Project Proposal

Robin Zhang, lz2811@columbia.edu

Project Description

Automatic Program Repair (APR) is massively adopted by the developers by automatically generating different patches so that the software could function without eliminating the expected features. But such repair has limitations: their patches is incorrect since the APR is not complete. One significant problem is patch overfitting, which means the APR program generates many patches that exceed the actual test suite. Thus, the usage of formal verification could be a possible way to mitigate such problems. Formal verification is known for its completeness and the adoption of static analysis, which is a great compensation for APR programs. Thus, I am proposing a tool/framework that evaluates the effectiveness of an APR tool for the developers by analyzing its possible of overfitting.

This program is mainly focused on assessing the overfitting problem on APR tools working on Java programs. The Java developers could see the possibility of a certain APR tool, such as NOPOL, overfitting their program using this framework, which takes different modules from the program and verifies the patches the APR tool generates. OpenJML will be used as the verifier of Java programs in this framework. And the testing examples and benchmark standards are from Defects4J and an open repository[1]. The repository provides 537 buggy sample programs with written formal specifications which can be used on different APR tools. They cover a large number of functionalities normally seen in common Java programs. And these sample programs could be extended to be tested on more Java APR tools used in the repository. But I might modify the main functionalities of the framework I proposed as I get more handy on the code and automatic bug fixing theory.

Personal Motivation

As a more theory-interested person, I focus on abstract concepts in the field of computer science at most of the time. But in the end, the theory has to be properly utilized in the real world to make a difference. So one practical issue that programmers are dealing with everyday is debugging, and APR is one of the best tools since it reduces the time on manually validating the code. Instead of working on the APR tools themselves, I think it is more applicable of formal verification on assessing APR tools. Since I have researched a bit on formal verification for my midterm paper, my knowledge could help me to deal with how to verify patches and apply OpenJML, so I could extend my understanding of formal verification and dive into APR.

[1] https://github.com/Amirfarhad-Nilizadeh/BuggyJavaJML