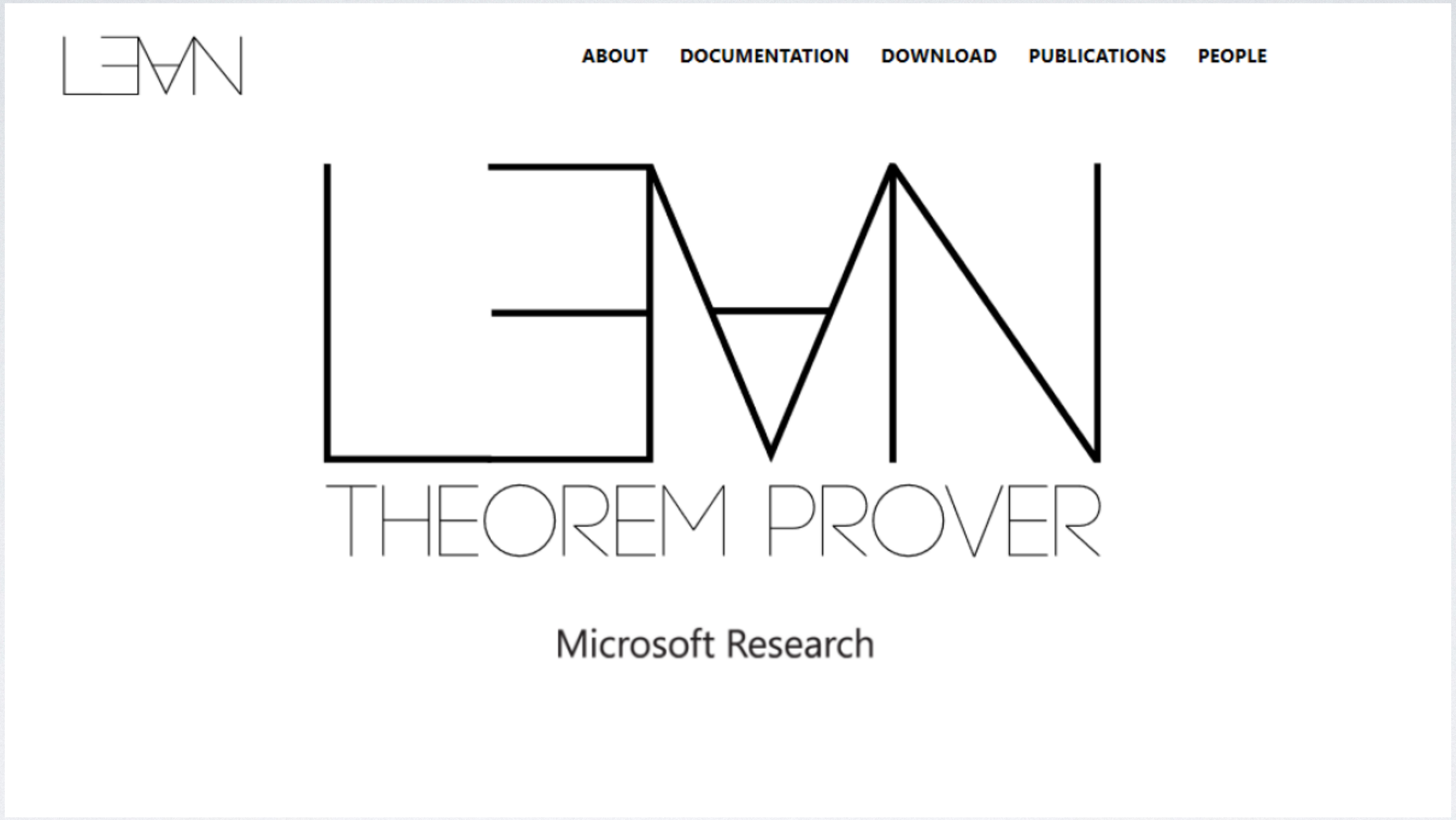
FORMAL METHOD IN TEACHING

INTRODUCTION TO PROOFS WITH LEAN

Sina Hazratpour

Johns Hopkins University


ABOUT
DOCUMENTATION
DOWNLOAD
PUBLICATIONS
PEOPLE
LEAN
THEOREM PROVER
Microsoft Research

(LEAN DEMO)

COURSE INFORMATION

- Department of Mathematics, Johns Hopkins
- Introduction to proofs (AS.110.301.S22)
- Undergraduate Course (4.00 Credits)
- No Math/CS prerequisites

FROM THE COURSE DESCRIPTION

A practical introduction to

- the **language of mathematics**
- the **methods of proof**

LANGUAGE & PROOF

- A language capable of talking about complicated mathematical objects and constructions
- Methods of writing proofs which are rigorous, readable, and elegant

PROOF AND UNDERSTANDING

- Humans are very capable to live with vagueness and ambiguity in thinking; computers are pedantic and cannot deal with ambiguity.
- Proofs are not only to show absolute correctness; they are devices to transfer human understanding and there are varieties of understanding.
- We think of a proof assistant as a tool; it will not replace human understanding, but like all human-made tools, such as telescope and microscope helps us to see further and in more details.

PREVIOUS OFFERING (IBL STYLE)

- Sina Hazratpour (Spring 2022, Fall 2021)
- Emily Riehl (Spring 2021, Spring 2019)
- Valentin Zakharevich (Spring 2020)

LECTURE/ASSESSMENT MODALITY

- Fall 2021: ***Pen&Paper***
- Spring 2022: ***50% Pen&Paper*** - ***50% Lean***
- Fall 2022: ***100% Lean***

MOTIVATION FOR ADDING LEAN

Why did I consider introducing a proof assistant in teaching this course?
(Fall 2021 → Spring 2022)

How empowering it was for many students to use logical reasoning as a natural way to formulate and think about mathematical problems (e.g. in metric space topology). I wanted to make this more fun like playing a game with Lean.

SPRING 2022

Lean only in the later half of the course.

Students checked the correctness of their proofs about metric spaces in Lean.

MOTIVATION FOR TEACHING FULLY WITH LEAN (SPRING 2022 $\rightarrow$ FALL 2022)

- *The interaction with Lean did not facilitate the creation of proofs (by that point of the course, students were already writing pen-and-paper proofs)*
- *Not enough momentum for the majority of students to be enthusiastic about Lean.*
- *Not fun for slower students and those without coding background*
- *Repetitions (e.g. natural deductions and Lean proof tactics) & a lot of uphills*
- *Students got lost in vast libraries of Lean trying to find lemmas.*

FALL 2022

Main goal was

to help students achieve a deeper understanding of pen-and-paper proofs by formalizing them.

Which was facilitated by

getting instant feedback from Lean to help students with completing their proofs.

THE SYLLABUS

Introduction to Lean Prover

Definitions, Examples, and Theorems in Lean

Basic Algebraic Identities

Functions

Equality of Functions

Algebra of Functions

Logic of Propositions

Logic of Predicates

Injections and Surjection

Permutations

Bundled Structures

Unbundled Structures

Surjection-Injection Factorization of Functions

The Algebraic Hierarchy with Type Classes

Inductive Types (natural Numbers and lists)

Quotient Types (integers and rationals)

Basics of Categories

Functors

Natural Transformations

The Yoneda Lemma

SETTING

- course website: https://sinhp.github.io/teaching/2022-introduction-to-proofs-with-Lean
- One grad TA, one undergraduate CA
- Two 75-minute lecture (Mon, Wed) + a 75-minute section (Fri)
- 2 projectors, one for me, one for students to project their codes for answering challenges during the lecture
- Collaborative live-coding lectures
- Zulip as replacement for the course website for real-time debugging
  The best effect : students started helping each other on Zulip
- Formalization Hackathon instead of midterm exam
- Mini Projects instead of final exam

TEXTBOOK

- **No textbook; we started (almost) from scratch and built our own Lean library.**
- **Greatly inspired by Mathematics in Lean by Avigad, et al.**
- **Gave students greater sense of autonomy; they were the co-creators of their own learning material.**
- **Resulted in ProofLab code repo: https://github.com/sinhp/ProofLab/tree/main/src/lectures**

THE FLOW OF LECTURES

- The main underlying mathematical concept of each lecture (e.g. getting a partition from an equivalence relation)
- First informal ideas; sketching on the whiteboard some informal and non-rigorous forms and figures
- This is about 15-30 minutes depending on the inherent difficulty of the concept
- Start formalizing them together with students; first wait for students suggestions

ASSESSMENT

Students were assessed based on

- Homework assignments
- Active participation in class
- Active participation in the Zulip server of the course
- Performance in the hackathon contest
- Mini-project completion

FUNCTIONS BEFORE LOGIC

Most other Lean undergrad courses start with Logic.

```
-- Depenedent Modus Ponens (tactic style)
lemma dep_modus_ponens:
  P → (P → P → Q) → Q :=
begin
  -- we want to prove P → ((P →(P → Q)) → Q) so we
use intro rule of →
  intro h₁,
  -- we want to prove (P →(P → Q))  → Q so we use
intro rule of →
  intro h₂,
  --  we want to Q; we create a new subgoal P → Q,
and prove it using tactic `have`
  have h₃ : P → Q, from h₂ h₁,
-- we use `h₃` to prove `Q` by application
elimination.
  exact h₃ h₁,
end
```

We started with functions before logic.

```
/- ## Challenge -/
-- Depenedent Modus Ponens (term style)
example :
  P → (P → P → Q) → Q :=
sorry
```

- Benefit: after many tactic proofs to prove stuff in propositional logic, I could introduce term proofs via proposition-as-type paradigm.

TEACHING TACTICS

We wrote all the proofs using tactics; only very late in the course we used term proofs.

We used a lot of tactics.

Challenge: how to remember all these tactics?

Solution: many tactics (related to logic) correspond to step-by-step extension of logic (connectives, quantifiers, etc) — Introduce tactics in harmony with logic.

BASIC TACTICS

Introduction to Lean Prover

Definitions, Examples, and Theorems in Lean

Basic Algebraic Identities (`refl`, `exact`, `rw`, `change`, `calc`, `ring`)

Functions

Equality of Functions (`funext`)

Algebra of Functions (`dsimp`, `simp`)

Logic of Propositions (`split`, `cases`, `intro`, `apply`, `have`, `left`, `right`, `suffices`, `assumption`)

Logic of Predicates (`intro`, `apply`, `use`, `cases`)

Injections and Surjection (`unfold`, `obtain`, `rcases`)

Permutations (`let`)

Bundled Structures (`repeat`, `wlog`)

Unbundled Structures

Surjection-Injection Factorization of Functions

The Algebraic Hierarchy with Type Classes

Inductive Types (`induction`)

Quotient Types (integers and rationals)

Basics of Categories (`~~.obviously~~`)

Functors

Natural Transformations

The Yoneda Lemma *(Demo in Lean*)*

RISKS OF AUTOMATON TACTICS

Challenge: There is a real risk that students could do an "automatic" proof without actually understanding their "proof".

`Automation tactics: `simp`, `linarith`, `suggest`, `tauto`, `tauto!`, …`

Solution: Structured homework proofs *(Demo in Lean*)*

- Informal proof strategy (with numbered steps) before Lean proof
- Lean proof with comments (referring to the steps of proof strategy)

EXAMPLES

Proof assistants are designed for formalizing abstract math.

Challenge: They are Not as good for

- examples and counter-examples
- several instances of the same structure
- concrete computations

which are crucial to any undergraduate mathematics course.

Solution:

- ad-hoc type-classes (`has_add`, `has_mul`, `add_monoid`, `mult_monoid`, ...)
- merging proof assistants with computer algebra systems (e.g. Sage, Mathematica, c.f. `linear_combination`, `polyrith` Lean tactics by Rob Lewis and Heather Macbeth).

FOCUS ON MATH

I wanted students to focus on mathematical ideas.

- Did not overemphasize type vs sets.
- For exercises, included all the necessary lemmas and definitions.
  Library search: big distraction!
- Minimized discussion of universe levels in category theory.
- Gave as much help needed for the Lean part of homework problems.

WITH AND WITHOUT LEAN

Fall 2021(P&P, IBL) $\rightarrow$ **Spring 2022** $\rightarrow$ **Fall 2022 (Lean)**

Conservatively radical $\rightarrow$ **radically conservative**

FROM COURSE EVALUATIONS

“Thank you for giving me confidence that I could succeed in a math class at a very rigorous level. I hope to do more of that in the future. It was a great first step in to college-level mathematics.”

“It was enjoyable to learn a new programming skills and delve into why the mathematical concepts we learn about in class are true or built.”

“The best aspect of this class was the unique nature of doing assignments in Lean and our competition midterm. ”

FEEDBACK FROM COURSE ASSISTANT

“The obvious differences from a traditional section are the application of computer science skills. Because the course is primarily advertised as a mathematics course, the students do not necessarily have prerequisite experience with coding, and this can cause some difficulties. However, most students are able to overcome this easily.”

“Specifically with propositional logic, Lean can be very useful. Because the building blocks of propositional logic are fairly consistent, just with propositions, so there is less confusion. Lean offers the advantage of the context menu, which in propositional logic very clearly holds and displays the information available. The use of Lean tactics also allows the user to approach the proof from both directions very easily, as in proving directly or simplifying the goal”.

IMPROVEMENTS WITH LEAN
(COMPARED TO FALL 2021 P&P PROOF-BASED COURSE)

- **Covered twice as much**
- **Much better grades**
- **As part of writing proofs in Lean 100% of students could identify strategies for proving a statements based on its logical structure.**
- **Writing proof strategies and code comments improved students to write much better articulated arguments.**
- **Much improved ability in discerning the assumptions and conclusions and knowing where to start and where to end in constructing proofs.**
- **The classes were way more engaging and fun.**

RESOURCES

- *Avigad, et al.* Mathematics in Lean, https://leanprover-community.github.io/mathematics_in_lean/
- ProofLab, https://github.com/sinhp/ProofLab/
- Fall 2021 course website: https://introproofs.github.io/jhu301-f21/
- Spring 2022 course website: https://introproofs.github.io/s22/
- Fall 2022 course website: https://sinhp.github.io/teaching/2022-introduction-to-proofs-with-Lean