

CS 315: Automated Verification

Instructor: Sherif Abdelwahed

Introduction

Several notations and methods have been developed to help the designer specify clear and unambiguous system requirements, verify that the requirements are consistent and correct, and verify that the refined design meets its specification. However, these methods are time-consuming and error-prone, and can be applied more effectively if there are tools to check their correctness. The goal of the course is to emphasize formal notations and methods that have tool support. We will cover the basis of underlying theory for the tools. The course will use material from the following references:

- M. Huth and M. Ryan, *Logic in Computer Science: Modelling and Reasoning about Systems*, Second edition, Cambridge University press, 2005.
- E. Clark, O. Grumberg, and D. Peled, *Model Checking*, MIT press, 1999.

Prerequisite

Experience with formal methods, although helpful, is not necessary. However, the course assumes familiarity with basic computer science concepts: relations, functions, graph theory, and finite-state machines.

Syllabus

- 1 System Verification
 - Definitions and motivation
 - Methods: Simulation, Testing, Formal Verification, Model Checking, Automated Theorem Proving
 - Industrial Case Studies
- 2 Introduction to Propositional and Predicate Logic
 - Propositional Logic: Natural Deduction, Semantics of PL, Normal Forms
 - Predicate Logic: PL as a formal language, Proof theory of PL, Semantics of PL, Undecidability of PL
- 3 Model Checking Linear Temporal Logic
 - Syntax and semantics of PLTL
 - Specifying properties in PLTL
 - Basic model checking scheme for PLTL
 - The model checker SPIN
- 4 Model Checking Branching Temporal Logic
 - Syntax and Semantics of CTL
 - Expressiveness of CTL, CTL*, PLTL
 - Specifying properties in CTL
 - Model checking in CTL
 - Fixed point characterization of CTL formulas
 - Fairness
 - The model checker SMV
- 5 Model Checking Real-Time Temporal Logic
 - Syntax and Semantic of Timed Automata
 - Syntax and Semantics of TCTL
 - Specifying timeliness properties in TCTL
 - The model Checker UPPAL
- 6 State Space Reduction Techniques
 - Binary decision diagrams
 - Other reduction techniques; partial order, memory managements, symmetry, bisimulation and equivalences

Grades

To encourage hands-on experience, there will be four verification and specification assignments involving prototype-quality verification CASE tools. In addition, each student will have to complete a research project to specify and verify a larger example. Reports on the projects will be written up, and results will be presented in class.

- | | |
|------------------------|-----|
| • Assignments (6) | 60% |
| • Project | 30% |
| • Project presentation | 10% |

Credit hours

Students will receive 3 hours of credit for this course.