CS 4/510 Theorem Proving and Program Verification – Winter 2021

Instructor: Andrew Tolmach
email: tolmach@pdx.edu
Course web page: http://web.cecs.pdx.edu/~apt/cs510coq

Description

Program verification is used to prove that programs will behave properly for all possible inputs and execution paths (unlike testing, which can only show correct behavior for particular inputs and executions). The past decade has seen enormous advances in the verification of important software infrastructure, including realistic compilers and operating systems.

This course will be a hands-on introduction to interactive theorem proving and program verification 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. If you like solving puzzles, using a proof assistant is fun and can be addictive!

We will begin by learning how to verify properties of very simple functional programs, and then progress to proofs about more interesting functional programs, imperative programs and other formal systems. If time permits, we will take a brief look at fully automated theorem provers, which work without direct human guidance.

Goals

Upon the successful completion of this class, you will be able to:

- Prove facts about elementary logic, arithmetic, and correctness of functional programs using the Coq proof assistant.
- Develop formal specifications for program behavior.
- Make fluent use of a core set of Coq tactics.
- Prove simple facts about imperative programs using a language-specific program logic.

Prerequisites

- Required:
  - CS250 and 251 Discrete Structures I and II or equivalent background in elementary discrete math and logic
- Strongly recommended:
  - CS350 Algorithms and Complexity or equivalent
  - CS320 Principles of Programming Languages or CS558 Programming Languages or equivalent
- Useful:
  - CS457/557 Functional Programming

Readings

There is no published textbook, but for the bulk of 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, a midterm and a final.

The course grade will be distributed as follows:

40%	Homework
25%	Midterm (take-home)
35%	Final (take-home)

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 8.12.2. 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 laptop; see the course web page for pointers.

To interact with Coq, an IDE is a must. There are three main choices:

- 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).
- If you are a VSCode user (or would like to become one), there is a VSCoq environment available. (Note that vim keybindings are available for VSCode.) This environment may be a bit buggy.
- Otherwise, you can use CoqIDE, which runs as a stand-alone tool on linux, MacOS, and Windows. The editing support is a bit primitive, and configuration can be a bit tricky on some platforms.

The course web page has pointers for all these.

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 proof assistant 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.

Title IX Reporting Obligations

Portland State is committed to fostering a safe, productive learning environment. Title IX and our school policy prohibit gender or sex-based discrimination and sexual misconduct (including harassment, domestic and dating violence, sexual assault, and stalking). We expect a culture of professionalism and mutual respect in our department and class. You may report any incident of discrimination or discriminatory harassment, including sexual harassment, to either the Office of Equity and Compliance (`https://www.pdx.edu/diversity/equity-compliance`) or the Office of the Dean of Student Life (`https://www.pdx.edu/student-life/dean-of-student-life`).

Please be aware that members of the faculty have the responsibility to report any instances of sexual harassment, sexual violence and/or other forms of prohibited discrimination to PSU's Title IX Coordinator, the Office of Equity and Compliance or the Dean of Student Life and **cannot keep information confidential**. If you would rather share information about sexual harassment or sexual violence to a confidential employee who does not have this reporting responsibility, you can contact a confidential advocate at 503-725-5672 or by scheduling on-line (`https://psuwrc.youcanbook.me`) or another confidential employee found on the sexual misconduct resource webpage (`https://www.pdx.edu/sexual-assault/get-help`).

Recording

We will use technology for virtual meetings and recordings in this course. Our use of such technology is governed by FERPA, the Acceptable Use Policy and PSU's Student Code of Conduct. A record of all meetings and recordings is kept and stored by PSU, in accordance with the Acceptable Use Policy and FERPA. Your instructor will not share recordings of your class activities outside of course participants, which include your fellow students, TAs/GAs/Mentors, and any guest faculty or community based learning partners that we may engage with. You may not share recordings outside of this course. Doing so may result in disciplinary action.

Tentative Schedule

This schedule is highly subject to change.

dates	*topics*
Jan 5 & 7	Introduction to Coq; Simple functional programs and proofs
Jan 12 & 14	Induction over numbers and data structures
Jan 19 & 21	Polymorphism; More Coq tactics
Jan 26 & 28	Coq's logic in depth
Feb 2 & 4	Inductive Propositions
Feb 9	**Take-home Midterm** (no class)
Feb 11	Proofs are programs: the Curry-Howard isomorphism
Feb 16 & 18	More automation; Verifying sorting algorithms
Feb 23 & 25	Verifying search algorithms and abstract data types
Mar 2 & 4	Hoare logic for imperative programs
Mar 9 & 11	Fully automated program provers
Dafny Mar 15–19	**Final Exam Week**