Honors Thesis Project

John Rafael Matteo Munoz-Grenier

February 22, 2024

Contents

Introduction **v**
0.1 What is Lean? . . . v
0.2 Why use Lean? . . . v
0.3 Who else uses Lean? . . . vi

Development **vii**
0.4 Summer 2023 . . . vii
0.5 Fall 2023 . . . vii
0.6 Spring 2023 . . . viii
0.6.1 Finishing the library . . . viii
0.6.2 Insights as a TA . . . viii
0.6.3 Honors Thesis development . . . viii

Structure **ix**
0.7 Introduction to Lean and Logic . . . ix
0.8 Set Theory . . . ix
0.9 Topology . . . ix

Introduction

Most undergraduate Mathematics programs require students to take a proof-writing course, as proofs insert formal rigor into the study of Mathematics. Still, there's some degree of uncertainty, since human error is still involved in evaluating the proofs and determining their consistency. [Insert examples of theorems which have been erroneously proven throughout history]. For simpler proofs, the risk of human error is a lesser problem, since a discerning eye can quickly catch holes in a faulty proof. Nonetheless, grading the quality of a slew of proofs is time-consuming, especially when factoring in the time a grader might take to provide feedback for the specific errors in a poorly-written proof. The Lean Theorem prover offers a solution to the first problem, and this project is an effort to simplify the grading process of proofs by integrating Lean in a proof-writing course.

0.1 What is Lean?

Lean uses dependent type theory and the Curry-Howard isomorphism to build a programming language capable of defining and proving theorems in Mathematics. Although other theorem provers exist, Lean was chosen for this project due to its extensive math library, Mathlib, and its large, active community of users. Lean also has a "tactic mode," wherein all of the premises and goals of a proof are displayed [insert screenshots] as they develop throughout the proof, and functions called "tactics" can be used to advance through the proofs using automation and type inference. Lean's large user base and committed development team have also created a variety of supplemental resources for new users to learn Lean and for experienced users to reference, most notable of which is Mathematics In Lean.

0.2 Why use Lean?

1. Autograding! Any proof written in Lean which compiles is guaranteed to be correct.
2. Lean Infoview makes writing and reading proofs easier; the precise state of what is already proven and what remains to be shown is made explicit for every step of the

proof.

3. Tactics can automate some of the tedious parts of a proof, like unfurling definitions and other procedures which can be done algorithmically.
4. All of the essential parts of a proof are made explicit; hypotheses which go unused are automatically marked by Lean, and each use of some hypothesis or lemma is tracked by its variable name.

Downsides:

- Some parts of a proof which might be natural or trivial may be challenging to formalize.
- Appealing to some well-known lemma requires knowledge of its name in Lean.
- Lean proofs are not written like human language proofs, and don't directly develop a student's capacity to write readable human language proofs.
- Lean takes some time to learn, time which could instead be used for purely mathematical pursuits.
- Lean requires some level of tech savviness to access, since it must be downloaded and configured, and either the terminal or a code editor is necessary to write Lean code.

0.3 Who else uses Lean?

Heather Macbeth [Insert information about her course] [Insert other examples]

Development Process

This honors thesis project began in June 2023, stemming from an idea Dr. Sergey Cherkis had for automating the grading process in his topology class. Over the summer, I endeavored to learn Lean and become comfortable using Mathlib. In the Fall, I began building the library, and I finished in the Spring.

0.4 Summer 2023

Lean has many resources available to newcomers, so I began by tackling the introductory textbooks Mathematics In Lean and Theorem Proving In Lean. Mathematics In Lean is peppered with exercises for the reader, which I dutifully completed. When I read through MIL, it was still written for Lean3, and I only covered the sections introducing Lean itself, set theory, and topology.

The topology section of Mathematics in Lean was sparse, and took an approach to Topology centered in the notion of filters.

Definition 1 *A* **Filter** F *is a collection of sets over a space* X *such that*

1. *if* $S \in F$ *and* $\hat{S}$ *is a superset of* S*, then* $\hat{S} \in F$*, and*
2. *for any two sets* $S, T \in F$*, then the intersection* $S \cap T \in F$*.*

0.5 Fall 2023

In Late August, I began to work through the Munkres textbook "Topological Spaces" with a focus on identifying which parts of the textbook were more or less challenging to formalize. Starting in September, I pivoted to working on the first section of my instructional repository, Logic in Lean. I spent the first few weeks of september deciding how to structure the introduction to formal logic, which ultimately culminated in the current path starting with Propositions, truth and falsity, then leading through implication, disjunction, conjunction, negation, and the 2 quantifiers. I actually began writing code and comments in

late september, and by mid-october I had written files explaining proofs, proposition, and implication; true, false, introduction rules, and elimination rules; negation; conjunction and disjunction; and the existential and universal quantifiers. This was also the time when I began backing my files up on github. The remainder of the semester was spent continuing to flesh out the library, adding a section on set theory and a section on Topology.

0.6 Spring 2023

Over the Winter break, it was confirmed by the University of Arizona mathematics department that Dr. Cherkis would teach a graduate class in the coming spring, Math 529: Proof Writing and Proof Checking with a Computer. Furthermore, I was accepted as a Teaching Assistant for this class. This class gave me the opportunity to discern how approachable formalization of Mathematics in Lean is in an actual classroom environment. During this semester, I also began writing this Honors thesis, and I finished working on the code library which introduces students to Lean.

0.6.1 Finishing the library

0.6.2 Insights as a TA

For the first month and a half of the class, Dr. Cherkis lectured in parallel to the structure of Mathematics in Lean 4. I was taken aback by the difficulty students had in grasping the "apply" tactic, which transforms a tactic state with goal Q to a tactic state with goal P by using an implication $P \to Q$.

0.6.3 Honors Thesis development

Structure

0.7 Introduction to Lean and Logic

0.8 Set Theory

0.9 Topology