CS 510 Computer-Assisted Theorem Proving – Spring 2013

Instructor: Andrew Tolmach
120-23 FAB (503) 725-5492
email: apt@cs.pdx.edu
Office Hours: Friday 2-3pm & by appt.
Course web page: http://www.cs.pdx.edu/~apt/cs510coq

Description

The course will be a hands-on introduction to interactive theorem proving using the Coq proof assistant. A proof assistant doesn't prove things itself; rather, it helps a human construct a proof more easily and reliably. We will begin by learning how to verify properties of simple functional programs, and then progress to proofs about other formal systems, such as programming language semantics. We'll also examine verification of imperative programs in some depth. If time permits, we will take a brief look at fully automated theorem provers, which work without direct human guidance.

Prerequisites

There are no formal course prerequisites, but prior exposure to basic discrete mathematics and formal proof is essential. Undergraduates should have completed CS250, CS251 and preferably CS311. Prior exposure to functional programming (e.g. CS457/557 or CS558) will also be very useful.

Readings

There is no published textbook, but for the first part of the course, we will be using the on-line textbook Software Foundations, by Benjamin Pierce et al. This book is under continuous revision; the version we are using will be made available on the course web page as we proceed.

Requirements

There will be weekly homework exercises and a take-home midterm and final.

The course grade will be distributed as follows:

Homework	20%
Midterm	35%
Final	45%

Although it will not be formally assessed, class participation is strongly encouraged, and may affect borderline grades.

Computing Facilities

The course will require use of the Coq proof assistant. While the precise version we use doesn't matter too much, it will be easiest and safest if we all use the *same* version, namely version 8.4pl1. This is (or will be) available on the CS linux lab machines as package `coq`. You will probably also want to install it yourself on your own machine, particularly if you have a laptop; see the course web page for pointers.

To interact with Coq, an IDE is a must; there are two choices of roughly equal quality. If you are an emacs user (or would like to become one), try the ProofGeneral environment (a general-purpose theorem proving IDE that has a Coq binding). Otherwise, you can use CoqIDE, which runs as a stand-alone tool on linux, MacOS, and Windows. See the course web page for pointers.

Windows Warning: CoqIDE 8.4pl1 on Windows is known to have a serious bug that makes it impossible to interrupt Coq if it goes into a long or infinite loop (which unfortunately does happen). If you must use Windows, you are advised to use emacs and ProofGeneral, or else use coq8.3.

Mac Warning: the Coq downloads site only provides a Mac package for version 8.4. Although full sources are available for 8.4pl1, the build is slightly tricky for plain coq, and definitely tricky for CoqIDE. Using the 8.4 version is probably adequate.

Individual Work

It is permitted (even encouraged) for you to work together on homework assignments. However, all homeworks must be prepared and submitted individually; the whole goal of the course is for you to become comfortable using the prover yourself.

Exams must be completed individually without any collaboration. Plagiarism or collaborating on an exam will result in an automatic zero grade and the initiation of disciplinary action at the University level.

Disabilities

If you are a student with a disability in need of academic accommodations, you should register with Disability Services for Students and notify the instructor immediately to arrange for support services.

Tentative Schedule

This schedule is highly subject to change.

dates	*topics*
Apr 1 & 3	Introduction to Coq; Simple functional programs and proofs
Apr 8 & 10	Lists, polymorphism, higher-order functions
Apr 15 & 17	More Coq tactics; Propositions
Apr 22 & 24	More propositions; Logical connectives
Apr 29 & May 1	Proofs and computations; Coq automation
May 6	**(take-home midterm due)** Case study: Imperative Languages
May 8	Language Metatheory
May 13 & 15	Hoare Logic
May 20 & 22	Heavy-duty automation
May 27	**No Class** Memorial Day
May 29	Dependent types in programming and proof
Jun 3 & 5	Coq in context
Jun 12	**Take-home final exam due**