STUDENT-LED COLLOQUIUM PROPOSAL: COMPUTER THEOREM PROVING

BERKELEY CHURCHILL, JARED ROESCH, ADELBERT CHANG

1. Purpose

The purpose of this course is to introduce computer-aided theorem proving in the context of software verification and the theoretical study of programing languages. The course will center around learning to use the theorem prover coq. Ultimately, students will learn how to prove statements about computer programs as mathematical objects. Along the way, topics covered will include functional programing; logical systems (classical, constructive, Hoare); the Curry-Howard correspondence; syntax; and semantics. The course will use the online textbook, "Software Foundations" by Benjamin Pierce.

The first three weeks will cover functional programing in Haskell, which is a prerequisite to theorem proving, and will be led by Jared Roesch and Adelbert Chang. The following seven weeks will cover theorem proving in coq, and will be led by Berkeley Churchill. Weekly homework assignments will be provided throughout the colloquium series.

The prerequisite for taking this course is basic knowledge of a programing language and discrete mathematics.

2. Tentative Schedule

Week 1. Functional programing in Haskell (I)
Week 2. Functional programing in Haskell (II)
Week 3. Functional programing in Haskell (III)
Week 4. Introduction to coq, generalizing induction. Homework: "Basics", "Lists".
Week 5. Type systems and polymorphism. Homework: "Lists" (continued) and "Poly"
Week 6. Constructionist versus classical logic. Homework: "Gen", "Prop".
Week 7. Curry-Howard Correspondence. Homework: "Prop" (continued) and "Logic".
Week 8. Big-Step semantics and proofs. Homework: "Imp" and "ImpList".
Week 9. Hoare Logic. Homework: "Hoare", "HoareAsLogic".
Week 10. Proving theorems in mathematics.

The homework above corresponds to chapters in the textbook.