CS453: Software Verification Techniques

Moonzoo Kim
Provable Software Laboratory
CS Dept. KAIST
Role of S/W: Increased in Everywhere

자료출처: Watts Humphrey 2002
Motivation: Poor Quality of SW
PROVABLE SW LAB
• **SW developers have to follow** scientific disciplines **for building and analyzing software with high quality**
  – This class focuses on the analysis activities
Software Development Cycle

• A practical end-to-end formal framework for software development
• **Embedded systems where highly reliable SW technology is a key to the success**
  - The portion of SW in commercial embedded devices increases continuously
  - More than 50% of development time is spent on SW testing and debugging
How to Improve the Quality of SW

1. Systematic testing
   - Coverage criteria
   - Mutation analysis

2. Debugging through automated analysis tools
   - Scientific treatment of SW with computing power
   - Useful tools are available

3. Formal verification
   - Guarantee the absence of bugs!!!
Questions???

• Is formal analysis really feasible in industry?
  – Yes, several case studies even in Korea

• Is formal analysis academically significant?
  – Yes, 3 Turing awardees in ‘07

• Is formal analysis too hard to learn and use?
  – No, there are tools available
Verification of High-Availability Protocol

- We develop a formal model of high-availability protocol used in commercial security appliances
  - HA protocol coordinates a group of firewalls
- We found several problems in HA regarding a master election procedure
Home Service Robot

- Designed for providing various services to human user
  - Service areas: home security, patient caring, cleaning, etc.
• Each memory cell can be written limited number of times only

• XIP by emulating NOR interface through demand-paging scheme

• Performance enhancement
Research Trends toward Quality Systems

• Academic research on developing embedded systems has reached stable stage
  – just adding a new function to a target system is not considered as an academic contribution anymore

• Research focus has moved on to the quality of the systems from the mere functionalities of the systems
  – Energy efficient design, ez-maintenance, dynamic configuration, etc

• Software reliability is one of the highly pursued qualities
  – NSDI 2007 Best paper
      – Heuristic application of model checking to detect liveness bug
  – OSDI 2004 Best paper
    • “Using Model Checking to Find Serious File System Errors” @ Stanford
      – Application of software model checking to find FS bugs
Formal Verification as a Foundational and Promising CS Research

• 2007 ACM Turing Awardees
  – Prof. Edmund Clarke
  – Dr. Joseph Sipfakis
  – Prof. E. Allen Emerson

• For the contribution of migrating from pure research to industrial reality

• One of the four Microsoft Research main areas
Tool-based Interactive Learning

- **Model checker**
  - Explicit model checker: [Spin home page](#)
  - Symbolic model checker: [NuSMV home page](#)
- **Software model checker**
  - Bounded model checker for C program: [CBMC home page](#)
  - Predicate abstraction for C program: [BLAST home page](#)
- **Satisfiability solver**
  - [MiniSAT home page](#)
- **Satisfiability Module Solver**
  - [Yices home page](#)
  - [Z3 home page](#)
- **Concolic testing tools**
  - [CREST home page](#)
- **Formal proof**
  - [WHY home page](#)
Class Schedule

- wk1: overview on formal SW analysis techniques
- Wk2-3: conventional testing techniques
- wk4: background on Propositional logic and SAT (Satisfiability) solvers
- wk5: SAT solver heuristic and tool application 1: MiniSAT
- wk6: background on First order logic
- wk7: Satisfiability Modulo Theory (SMT) basic
- wk8: midterm exam
- wk9: advanced application of SMT solvers
- wk10: directed automated random testing
- wk11: tool application: CREST
- wk12: basic temporal logic for requirement property
- Wk13-14: tool application: Spin & NuSMV
- wk15: state space minimization techniques
- wk16: final exam
Administrative Stuff

- Instructor: Prof. Moonzoo Kim
- Class time: Tue/Thr 9:00 -10:30 AM
- Office hour: TBD
- Grade policy
  - HW 50%
  - Attendance & quiz 20%
  - Mid exam 15%
  - Final exam 15%
- TA: Youngjoo Kim (Rm#2438)
- Web page: http://pswlab.kaist.ac.kr
Final Remarks

• For undergraduate students:
  – Highly recommend URP studies or independent studies
    • Ex. 이준희 (05학번) got a silver award and macbook air notebook 😊
      – Debugging Linux kernel through model checking to detect concurrency bugs
    • Ex2. Nam Dang wrote down a paper on distributed concolic testing
      – Y.Kim, M.Kim, N.Dang, Scalable Distributed Concolic Testing: a Case Study on a Flash Storage Platform, Verified Software Track @ Intl. Conf. on Theoretical Aspects of Computing (ICTAC), Aug 2010
Final Remarks

• For graduate students:
  – Welcome research discussions to apply formal analysis techniques
    • Systematically debugging C programs
    • Concurrency bug detection
    • Model-based testing
    • Prove the correctness of algorithms, etc