Model Checking and System Verification

Instructor: Mandayam Srivas
TA: Zubin Duggal

Aug-Nov 2020

Techniques for ensuring that a reactive computing (hardware + software) system is free of bugs are of utmost importance since vulnerabilities in such systems when embedded in devices can often cause huge damage in real life. Hence there is tremendous interest within computer science community and industry to find methods that can guarantee absence of bugs in systems when compared to traditional testing and simulation, which are able to only show presence of bugs. Model Checking is one such promising automatic method which applies foundational concepts and techniques from symbolic logic and automata theory to formally verify a model of a system satisfies a desired property for all possible behaviors of the system. Since its path-breaking Turing-award cited invention in 1986, several technical break-throughs have made model checking a viable technology for design validation in industrial practice while enhancements to it is continuing to be pursued as an active research area. This course covers both the theory and techniques of model checking at advanced undergraduate and beginning graduate levels. The course gives special emphasis on symbolic model checking with SAT solvers which is a most widely used version of it in practice and also forms the current research direction in this field. The course will also give a hands-on exposure to a few state-of-art model checking tools.

Learning Outcomes:

- The basic theory of model checking in propositional temporal logics
- Symbolic Model checking using SAT solvers
- Abstraction and Inductive Techniques to scale model checking
- Practical exercises on symbolic model checking tools

Targeted Audience:

- People that want to pursue formal verification as an industrial career
- People that are interested in learning about an exciting applied field of logic and automata
- People that want to pursue a research career in formal verification and validation

Desired Background: Logic, Theory of computation, Algorithms

Course Outline:

1. Modeling Systems (Kripke Structures)
2. Specifying Properties (Temporal logics: CTL, LTL, CTL*)
3. Verifying Properties (Model Checking for Temporal logics)
4. Abstraction Techniques
5. Automata-Based Model Checking

Course Work and Grade Distribution

1. Written Assignments: 30%
2. Tool Exercises: 30%
3. 2 Quizzes: 40%