Abstract

Program analysis techniques have many uses in the real world. However, there are aspects that can help improving its widespread adoption. This thesis deals with techniques aimed at improving two limiting aspects: the lack of customizability and the burdensome learning process.

The lack of customizability of program analysis comes from the high complexity of program analysis algorithms that prevents untrained developers from creating specific analysis that help to improve the quality of their software. Declarative program analysis aims at reducing the effort for implementing analyses by incrementing the level of abstraction of the specification language, and by offering an efficient execution of the analysis specification comparable with traditional implementations.

In this thesis, we improve declarative program analysis of JAVA programs based on the logic specification language Datalog in two aspects. On one hand, we translate Datalog specifications into *Boolean Equation Systems* for easing the distribution of the analysis computation and speeding it up. *Boolean Equation Systems* provide efficient and distributed algorithms for their evaluation and industrial tools that implement them. On the other hand, we translate Datalog specifications into *Rewriting Logic* theories to support the extension of the specification language Datalog for expressing more sophisticated analysis, for example, those involving the use of *reflection*.

Another contribution of this thesis is relative to the automated inference of specifications to assist techniques for improving software quality (program analysis, verification, debugging, documentation, ...). In particular, specifications are at the core of program analysis, since every program analysis checks the conformation of the program of interest with respect to a specification. Untrained developers may not be capable of formulating program specifications that could be used as input of other tools like static analyzers, testing or program verifiers, or just for human inspection and documentation. Automated inference of specifications aims at reducing the effort for creating software specifications by producing approximated ones without human intervention. We improve automated specification inference for the multiparadigm language CURRY and for object-oriented programs in general by proposing two new approaches. On one hand, we present a technique to infer algebraic specifications for CURRY programs by classifying expressions built from their

signature according to their abstract semantics. Contrary to other existing approaches, this technique allows to discriminate between *correct* and *possibly correct* parts of the specification. On the other hand, we present a technique to infer high-level specifications in the form of pre/post-conditions for object-oriented languages. This technique is formalized in the *Matching Logic* verification setting to allow the verification of the inferred specifications.