Michigan Lean Study Group Welcome

2 February 2026

Slides available @
https://ghseeli.github.io/lean/lean_study_group_W26.html

What is formal verification?

- **Formal verification** proves (or refutes) that a system meets a **precise specification**.
- Two common approaches:
  - **Model checking:** algorithmically explore a (finite/abstracted) state space.
  - **Interactive theorem proving:** human-guided, machine-checked proofs with automation.
- Some history: From LCF to Isabelle/HOL (arXiv)

Why?

- In programming:
  - Higher assurance
  - Find deep bugs
  - Clarify designs
- In mathematics:
  - Verify proof is correct!
    - Refereeing
    - Large scale collaboration
    - Increasing trust (e.g., with LLMs)
    - Find hard to catch mistakes
  - Enable automated proofs.

Examples

Kevin Buzzard: Where is Mathematics Going? | Terry Tao's Blog | Leo de Moura: Bridging Formal Mathematics and Software Verification

An Example

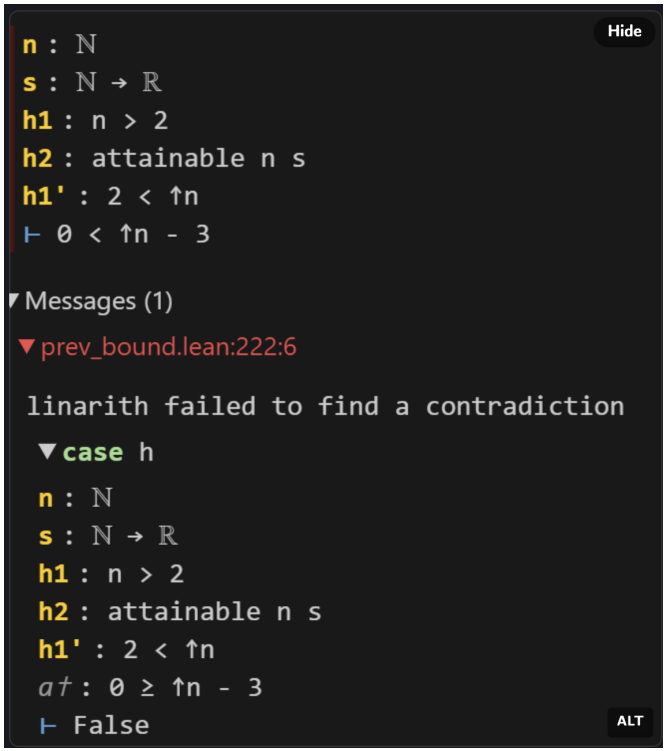
Terence Tao
@tao@mathstodon.xyz

As a consequence of my #Lean4 formalization project I have found a small (but non-trivial) bug in my paper! While in the course of formalizing the arguments in page 6 of arxiv.org/pdf/2310.05328.pdf , I discovered that the expression $\frac{1}{2}\log\frac{n-1}{n-k-1}$ that appears in those arguments actually diverges in the case $n = 3, k = 2$! Fortunately this is an issue that is only present for small values of n, for which one can argue directly (with a worse constant), so I can fix the argument by changing some of the numerical constants on this page (the arguments here still work fine for $n \geq 8$, and the small n case can be handled by cruder methods).

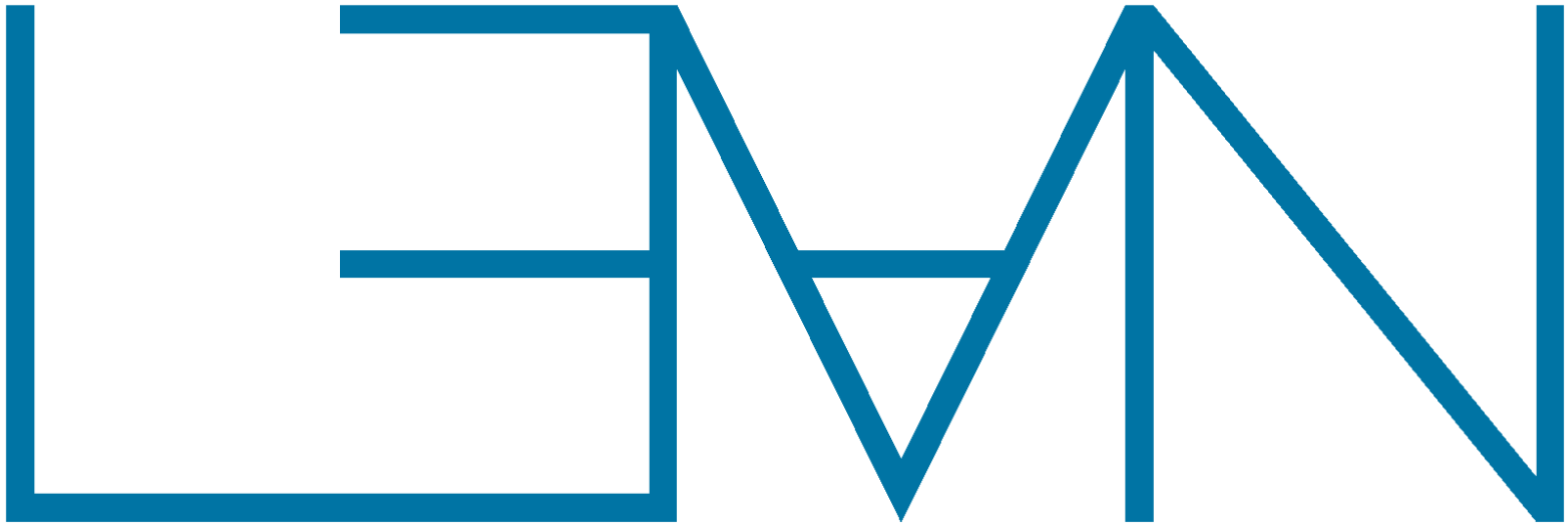
Enclosed is the specific point where the formalization failed; Lean asked me to establish $0 < n - 3$, but the hypothesis I had was only that $n > 2$, and so the "linarith" tactic could not obtain a contradiction from the negation of $0 < n - 3$.

I'll add a footnote in the new version to the effect that the argument in the previous version of the paper was slightly incorrect, as was discovered after trying to formalize it in Lean.

Mathstodon



Lean 4



An open-source functional programming language and interactive theorem prover.

Site | repo

Image from Leo de Moura: How the Lean language brings math to coding and coding to math

Lean

- Many formal verification / proof assistant systems have existed for a while (e.g., Roqc (Coq), HOL, ACL2, Isabelle,...)
- Lean 4 has become increasingly popular in mathematics.
- Lean is based on *Calculus of Constructions*, a version of *dependent type theory*.
- Lean is also a fully-featured functional programming language.
- Lean's mathlib contains an ever-growing list of theorems! (Current status of mathematical areas)

Notable Lean Projects

- mathlib4: site | repo (core library)
- Liquid Tensor Experiment: blueprint | repo | completion post
- Xena Project: site | GitHub
- Equational Theories Project: site | repo
- Fermat's Last Theorem Project: intro | repo
- See more here.

Our possible directions

- Rough plan for pre-spring break:
  - Work on exercises in small groups
  - Discuss exercises in small groups
  - Mini-lectures on anything someone wants to talk about (type theory? formalizations of objects in your fields? other topics?)
- Post-spring break goals
  - More mini-lectures
  - Work on bigger formalization projects (making objects, proving theorems about them)

Getting set up

Installations

- The Lean community has some options for how to run Lean (link). We recommend their VS code option.
- Additionally you will need to setup a GitHub.

Setting up a project

- The Lean community has instructions on starting a local or shared project (link)
- If you have issues, you may have better luck with these instructions (link).

Practice Exercises

Mathematical exercises

- **Natural Numbers Game**: mostly Peano arithmetic play here
- **Mathematics in Lean (Lean 4 + mathlib)**: full of exercises on several undergraduate math topics online | PDF
- Kevin Buzzard's notes/course materials (Lean 4): logic and algebra notes site | course repo

More type theory based resources

- **Functional Programming in Lean (FPiL)**: very helpful for beginners online
- Theorem Proving in Lean 4 (TPiL): online
- Metaprogramming in Lean 4: online

Other resources

Problem-solving help

- leansearch.net: Search for statements in Lean
- Tactic reference
- Manual

Lean community resources

- Learning Lean resources (more practice problems!)
- List of (past) courses
- Lean developments and documentation (many projects)