Bachelor/Master Thesis Topic

Automated Validation of Patch Correctness and Maintainability with Symbolic Execution

Motivation and Background
Automated patch validation for real-world software has been drawing growing attention in the field. Many tools have been developed to examine the correctness of patched programs. For example, the state-of-the-art KATCH [1] aims to automatically generate high coverage tests for patches to discover regression bugs, while other tools leverage test/execution similarity between original buggy and patched programs [2] or investigate in better oracle for automatically generated patches [3, 4]. In these tools, techniques such as dynamic symbolic execution [5, 6] are usually involved. As for patch maintainability, despite of previous contribution to the maintainability problem of patches [7, 8] or programs [9], the evaluation of the maintainability of patches still lack standard. The challenge of this thesis is to identify both the correctness and the maintainability of (manually or automatically generated) patches.

Goals
The student uses techniques such as symbolic execution and dynamic program analysis to validate patches (e.g. discovering regression bug/crashes). After that, the student should further evaluate the patches’ maintainability from different aspects (e.g. syntax, semantic and AST [10]), which may be concluded in some metrics. The validation of the resulting tool can be conducted on real-world patches or patches generated by automated program repair tools.

Description of the Task
A detailed description of the task and the underlying techniques will be given personally on interest.

Research Type
Theoretical Aspects:  ****
Industrial Relevance:  ****
Implementation  ****

Prerequisite
The student should be enrolled in the bachelor/master of software engineering or bachelor/master of computer science program, and has completed the required course modules to start a bachelor/master thesis.

Skills required
Programming skills in (preferably) Java and C/C++, Understanding of, or willingness to learn, the architectural and statistical foundations needed for the project.

References


Contacts
Lars Grunske/Minxing Tang, Humboldt-Universität zu Berlin, Institut für Informatik, Lehrstuhl Software Engineering, Unter den Linden 6, 10099 Berlin, Germany
Application
Please contact during office hours or write an email with the title: “AV-PCM-SE” to se-career@informatik.hu-berlin.de