Type-based Enforcement of Infinitary Trace Properties for Java

Serdar Erbatur[1], Ulrich Schöpp[2], and Chuangjie Xu[2]

[1] University of Texas at Dallas, USA
[2] fortiss GmbH, Munich, Germany

The relative simplicity of typical programming guidelines makes them a good target for fully automatic static analysis techniques. Erbatur *et al.* [2] introduce a region-based effect and type system to ensure adherence to programming guidelines by tracking event traces of terminating programs. Building on the ideas of Hofmann and Chen [3], we extend their work to capture also *infinite* traces produced by *non-terminating* programs. We develop a type and effect system for Featherweight Java [4] that can express properties of both finite and infinite traces, and prove its soundness with respect to the operational semantics. The system can compute information about the possible infinite traces of Featherweight Java programs. Specifically, the set of infinite traces of a method is constructed as the greatest fixed point of the operator which calculates the possible traces of method bodies. Our type inference algorithm is realized by working with the *finitary abstraction* [1] of the system based on Büchi automata. We have a prototype implementation of type inference based on the Soot framework [5] and are in the process of developing it into a full analysis tool.

References

1. Patrick Cousot and Radhia Cousot. Abstract interpretation: A unified lattice model for static analysis of programs by construction or approximation of fixpoints. In Robert M. Graham, Michael A. Harrison, and Ravi Sethi, editors, *Principles of Programming Languages (POPL 1977)*, pages 238–252, 1977.
2. Serdar Erbatur, Martin Hofmann, and Eugen Zălinescu. Enforcing programming guidelines with region types and effects. In Bor-Yuh Evan Chang, editor, *Programming Languages and Systems (APLAS 2017)*, volume 10695 of *Lecture Notes in Computer Science*, pages 85–104. Springer, Cham, 2017.
3. Martin Hofmann and Wei Chen. Abstract interpretation from Büchi automata. In *Computer Science Logic and Logic in Computer Science (CSL-LICS 2014)*, pages 51:1–51:10. Association for Computing Machinery, 2014.
4. Atsushi Igarashi, Benjamin C. Pierce, and Philip Wadler. Featherweight Java: a minimal core calculus for Java and GJ. *ACM Transactions on Programming Languages and Systems*, 23(3):396–450, 2001.
5. McGill University Sable Group. Soot - A framework for analyzing and transforming Java and Android Applications. URL: `https://soot-oss.github.io/soot/`.