Dynamic Invariant Detection for Program Understanding and Reliability
MIT2001-01

Progress Report: January 1, 2002—June 30, 2002

Michael Ernst

Project Overview

The Program Analysis Group (PAG) of MIT is collaborating with Tadashi Araragi of NTT on problems raised by NTT's Erdös internet agent system. The goal is to formally verify mobile agent systems in which agents may be widely separated and may move about a network, yet must communicate, coordinate, and negotiate in order to perform useful work. Examples include internet marketplaces (in which directories of shops must be maintained), auctions (in which programs may be dynamically downloaded), and airline ticket reservation (in which multiple ticket agencies must have a consistent view of a database). In each of these cases, it is crucial that the system perform correctly, yet it is difficult to understand and verify it. PAG has provided several technologies that can increase confidence in such systems.

Progress Through June 2002

PAG's foremost contribution is a technique for automatically generating program specifications, and an implementation (named Daikon) of the technique. Program specifications are required for most program understanding, maintenance, and verification tasks, but are rarely present. One reason is that people find it tedious, and error-prone to write specifications; another is that it is as difficult to write a good specification as to write the program in the first place. Therefore, automating the task holds great promise. The Daikon system automatically generates specifications; it requires only a program and a test suite.

Between January and June 2002, we have made additional progress on the project's goals.

We have continued to improve our tools infrastructure and make it easier to use. In particular, we made seven releases of the Daikon system. These incorporate an expanded manual, a significantly more robust C front end, new user-settable configuration and debugging flags, and bug fixes. Three more substantive projects were transitioned from internal research to the public software releases. The first compares formal specifications (that is, sets of invariants) and indicates differences between them. This permits different programs, or different test suites, or different versions of a program to be compared to one another, permitting easy understanding of how they differ in behavioral semantics. The second inserts invariants into a program as stylized comments; presently, these comments can be read by the ESC/Java static checker, but work is underway to extend this work to the Java Modeling Language (JML) toolkit and other tools. This permits the system to be integrated with other research and commercial systems that work with specifications. The third augmentation adds a static analysis technique that aids in detecting implications, or conditional invariants. These are properties of the form "*p* □ *q*", where *q* is not universally true, but holds only when *p* does. (In July, we added a clustering technique for the same goal.)

Because the final goal of this NTT-MIT project is program verification, PAG has continued to press forward on proofs of program correctness in two domains: Java and the IOA programming language. The Java experiments have used the ESC/Java program checker from the Compaq Systems Research Center. They have resulted in two conference publications. One shows that the technique is much more accurate than might have been predicted of a dynamic tool that operates over runs; this indicates that users need not be unduly concerned about its unsoundness. The other shows that the output aids users who are interested in proving that their programs do not throw run-time exceptions. We have continued to use the techniques in other research.

IOA is a language for modeling distributed systems that is being developed by Prof. Nancy Lynch's Theory of Distributed Systems (TDS) group. It is the subject of another NTT-MIT collaboration, and both research projects have benefited from the close relationship between the research topics and the principal investigators. The IOA work has been done in collaboration with TDS. We have completed three semi-automated Daikon-assisted proofs of distributed algorithms: mutual exclusion, a consensus protocol, and a distributed cache. The IOA experiments use the LP theorem-prover written by Dr. Stephen Garland (another NTT-MIT PI); there is an automatic mapping from IOA to LP formulas. However, the human user must still provide many small facts about the program; these are distracting and easy to forget. Daikon produces many of the desired invariants and intermediate steps required for a proof in LP. Human direction is still required for the high-level direction of the proof and for some of the smaller steps. (Since July, we have started investigating use of the Isabelle theorem-prover from Cambridge University and TU Munich. Although we do not already have local expertise with it, it has a larger user base, it is under active development, it has built-in support for arithmetic, and it has user-programmable tactics that can obviate much user input. Applying our system to a second theorem-prover will further indicate its generality and utility.)

The results for both Java and IOA programs suggest that similar verification results may be possible in the Erdös system.

We have obtained preliminary results in detecting temporal invariants -- properties that describe behavior over time. For example, the property of a traffic signal, “no two directions show green simultaneously” is true over all time, but “after a car arrives, the signal eventually turns green” is a temporal property. We have implemented a prototype system that detects temporal properties and have applied it to both a set of textbook examples and to commercial codes. In the latter, we discovered deficiencies in the test suite, confirmed sets of behaviors that were intended to hold, and increased confidence in a (proprietary) compiler. Presently, it has an impoverished notion of scope, and it checks only simple relationships such as “preceeds” and “follows”. We will continue to extend this work in the next six months.

Research Plan for the Next Six Months

Our research is just beginning to bear fruit, and we will continue to improve the results and extend them into new directions. We will continue to support the Daikon system and its users, adding functionality and making research results more broadly available. We will integrate Daikon with the Isabelle theorem-prover, generalizing our techniques as necessary and producing fully automatic proofs that require no human intervention whatever – not even to indicate what goal to prove! We will improve our detection of temporal invariants, moving it from a rough prototype to something that can be run on substantial programs. We will collaborate with Dr. Tadashi Araragi and Dr. Seung Mo Cho of NTT in their use of the Daikon tool and its applications to theorem-proving, temporal properties, and other problems. If possible, we will obtain NTT programs in order to analyze them in parallel with our NTT colleagues.