Theorem Proving and Lean

Bhavik Mehta

Bernoulli Center

September 9, 2024

- How we got here
- Introduction to Lean
- Writing simple proofs in Lean
- Writing difficult proofs in Lean

- History of automated reasoning
- Interactive proving and formal verification
- AI for mathematics
- Recently formalised combinatorics

Prehistory

- Principia Mathematica, Whitehead and Russell, 1910s, derive mathematical expressions from a minimal set of axioms
- Presburger arithmetic, 1929, natural numbers with addition and equality
- Gödel Incompleteness, 1931
- Logic Theorist, 1956, could prove around 38 of its first 52 theorems
- Automath, 1967, de Bruijn

Restricting the logic

- Propositional logic is decidable
- DPLL algorithm, SAT solvers
  - Boolean Pythagorean triples
  - Schur number
- First order logic is semi-decidable: you can sometimes make progress (Prover9, Vampire)
- Equational logic, eg Robbins conjecture. If $\vee$ is commutative and associative, does $\neg(\neg(a \vee b) \vee \neg(a \vee \neg b)) = a$ imply we have a Boolean algebra? Yes, EQP 1996.

Special-purpose algorithms

- Four colour theorem (1976) with 1834 configurations to check
- Kepler conjecture (2000s), 100,000 linear programs to check
- Is the Lorenz attractor strange? Interval arithmetic proof
- Rubik's cubes, Sudokus
- Computer algebra systems

General purpose

- Switch to *interactive* theorem proving in an expressive language
- Can formally verify much more, at the cost of less automation
- Many but not all systems inspired by Automath
- *All* have a ‘kernel’, small but trusted code-base

Some successes

- Four colour theorem, Coq, 2005
- Odd order theorem, Coq, 2012
- Kepler conjecture, HOL Light + Isabelle, 2014
- Independence of continuum hypothesis, Lean, 2019

Explaining and learning mathematics

- Allowing the reader to choose their level of detail: a dynamic proof document
- Can go all the way to the axioms, if you really want
- Can modify an explanation or statement and ensure nothing breaks
- Allows the writer to fix assumed background knowledge
- Allows the writer to focus on exposition, and let the formal version focus on rigour
- Easier to search?

Creating mathematics

- Change your definitions and see how the proof needs to change (optimise the constants more easily!)
- Which hypotheses are really needed? How do the lemmas really depend on each other?
- Encourages simplifying arguments: asymptotics for cap-set
- Encourages powerful abstractions

Collaboration and fun

- Easier to coordinate when you're precise
- Contribute with a lower barrier to entry
- Liquid Tensor Experiment (Commelin, Scholze)
- Polynomial Freiman-Ruzsa (Tao)
- Addictive video game

Correctness

- A formally verified proof is correct!
- No missing corner cases in your lemma
- The definition which is nice enough to prove Lemma 2 is strong enough to be useful in Theorem 6
- It's okay that the paper you reference uses different conventions and notation

Many more!

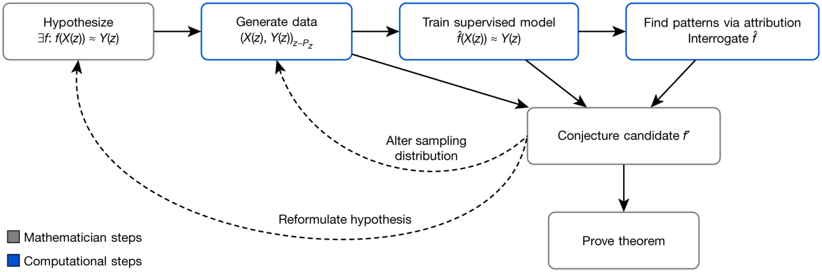
- *Why formalize mathematics?*, Patrick Massot,
- *Mathematics and the formal turn*, Jeremy Avigad

What about machine learning!

Computers for formulating conjectures

- Tables of primes were used to conjecture the prime number theorem
- Tables of elliptic curves were by Birch and Swinnerton-Dyer
- OEIS

Generating ideas


Hypothesize
$\exists f: f(X(z)) \approx Y(z)$
Generate data
$(X(z), Y(z))_{z \sim P_z}$
Train supervised model
$\hat{f}(X(z)) \approx Y(z)$
Find patterns via attribution
Interrogate $\hat{f}$
Conjecture candidate f'
Alter sampling distribution
Reformulate hypothesis
Prove theorem
Mathematician steps
Computational steps

Relate algebraic and geometric invariants in knot theory; new formula for Kazhdan-Lutzig polynomials in representation theory[1]

[1] Advancing mathematics by guiding human intuition with AI, Alex Davies et al.

Finding examples

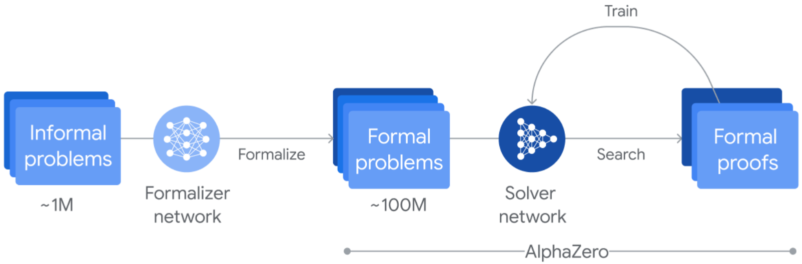
- Adam Wagner, reinforcement learning to find counterexamples to a number of graph theory conjectures[2]
- Romera-Paredes, et al, found large 3AP-free sets in F_3^n for fixed n (and these tensor)[3]

- But these are all discrete, finite objects!
- If only we had an expressive computer language for maths which can encode proofs and check them...

[2]Constructions in combinatorics via neural networks

[3]Mathematical discoveries from program search with large language models

AlphaProof



`http://dpmd.ai/imo-silver` This system fully solved, in Lean, 3 out of 5 IMO problems

What is Lean?

- Lean is an open source interactive proof assistant
- Written by Leonardo de Moura and his team at Microsoft Research
- Lean is a dependently-typed programming language
- Very young and very fast growing
- Emphasis on classical mathematics, and a single library mathlib
- A more concrete description coming up!

Additive combinatorics

- Cap-set problem (Dahmen et al)
- Solymosi sum-product $\frac{1}{5}|A|^{\frac{5}{4}} \leq \max |A+A|, |AA|$ (M.)
- Szemerédi's regularity lemma and $r_3(N) = o(N)$ (Dillies, M.)
- $N/\exp(4\sqrt{\log N}) \leq r_3(N) \leq N/\log\log N$ (M.)
- Plünnecke-Ruzsa inequality (Dillies)
- Erdős-Graham problem (Bloom, M.)

Highlights

- $R(k) \leq (4 - \varepsilon)^k$ (M.)
- Proved in March 2023, 57 page long paper, one person, formalised in 5 months

- Polynomial Freiman-Ruzsa (Tao)
- Proved in December 2023, 20 page long proof, around 30 people contributed, formalised in *three weeks*

But what really is Lean?

Aims:

- Understanding the language and basics of Lean
- How to actually use Lean to prove simple things
- How to use Lean for more practical things