CSE40514: Program Verification

2025 Fall

Woosuk Lee

Instructor

Woosuk Lee

- Associate Professor, Hanyang University
- Ph.D from Seoul Nat'l Univ.
- Postdoc at UPenn and Georgia Tech
- **Expertise**: Programming languages, program analysis, automated program generation
- **Office**: Rm 403, 3rd Engineering Building
- **Office** hour: Mon/Wed 10:30 - 12:00 AM

Time & Place

- Mon/Wed 9:00 - 10:15 AM
- Location
  - Rm 106 4th Engineering Building (Y304-0106)
- Course website: https://psl.hanyang.ac.kr/courses/cse4051_2025f

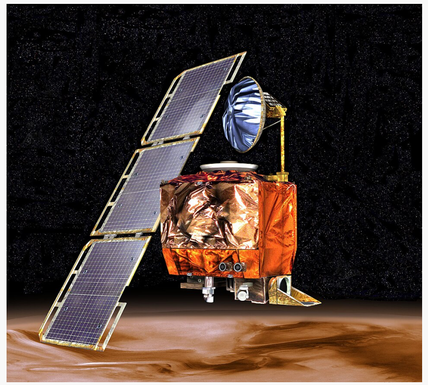
Why Learn?

- This course is about how to verify software will behave as expected.

- “Verify”: formally, (semi-)automatically, mathematically

- Why should we care correctness of SW?
  - Software is everywhere
  - Mission critical software: safety, security, reliability
  - Bugs can cause serious problems: financial losses, loss of life, etc

Software Bugs Can Be Catastrophic



- (1996) The arian-5 rocket exploded 37s after launch due to SW error (integer overflow).
- (1998) NASA's Mars climate orbiter lost in space due to SW error (wrong unit conversion).
- (2012) $440 million loss due to error in trading software of Knight Capital (functional bug).
- (2014) Heartbleed bug in OpenSSL library exposed sensitive data of millions of users (buffer overflow).
- … countless software errors in history

Topics

- (**Part1) Computational logic**: the use of logic to perform or reason about computation
  - Propositional logic
  - First-order logic
- (**Part2) SAT/SMT solvers:** how to use tools to check satisfiability of logical formulas and their internals
- (**Part3) Program verification**: proving correctness of programs using logical reasoning
  - How to specify desired properties?
  - How to prove properties hold minimizing human effort

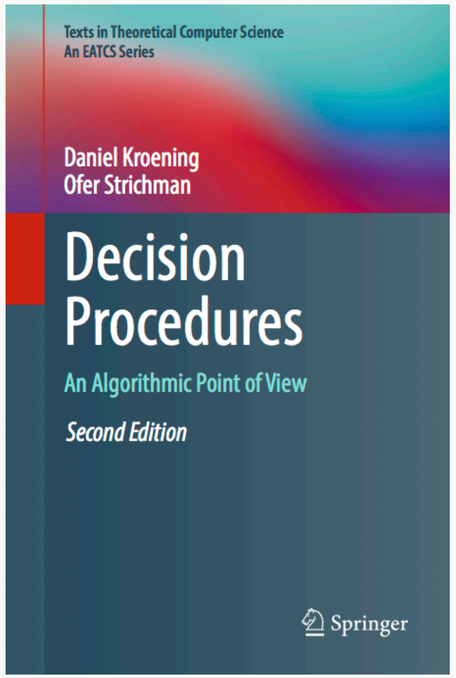
Course Schedule (Tentative) (contd.)

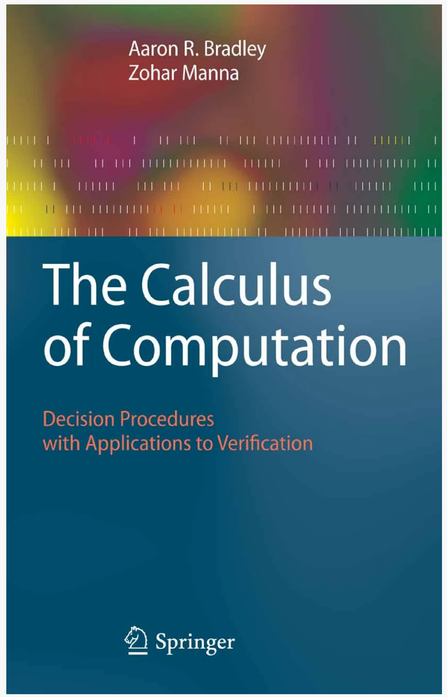
- Propositional logic (Part 1)
- Applications of SAT solvers (Part 2)
- CDCL algorithm (the core algorithm of SAT solvers) (Part 2)
- First-order logic (Part 1)
- First-order theories (Part 1)
- Applications of SMT solvers (Part 2)
- Overview of theory solvers (Part 2)
- Combining multiple theories (Part 2)
- DPLL(T) framework (the core algorithm of SMT solvers) (Part 2)

Course Schedule (Tentative)

- Hoare logic (Part 3)
- Verification conditions (Part 3)
- Proving partial correctness (Part 3)
- Proving total correctness (Part 3)
- Introduction to Dafny (Part 3)

Course Materials





- Aaron Bradley and Zohar Manna, The Calculus of Computation
- Daniel Kroening and Ofer Strichman, Decision Procedures: An Algorithmic Point of View
- Electronic versions of the books are available for download at lib.hanyang.ac.kr

Related Courses

- UT Austin, Automated Logical Reasoning by Prof. Işıl Dillig
- U of Washington, Computer-Aided Reasoning for Software by Prof. Emina Torlak
- Korea Univ. Computational Logic by Prof. Hakjoo Oh

Grading

- Homework — 16%
  - 4-5 assignments
  - Late submissions will get penalty
- Mid exam — 37%
- Final exam — 37%
- Attendance — 10%

Assignments (Tentative)

- Writing down proofs of some theorems
- Problem solving using SAT/SMT solvers
- Building your own SAT solvers (your grades will be based on the performance ranking of your implementations)
- Verifying properties of some algorithms using Dafny

Policy

- All programming assignments must be done individually.
  - Discussions with classmates are allowed, but you must write your own code.
  - No sharing of code or solutions.
- Code clone detectors will be used. Any detected plagiarism will result in a zero for the entire assignment. Also, your grade will be lowered by one letter.
- Using LLMs is allowed at your own risk of plagiarism (i.e., if you submit code generated by LLMs and it is detected as plagiarism, you will receive the penalty).