

Name: Cheng Zhang

Title: Frequency Estimation of Virtual Call Targets for Object-Oriented Programs

The information of execution frequencies of virtual call targets is valuable for program analyses and optimizations of object-oriented programs. However, to obtain this information, most of the existing approaches rely on dynamic profiling. They usually require running the programs with representative workloads, which are often absent in practice. Additionally, some kinds of programs are very sensitive to run-time disturbance, thus are generally not suitable for dynamic profiling. Therefore, a technique which can statically estimate the execution frequencies of virtual call targets will be very useful. In this talk we propose an evidence-based approach to frequency estimation of virtual call targets. By applying machine learning algorithms on the data collected from a group of selected programs, our approach builds an estimation model to capture the relations between static features and run-time program behaviors. Then, for a new program, the approach estimates the relative frequency for each virtual call target by applying the model to the static features of the program. Once the model has been built, the estimation step is pure static, thus does not suffer the shortcomings of existing dynamic techniques. We have performed a number of experiments on real-world large-scale programs to evaluate our approach. The results show that our approach can estimate frequency distributions which are much more informative than the commonly used uniform distribution.

\*\*\*\*\*

Name: Qiang Sun

Title: Probabilistic Points-to Analysis for Java

Probabilistic points-to analysis is an analysis technique for defining the probabilities on the points-to relations in programs. It provides the compiler with some optimization chances such as speculative dead store elimination, speculative redundancy elimination, and speculative code scheduling. Although several static probabilistic points-to analysis techniques have been developed for C language, they cannot be applied directly to Java because they do not handle the classes, objects, inheritances and invocations of virtual methods. In this paper, we propose a context-insensitive and flow-sensitive probabilistic points-to analysis for Java (JPPA) for statically predicting the probability of points-to relations at all program points (i.e., points before or after statements) of a Java program. JPPA first constructs an interprocedural control flow graph (ICFG) for a Java program, whose edges are labeled with the probabilities calculated by an algorithm based on a static branch prediction approach, and then calculates the probabilistic points-to relations of the program based upon the ICFG. We have also developed a tool called Lukewarm to support JPPA and conducted an experiment to compare JPPA with a traditional context-insensitive and flow-sensitive points-to analysis approach. The experimental results show that JPPA is a precise and effective probabilistic points-to analysis technique for Java.