Augmenting Programs with Analysis Results*

Adrian Prantl
Institut für Computersprachen
Technische Universität Wien, Austria
adrian@complang.tuwien.ac.at

Abstract

In this talk, we propose a new storage method for the results of a static program analysis. Instead of attaching the analysis results to the control-flow graph, we automatically transform the program into a augmented version that uses assert-statements to explicitly check the analyzed properties at runtime. This transformation opens up multiple applications:

- *Automatic testing of the analysis implementation:* This can be done dynamically, by generating an encompassing main function that executes the program with sensible input data. As a more expensive but complete alternative, this step can also be done statically via model checking.
- *Freezing of interfaces:* Since the transformed program is naturally a more restricted version of the original program, the method can also be used to explicitly encode e.g. calling conventions. This way, the program can be hardened against future modifications.
- *Aiding compiler optimizations:* By rewriting analysis results as explicit tests a subsequent compilation process can use the invariants to generate optimized code.

The presented method has been implemented in the SATIrE source-to-source analysis framework using the Termite program transformation library. First experiments showed promising results and proved already to be valuable for the debugging of analysis implementations.

References

SATIrE: `http://www.complang.tuwien.ac.at/satire/`
Termite: `http://www.complang.tuwien.ac.at/adrian/termite/`

*This work has been partially supported by the Austrian Science Fund (Fonds zur Förderung der wissenschaftlichen Forschung) under contract P18925-N13