analysis platform. *Commun. ACM*, (8), 2021. doi:10.1145/3470569.

[2] Patrick Cousot and Radhia Cousot. Abstract interpretation: A unified lattice model for static analysis of programs by construction or approximation of fixpoints. In *POPL 1977*, 1977. doi:10.1145/512950.512973.

[3] Patrick Cousot, Radhia Cousot, Jérôme Feret, Laurent Mauborgne, Antoine Miné, David Monniaux, and Xavier Rival. Combination of abstractions in the astrée static analyzer. In *ASIAN 2006*, 2006. doi:10.1007/978-3-540-77505-8_23.

[4] Simmo Saan, Michael Schwarz, Julian Erhard, Helmut Seidl, Sarah Tilscher, and Vesal Vojdani. Correctness witness validation by abstract interpretation. In Rayna Dimitrova, Ori Lahav, and Sebastian Wolff, editors, *Verification, Model Checking, and Abstract Interpretation - 25th International Conference, VMCAI 2024, London, United Kingdom, January 15-16, 2024, Proceedings, Part I*, volume 14499 of *Lecture Notes in Computer Science*, pages 74–97. Springer, 2024. doi:10.1007/978-3-031-50524-9_4. URL https://doi.org/10.1007/978-3-031-50524-9_4.

[5] Andreas Zeller, Rahul Gopinath, Marcel Böhme, Gordon Fraser, and Christian Holler. *The Fuzzing Book*. CISPA Helmholtz Center for Information Security, 2024. URL https://www.fuzzingbook.org/. Retrieved 2024-07-01 16:50:18+02:00.