Introduction to some verification tools

Qiang Zhang

Dec. 12, 2007

Introduction

- ESC/Java
- Java PathFinder
- KeY
- Caduceus

ESC/Java

- Extended Static Checker for Java
- parse JML-like annotations (pragmas) in a Java program
- warn about annotations that may not be justified
- ESC/Java2 extended to support JML annotations
- download:

  `http://kind.ucd.ie/products/opensource/ESCJava2/download.html`
- manual:

  `http://gatekeeper.research.compaq.com/pub/DEC/SRC/technical-notes/SRC`

ESC/Java

- decision procedure: Simplify
- attempt to find common run-time errors, but not to provide formally rigorous verification
- static analysis of the java source code
- Users control the amount and kinds of checking by annotating the code with pragmas
- trade-offs among missed errors, false alarms, checking time, effort to learn and use the pragmas
- demo

Java PathFinder

- a model checker for Java bytecode
- basically a virtual machine, executing the program theoretically in all possible ways
- check for property violations along all potential executions
- when a violation detected, an execution leading to it reported
- download

  `http://javapathfinder.sourceforge.net/doc/Obtaining_and_Installing_JP`
- details

  `http://ti.arc.nasa.gov/people/wvisser/ase00FinalJournal.pdf`

Java PathFinder

- first version based on Spin (Promela)
- current own-made model checker consisting of a JVM
- support all Java bytecodes, pure Java code can be analyzed
- CANNOT handle platform specific, native code
- check for deadlocks, invariants and user-defined assertions
- size limited around 10k lines of code
- demo

KeY

- a theorem prover for interactive and automated verification
- target language: JAVA CARD, a proper subset of Java, for smart card and embedded system
- use a dynamic logic for JAVA CARD (JavaDL) to express proof obligations
- support UML/OCL, JML
- download: `http://www.key-project.org/download/#key`
- details: `http://i12www.ira.uka.de/%7Ekey/doc/2005/sosym.pdf`
- tutorial for JML:

  `http://www.key-project.org/download/quicktour/quicktourJML-1.0.pdf`

KeY

- automatic proving: a collection of rules determined by Proof Search Strategy in advance
- interactive proving: apply one of the applicable rules to the focus term
- optional decision procedures: Simplify, ICS, CVCLite, SVC, Yices, SMT Translation
- DO NOT support full verification of a program
- instead generate proof obligations to prove selected properties
- demo

Caduceus

- a verification tool for C programs
- JML-like annotations: “The specification language is largely inspired by the JML. It has however significant difference, mainly due to the fact that unlike JML, we do not seek runtime assertion checking”
- generate proof obligations using Why tool
- a general theorem prover is used to establish the proof obligations
- download: `http://why.lri.fr/#download`
- tutorial: `http://caduceus.lri.fr/manual/caduceus.ps`
- details: `http://www.lri.fr/~filliatr/ftp/publis/caduceus.ps.gz`

Caduceus

- actually a compiler from annotated C programs to the Why input language
- Why: a verification condition generator(VCG) back-end for other verification tools
- details about Why: `http://why.lri.fr/index.en.html`
- theorem prover supported: Coq, PVS, Isabelle/HOL
- decision procedure supported: Simplify, Ergo, Yices, CVC Lite
- interaction with the theorem prover or the decision procedure may be necessary
- demo