Formal Proof and Verification (CSCI 1951X)

Robert Y. Lewis

1 Basic Info

- Fall 2024
- MW 9:30-10:50am
- Location: CIT 241
- Website: `https://BrownCS1951x.github.io`
- Instructor: Dr. Robert Y. Lewis
  - Email: robert_lewis@brown.edu
  - Website: `https://robertylewis.com`
  - Office: CIT 433
- TAs: cs1951xtas@lists.brown.edu
  - Komron Aripov (HTA)
  - Chloe Qiao

2 Brief Description

Proof assistants are tools that are used to check the correctness of programs. Unlike tools like model checkers and SAT solvers, proof assistants are highly interactive. Machine-checked formal proofs lead to trustworthy programs and fully specified reliable mathematics.

This course introduces students to the theory and use of proof assistants, using the system Lean. We will use Lean to verify properties of functional programs and theorems from pure mathematics. We will learn the theory of deductive reasoning and the logic that these tools are based on.

3 Extended Description

When we learn to program, we often think in terms of implementing instructions: a program is a list of steps the computer should take, in order to do what we want it to do. But there are important ideas missing from this picture. How do we *specify* precisely what we want the computer to do? And how do we *verify* that our instructions match this specification?

Languages have been developed that offer a unified framework for specification, implementation, and verification. This course is about the theory and use of these languages, focusing on one called Lean, a new language developed at Microsoft Research.

Lean is known as a "proof assistant." The kind of verification seen in most proof assistants is not probabilistic, as in writing large test suites. It is not refutational, as commonly seen with model checkers and similar tools. Nor is it always automated. Part of the programmer's task is to build a *derivation*, or proof, that formally guarantees their program meets their specification.

Because of this focus on proof, a system like Lean blurs the line between programming and mathematics. One can write down and prove mathematical theorems in the same way one writes down and verifies program specifications.

This course will cover a variety of topics related to formal proof and verification. We will look at the theory of deductive reasoning, and how it is realized in the type theory of Lean; paradigms from functional programming that make verification easier; automatic generation of proofs; and applications of these tools in pure mathematics. But most of all, we will learn to *use* Lean to write trustworthy, verified functional programs. This course puts the theory of deductive reasoning into practice.

This course is based off of a course called Logical Verification, taught at the Vrije Universiteit Amsterdam:

`https://lean-forward.github.io/logical-verification/2021/index.html`

4 Classroom Setting

This is a friendly and collaborative class! Please be kind and respectful to your fellow students, and talk and get to know each other. We want to hear from everyone throughout this course. If at any point you feel uncomfortable or unwelcome, we encourage you to get in touch (directly or via the anonymous feedback form on our website) so that we can make things right.

Please be aware of your actions and understand that your classmates come from many different backgrounds. Your perspective on the course or assignments may not match theirs, and comments that (unintentionally) dismiss their perspectives can be very discouraging.

Professionalism goes both ways: we ask you to treat your classmates and course staff as adults, and you should expect the same from us. If for any reason you feel uncomfortable coming to Rob or the TAs about an issue, you are welcome to contact the CS department Diversity and Inclusion committee, the department chair (Roberto

Tamassia), or the Brown Title IX office. As a department we are committed to taking seriously all complaints about unprofessional or discriminatory behavior.

5 Course Objectives

The main objective of this course is to learn how to specify and verify properties of programs and mathematical objects in type theory. You will learn to use the Lean proof assistant to implement functional programs, state their important properties, and verify that these properties hold. At the same time, you will learn the theory underlying these reasoning systems: what deductive proofs consist of, and why they are trustworthy.

By the end of this course, you should be able to:

- Write executable functional programs in Lean.
- Specify and verify properties of these functional programs.
- Write and prove mathematical statements in Lean.
- Write *metaprograms*: tactics that automatically search for proofs.
- Explain the semantics of functional programs.
- Distinguish *proof objects* (derivations, deductions) from other kinds of verifications or tests for program correctness

6 Prereqs

We recommend students to have taken either CSCI 1710 Logic for Systems or a proof-based mathematics course. Basic familiarity with functional programming (e.g. Haskell, ML) is helpful but not required.

7 Textbooks

We will follow *The Hitchhiker's Guide to Logical Verification* by Jasmin Blanchette, et al. A version of this text is available on our course website.

8 Assignments and grading

- 70%: There will be one homework assignment per chapter of the Hitchhiker's Guide. This is approximately weekly, although some weeks will not have an assignment. Homework assignments will be released on Mondays before class, and due 8 days later on Tuesday nights.

- 30%: An individual final "formalization project" will give students the chance to implement and verify something of their own interest in Lean. We will make project ideas available in advance, and students are encouraged to consult with the instructor.

Students may submit assignments one or two days late, up to a total of six late days over the semester. For example, you could be one day late on six different assignments; two days late on three different assignments; or any combination in between.

Students with a university-approved reason (a letter from the SAS office, dean's office, or health services) may request a two-day extension that does not consume late days. Longer extensions will be considered on a case by case basis; a university-approved reason is required. A form to request an extension is linked from the course website.

Students with SAS accomodations should contact the instructor at the beginning of the semester to discuss how we can best address your needs.

Assignments will be submitted on Gradescope, linked to from the course website. If you are unfamiliar with Gradescope please talk to Rob and/or the TAs.

9 TA hours

Our talented TAs are here to help! We'll keep a schedule of TA hours posted on the course website. (Rob will also have office hours during the week.)

One TA session each week will be designated a *lab session*. Lab sessions will be more structured than normal TA hours: students attending will be able to collaborate on extra practice problems that will supplement the homework problems. These lab sessions are optional and ungraded.

10 Collaboration

Students are strongly encouraged to work together on the exercise problems and to discuss the homework assignments. Written homework submissions should be your own work. We advocate the "erased whiteboard" policy: talk through your solutions with a classmate, but do not reference any written notes from that conversation when you implement your solutions.

A good rule of thumb is that your discussions should remain at the conceptual level. Ask "what is the syntax for a proof by induction," instead of asking whether your proof by induction is correct.

Above all, students should adhere to Brown's academic code:
`https://www.brown.edu/academics/college/degree/policies/academic-code`

11 Attendance

In-person attendance is strongly encouraged but not required. If you are not feeling well, please do not attend class! Classes will be streamed and recorded, although active remote participation will not be possible.

12 Time requirements

Our class meets for 160 minutes per week. You can expect to spend roughly 12 hours per week on assignments, labs, and the final project, for a total of at least 180 hours over the course of the semester.

13 Feedback

We very strongly encourage feedback on all aspects of the course: the textbook, classes, exercises, homeworks, projects, and more! Please contact the course staff with any feedback—positive or negative, big or small—or use the anonymous report form here:
`https://forms.gle/hfrzAVC8e82P1n3o8`

14 Further resources

We recognize that being a student is not easy and wish to provide support however we can. Beyond our staff, here are some resources that are available to you here at Brown:

- Counseling and Psychological Services: `https://www.brown.edu/campus-life/support/counseling-and-psychological-services/home`
- Student Accessibility Services: `https://www.brown.edu/campus-life/support/accessibility-services/`
- Student Advocates for Diversity and Inclusion: `https://cs.brown.edu/about/diversity/student-advocates-diversity-and-inclusion/`
- CS Health and Wellness Resources: `https://docs.google.com/document/d/1pHLq14jNOGvutTYOsj` `edit`