Thesis Defense

Institute for Software Research
Software Engineering

Automated Program Transformation for Improving Software Quality

Rijnard van Tonder

Thursday, August 22, 2019
10:00 AM - 1:00 PM
Gates & Hillman Center 9115

Software bugs are not going away. Millions of dollars and thousands of developer-hours are spent finding bugs, debugging the root cause, writing a patch, and reviewing fixes. Automated techniques like static analysis and dynamic fuzz testing have a proven track record for cutting costs and improving software quality. More recently, advances in automated program repair have matured and see nascent adoption in industry. Despite the value of these approaches, automated techniques do not come for free: they must approximate, both theoretically and in the interest of practicality. For example, static analyzers suffer false positives, and automatically produced patches may be insufficiently precise to fix a bug. Such limitations continue to impose substantial human effort amid the benefits of automation.

Software development activities revolve around changing code. Thus, performing and reasoning about program change has extensive bearing on the effectiveness of automated techniques. From this perspective, we develop new automated techniques for changing programs to improve analysis behavior, and correspondingly, use automated reasoning and analysis to specialize program changes for automated program repair. We present new evidence that automated program transformation, program analysis, and program repair are interrelated and cooperative. We first show that automated program transformation leads to higher quality static analysis (by reducing false positives) and dynamic fuzz testing (by reducing duplicate bug reports). We then demonstrate that high-quality static analyses enable new automated program repair techniques, and how automated repair further complements static analysis (e.g., by enabling the analysis to detect more true positive bugs). The thesis is that automated syntactic and semantic search and application of program transformations enables efficient, scalable, and unassisted techniques for improving the effectiveness of existing program analyses and end-to-end repair of real-world programs.

We show that our techniques are effective compared to current approaches in the respective domains of static analysis, dynamic fuzz testing, and program repair. We demonstrate real-world applicability on large, popular, and active projects across multiple languages. Our work presents new capabilities in automated program transformation that fosters effective ways to automate burdensome human effort and reasoning incurred by limitations in program analysis and repair.

Committee: Dr. Claire Le Goues (Chair), Dr. Christian Kästner, Dr. Jan Hoffmann, Dr. Manuel Fähndrich (Facebook)

institute for SOFTWARE RESEARCH | Carnegie Mellon University School of Computer Science