Post-doc Positions at National University of Singapore

Contact: abhik@comp.nus.edu.sg and ilya@comp.nus.edu.sg with your CV.

Sample Postdoc Position 1: Static Analysis-Guided Automated Program Repair

This project will focus on novel practical and theoretical aspects of designing, enhancing, and engineering static program analyses, as they are employed by *automated program repair* (APR). Unlike testing-based approaches, static analyses infer properties of programs without executing them, thus, accounting for all possible run-time executions, and therefore, side-stepping the over-fitting issues, common for APR. What makes static analyses particularly appealing for enabling APR in the style of Continuous Integration (CI) is the fact that the main by-product of their work, *program summaries*, already capture an abstraction of the code's run-time properties, serving both as its specification as well as a condensed characterisation of erroneous behaviors.

On the practical side, the goal of this project is to decrease the amount of *friction* (*i.e.*, required human effort) for adopting APR as an inherent part of the software development process. Towards this agenda, we will be investigating a spectrum of *automated static program analyses* with regard to the program properties they infer and verify: from general safety (e.g., crash and data-race freedom) to domain-specific functional correctness. The theoretical side of this project will be dedicated to formulating the results that adequately capture the desired notions of *soundness* and *completeness* of the analyses used for the purpose of APR.

Sample Postdoc Position 2: Proof-Based Program Repair

Deductive software verification is the discipline founded on mathematical logic and concerned with ensuring that programs are safe and correct with respect to an ascribed rigorous specification. Crucially, upon failing to verify that a program adheres to its specification, such verifiers often offer meaningful information on the cause and effects of the detected bugs in a form of a partially derived *proof derivation*, as opposed to only pinpointing to the locations where bugs manifest (which is the case for fully automated analyses). In this project, we will leverage the power of deductive software verification to provide a new approach to automated program repair, which creates *verified* patches, i.e., patches formally proved to fix the existing bugs and to introduce no new ones.

In the scope of this project, we will explore the proof-based repair techniques targeting both verified codebases (whose proof-preserving evolution still poses an open challenge), as well as existing large-scale software projects, which are to be instrumented with basic correctness specifications.