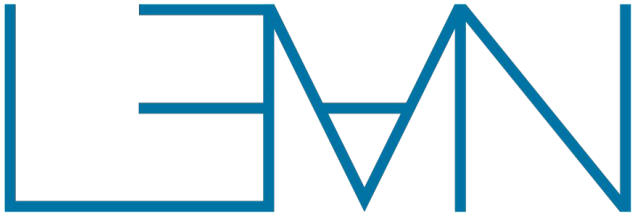


Interactive Theorem Proving with Lean

Dr.-Ing. Sebastian Ullrich
Head of Engineering, Lean FRO

January 27, 2026

About me

2010–2016 MSc in CS at KIT
- 2016 Master's thesis on Lean+Rust at CMU (Pittsburgh, USA)

2017–2023 PhD in CS at the Programming Paradigms group, KIT
- 2018 Internship at Microsoft Research (Redmond, USA), design draft of Lean 4

2023–now Co-founder of the Lean Focused Research Organization together with Leonardo de Moura (AWS)
- 21 people worldwide, 8 in Germany, 3 in Munich

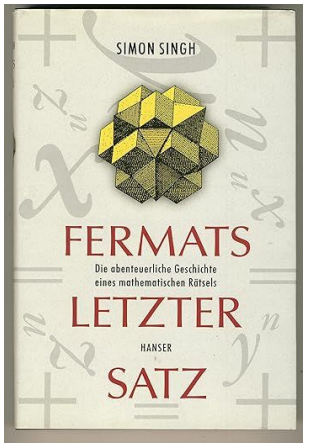
Why Prove?

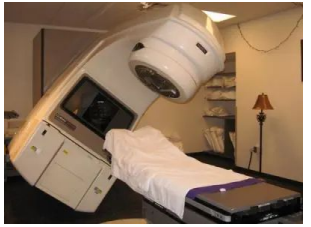
Unverified Mathematics: Mistakes in proofs or logical gaps that go unnoticed

Unverified Software: Bugs, vulnerabilities, and failures in critical systems

Unverified AI: Hallucinations, incorrect outputs, and unreliable reasoning steps

The Lean project started in 2013 with the goal of addressing challenges in software verification. Today, it has gained popularity in both mathematics and AI.





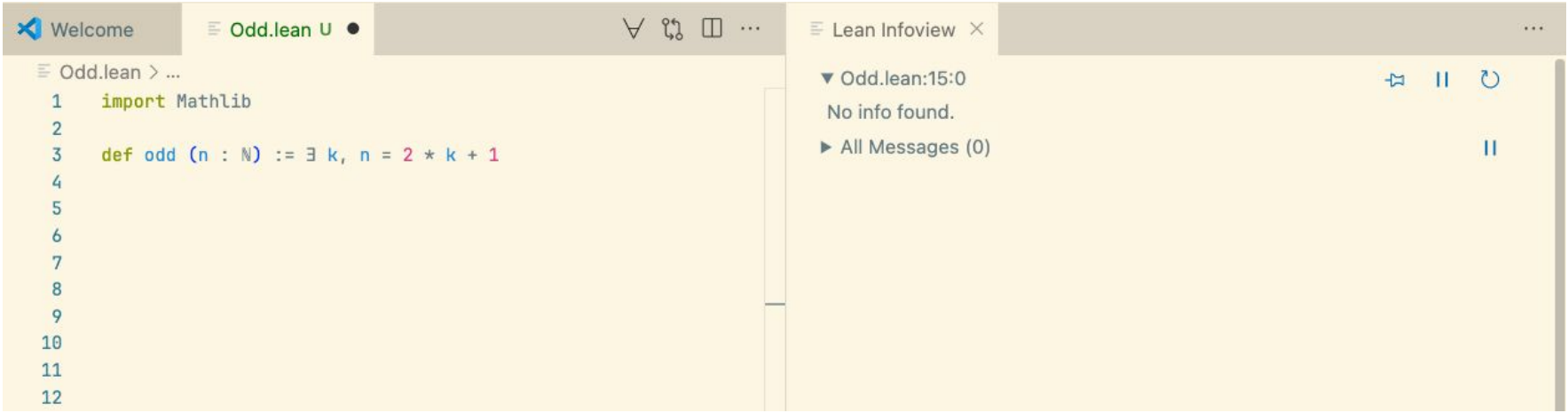
Lean is an open-source **programming language** and **proof assistant** that is transforming how we approach mathematics, software verification, and AI

Lean provides **machine-checkable proofs**

Lean addresses the “**trust bottleneck**”

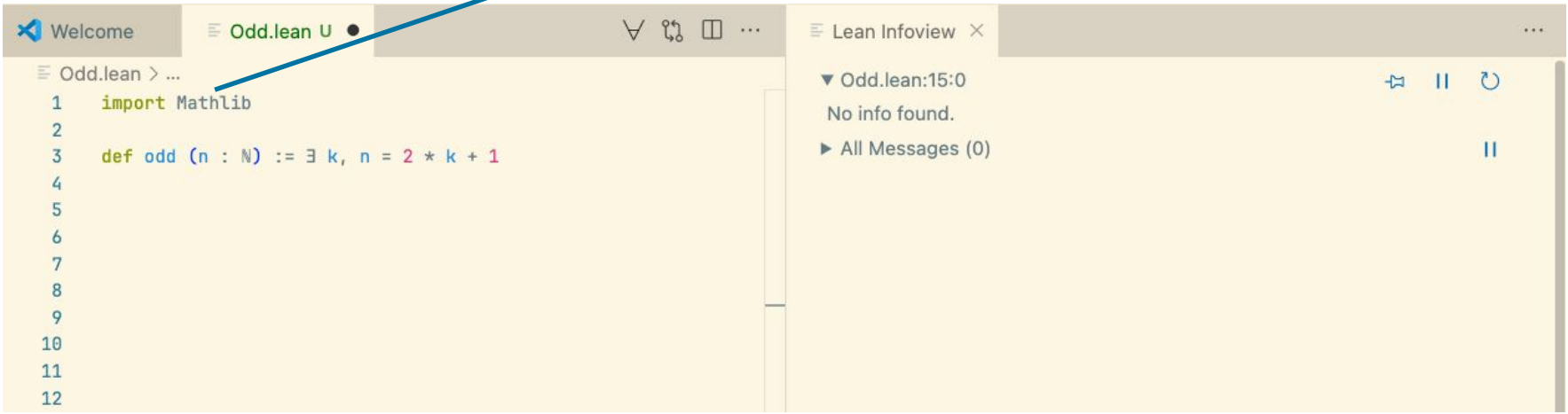
Lean opens up new possibilities for **collaboration**

A small example

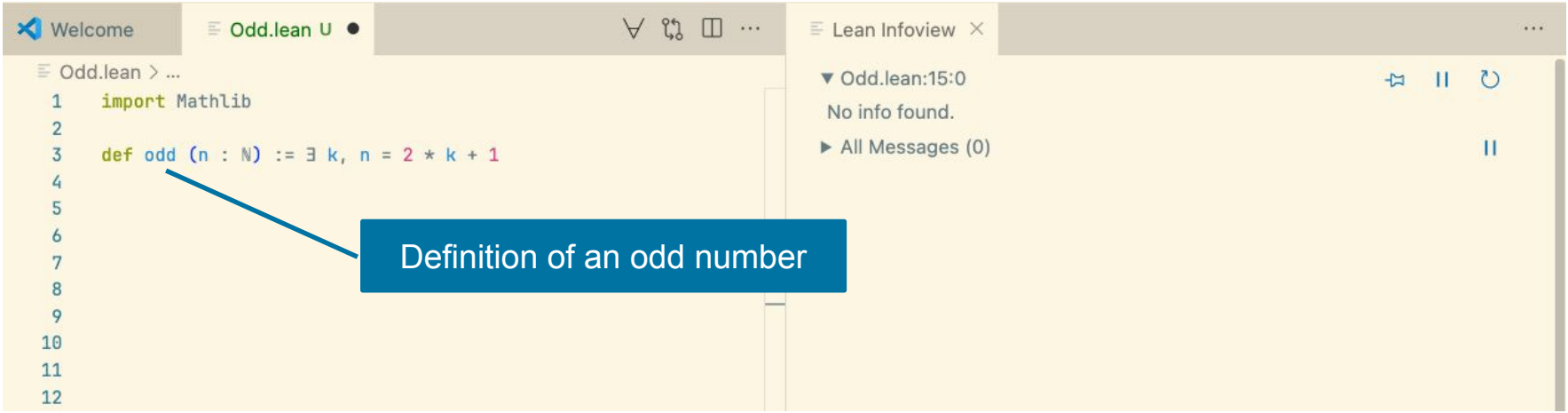


A small example

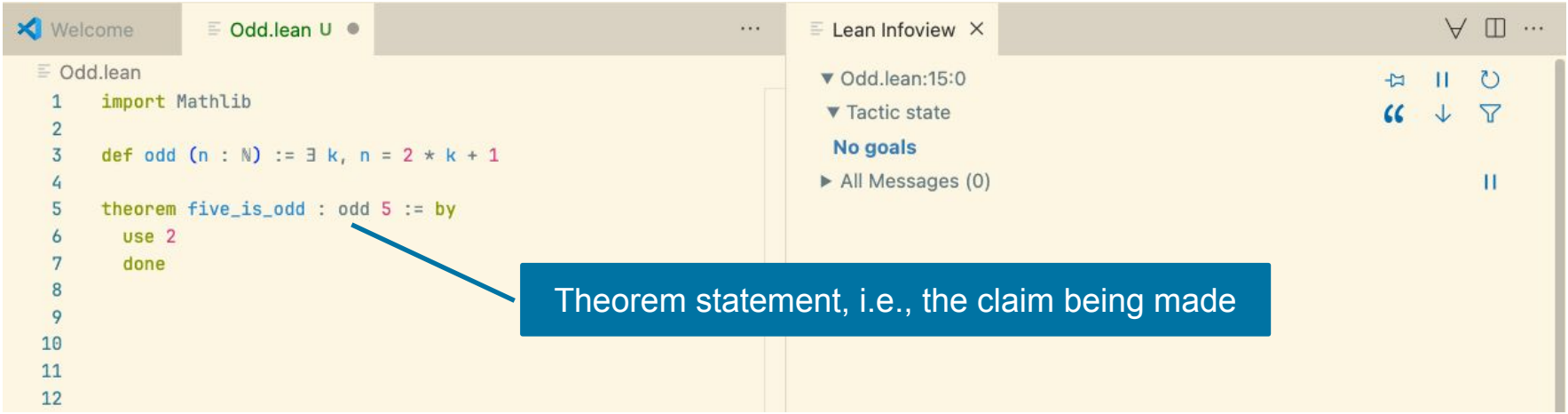
Mathlib is the Lean Mathematical library



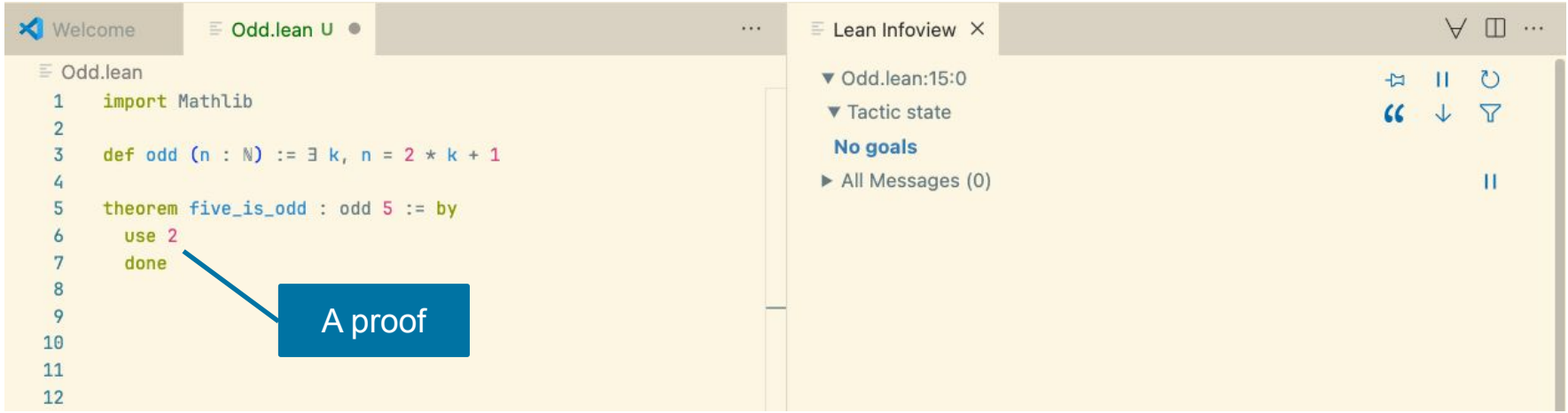
A small example



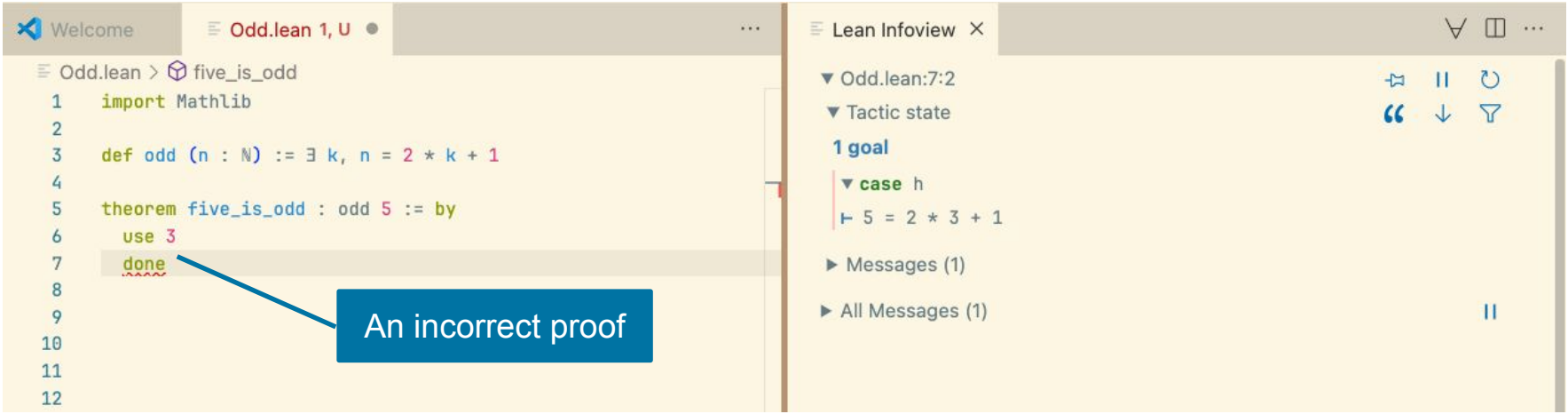
Our first theorem



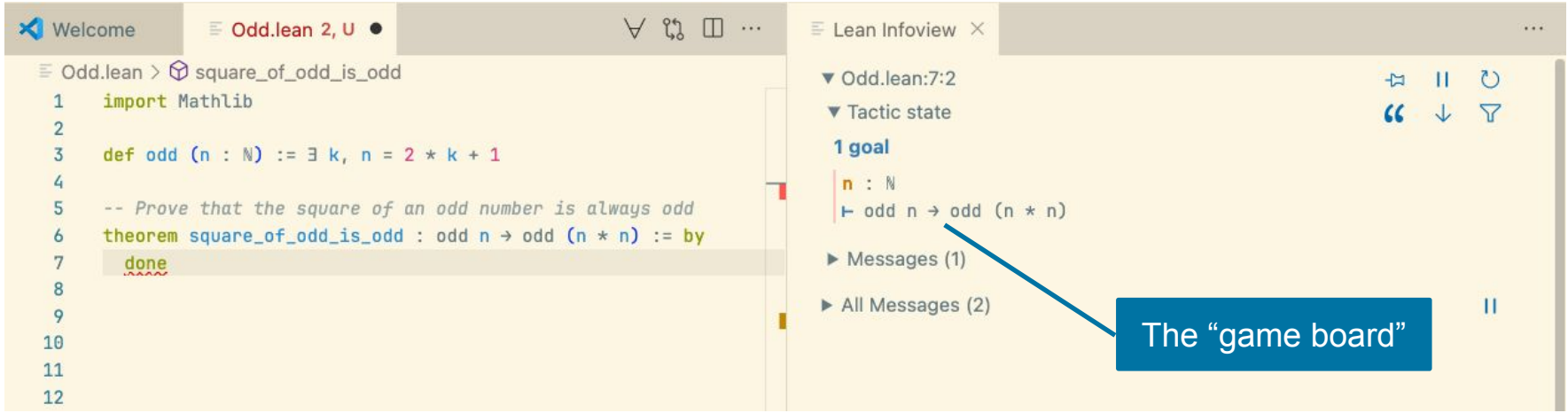
Our first theorem



Our first theorem

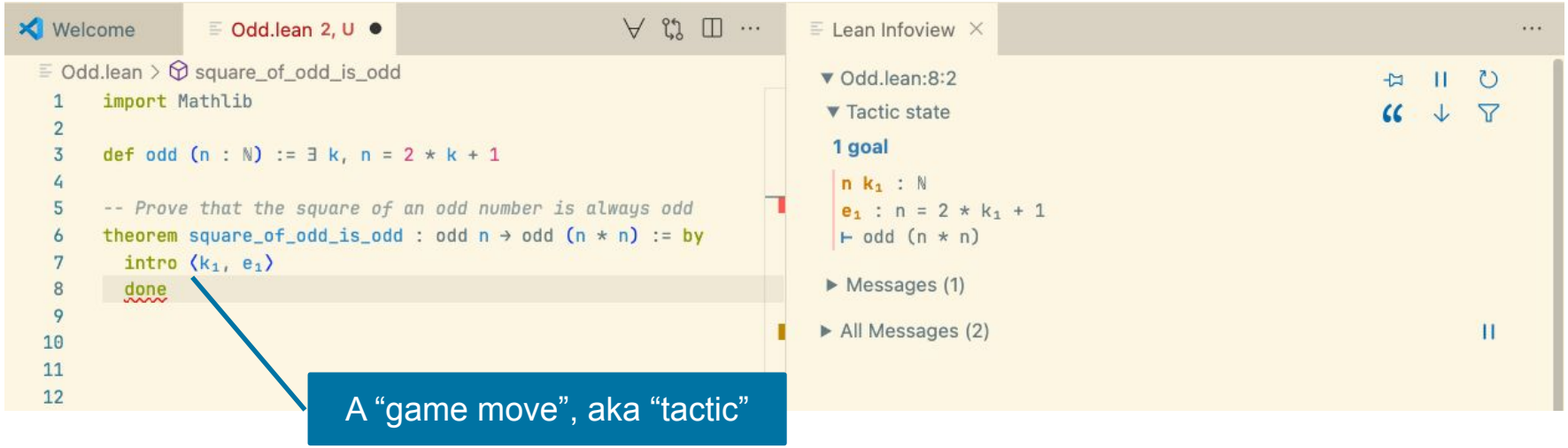


Theorem proving in Lean is an interactive game

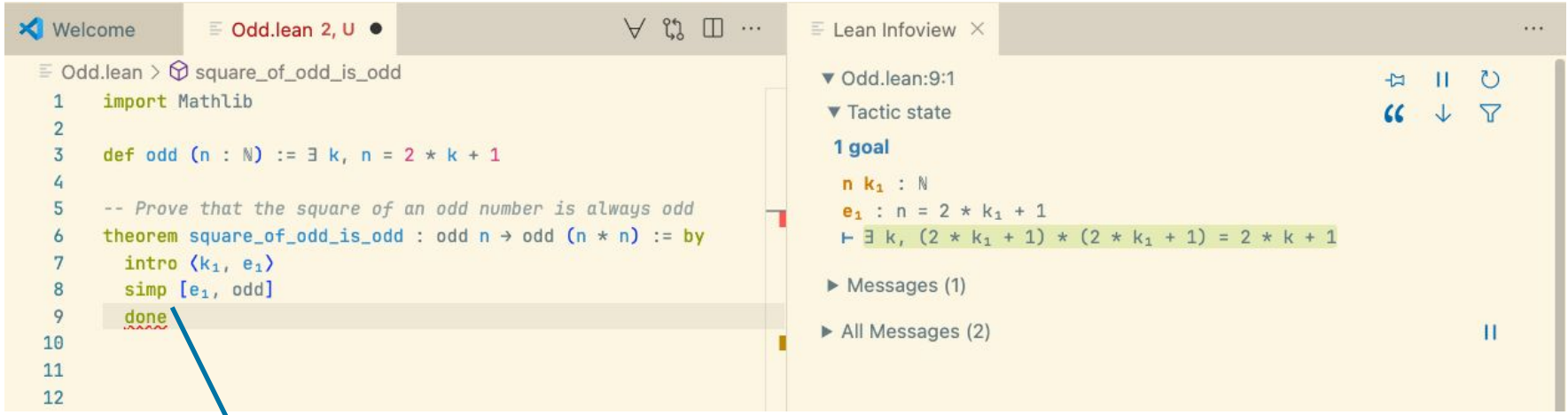


“You have written my favorite computer game” – Kevin Buzzard (Imperial College London)

Theorem proving in Lean is an interactive game

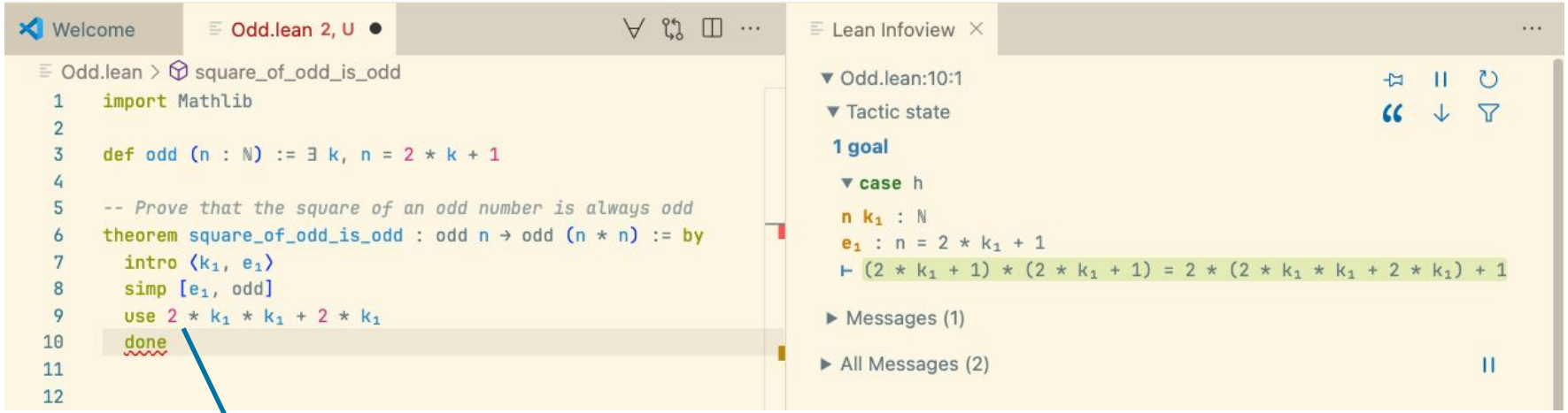


Theorem proving in Lean is an interactive game



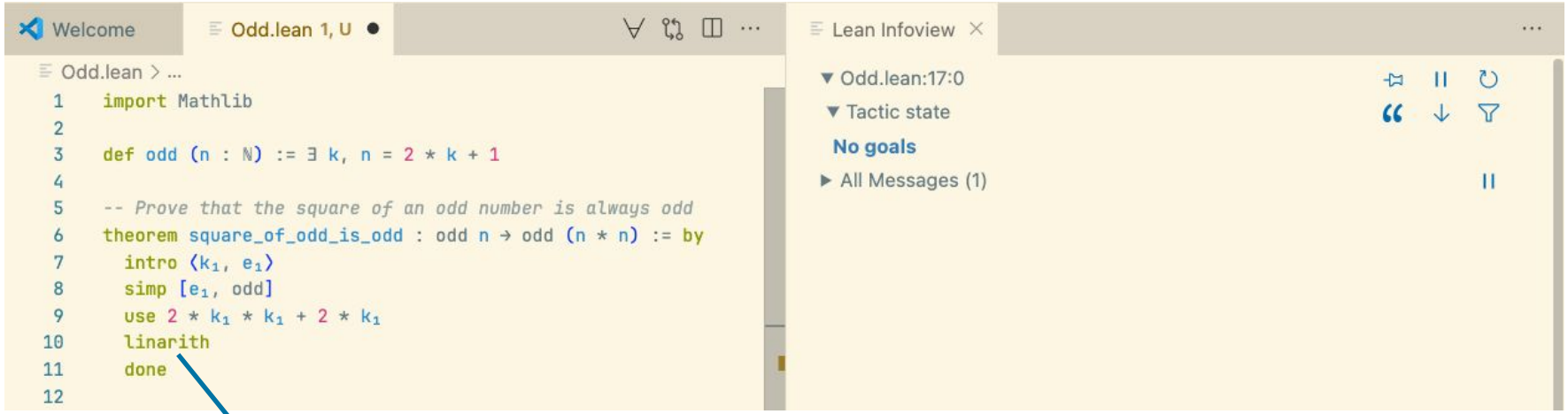
The “game move” `simp`, the simplifier, is one of the most popular moves in our game

Theorem proving in Lean is an interactive game



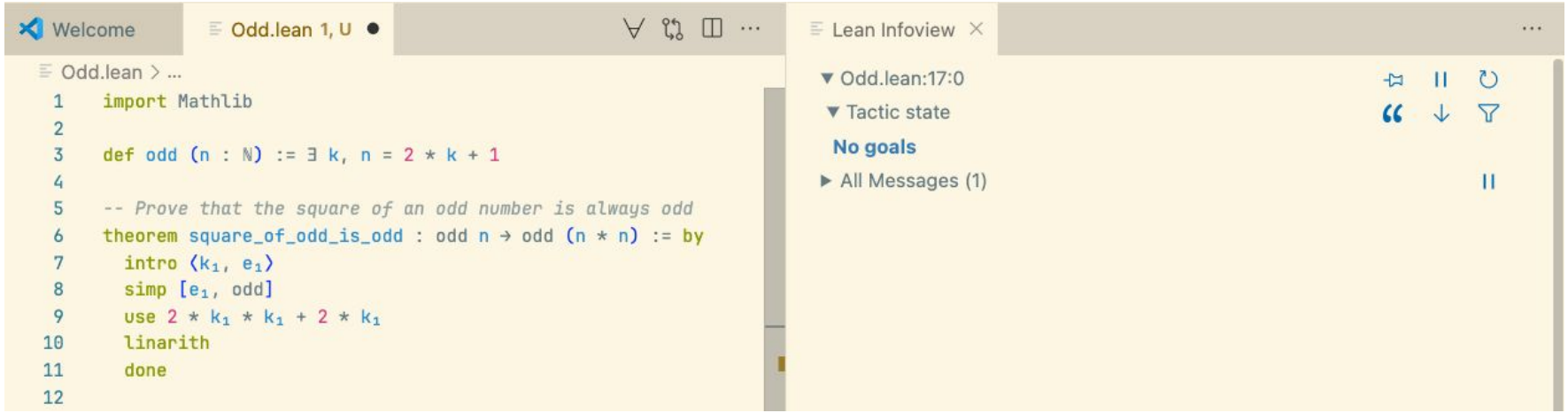
The “game move” `use` is the standard way of proving statements about existentials

Theorem proving in Lean is an interactive game



We complete this level using `linarith`, the linear arithmetic, move

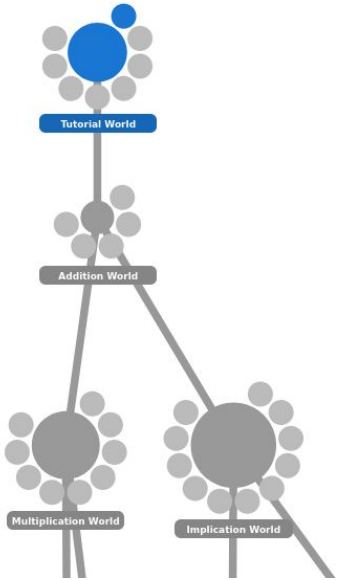
Theorem proving in Lean is an interactive and addictive game

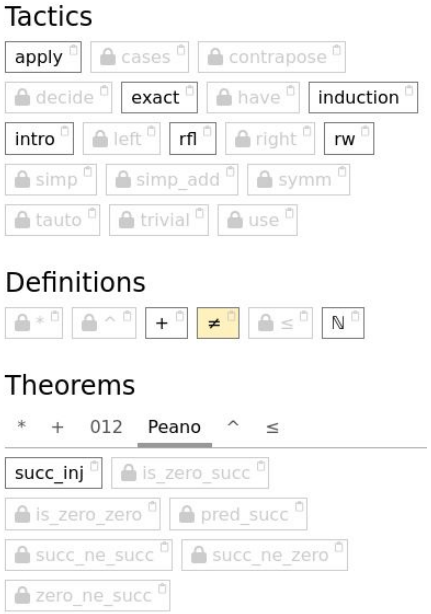


"You can do 14 hours a day in it and not get tired and feel kind of high the whole day.

You're constantly getting positive reinforcement" – Amelia Livingston (University College London)

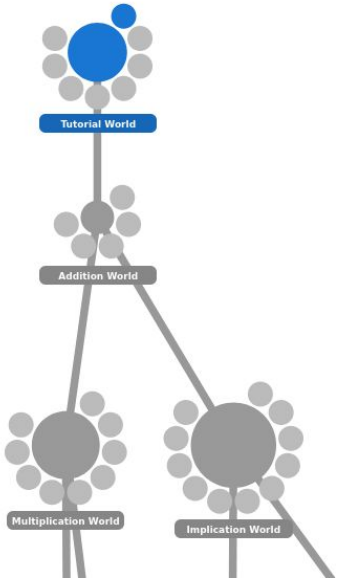
Trying it for yourself: The Natural Number Game

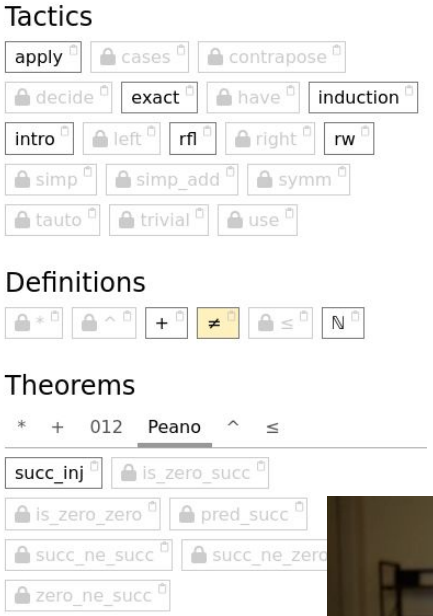




adam.math.hhu.de

Trying it for yourself: The Natural Number Game





adam.math.hhu.de

…as featured on Jonathan Blow's Twitch stream on Jan 25, 2026


Mathematics
Software
AI

Mathematics

Mathlib github.com/leanprover-community/mathlib4

The Lean Mathematical Library supports a wide range of projects

It is an open-source **collaborative project** with over 500 contributors and 2M LoC

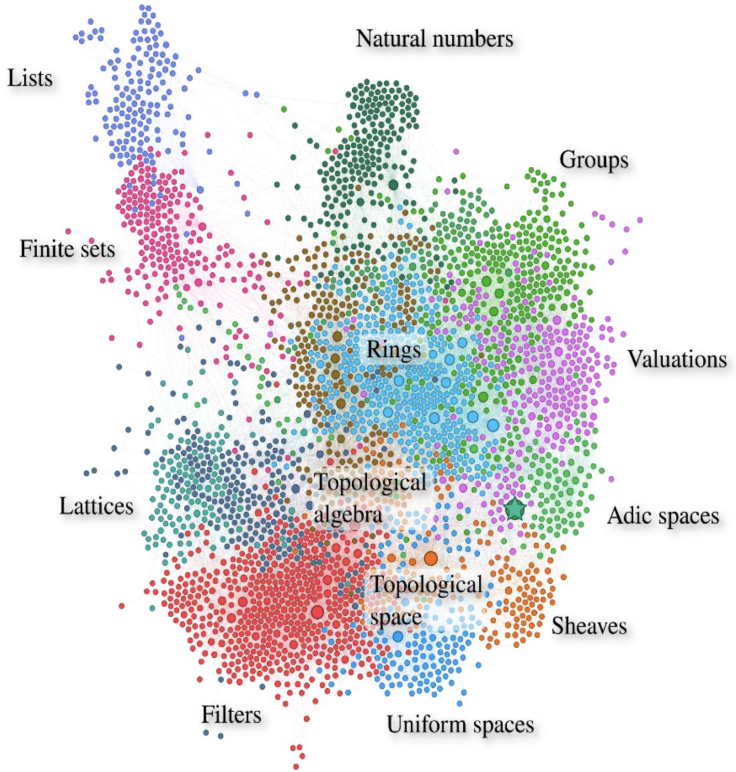
“I’m investing time now so that somebody in the future can have that amazing experience” –
Heather Macbeth (Fordham University)

KEVIN HARTNETT SCIENCE OCT 11, 2020 8:00 AM

The Effort to Build the Mathematical Library of the Future

A community of mathematicians is using software called Lean to build a new digital repository. They hope it represents where their field is headed next.

Mathlib


Natural numbers
Lists
Groups
Finite sets
Rings
Valuations
Topological
algebra
Lattices
Adic spaces
Topological
space
Sheaves
Filters
Uniform spaces

The Perfectoid Spaces Project

Kevin Buzzard, Patrick Massot, Johan Commelin

Goal: Demonstrate that we can **define complex mathematical objects** in Lean.

They translated Peter Scholze's definition into a form a computer can understand.

It not only achieved its goals but also demonstrated to the math community that **formal objects can be visualized and inspected with computer assistance**.

Math is now **data** that can be **processed**, **transformed**, and **inspected** in various ways.

The Perfectoid Spaces Project

mathoverflow

Home

What are "perfectoid spaces"?

72

Here is a completely different kind of answer to this question.

A *perfectoid space* is a term of type `PerfectoidSpace` in the Lean theorem prover.

Here's a quote from the source code:

```
structure perfectoid_ring (R : Type) [Huber_ring R] extends Tate_ri
(complete  : is_complete_hausdorff R)
(uniform   : is_uniform R)
(ramified  : ∃ ϖ : pseudo_uniformizer R, ϖ^p | p in R°)
(Frobenius : surjective (Frob R°/p))

/-
CLVRS ("complete locally valued ringed space") is a category
```

The Challenge

In November of 2020, Peter Scholze posits the Liquid Tensor Experiment (LTE) challenge

> *“I spent much of 2019 obsessed with the proof of this theorem, almost getting crazy over it. In the end, we were able to get an argument pinned down on paper, but I think nobody else has dared to look at the details of this, and so I still have some small lingering doubts”*

The First Victory

Johan Commelin led a team with several members of the **Lean community and announced the formalization of the crucial intermediate lemma** that Scholze was unsure about, with only minor corrections, in **May 2021**.

> *"[T]his was precisely the kind of oversight I was worried about when I asked for the formal verification. [...] The proof walks a fine line, so if some argument needs constants that are quite a bit different from what I claimed, it might have collapsed"* – Peter Scholze


nature

Explore content | Journal information | Publish with us | Subscribe

nature > news > article

NEWS | 18 June 2021

Mathematicians welcome computer-assisted proof in 'grand unification' theory

Success

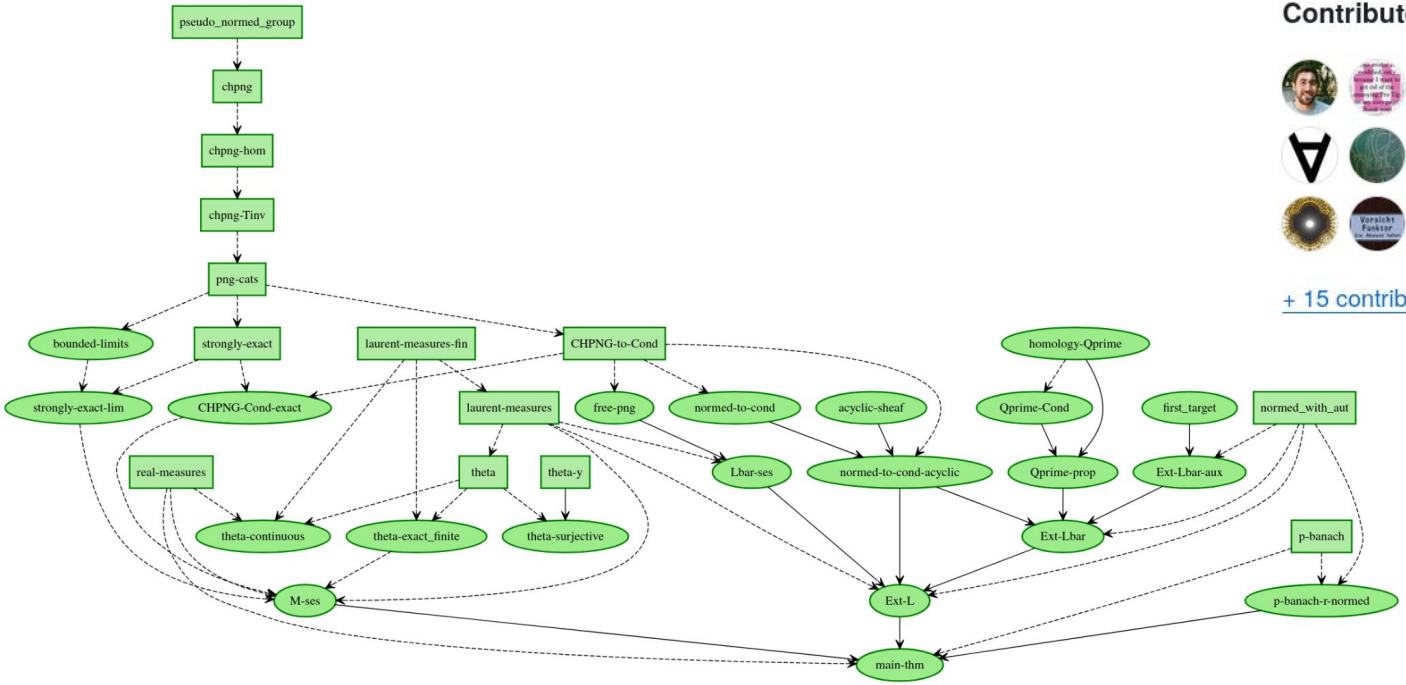
The full challenge was completed in July 2022.

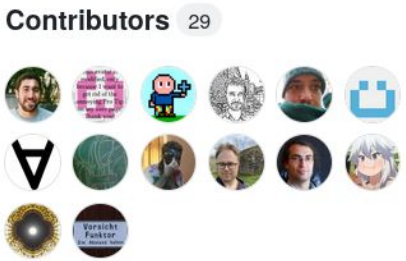
The team not only verified the proof but also simplified it.
Moreover, they did this without fully understanding the entire proof.

Johan, the project lead, reported that he could only see two steps ahead. **Lean was a guide**.

> *"The Lean Proof Assistant was really that: an assistant in navigating through the thick jungle that this proof is. Really, one key problem I had when I was trying to find this proof was that I was essentially unable to keep all the objects in my RAM, and I think the same problem occurs when trying to read the proof"* – Peter Scholze

Crowd-Sourced Mathematics





Only the Beginning

Independence of the Continuum Hypothesis, Han and van Doorn, 2021

Sphere Eversion, Massot, Nash, and van Doorn, 2020-2022

Unit Fractions, Bloom and Mehta, 2022

Consistency of Quine's New Foundations, Wilshaw and Dillies, 2022-2024

Polynomial Freiman-Ruzsa Conjecture, Tao and Dillies, 2023

Prime Number Theorem And Beyond, Kontorovich and Tao, 2024-ongoing

Carleson Project, van Doorn, 2024-2025

Equational Theories Project, Tao, Monticone, and Srinivas, 2024-2025

Fermat’s Last Theorem (FLT), Buzzard, 2024-ongoing, community estimates it will take 1M+ LoC

Sphere Packing, Birkbeck et al., 2025-ongoing

Summary

Machine-checkable proofs enable a new level of **collaboration** in mathematics

It is not just about proving but also understanding complex objects and proofs, getting new insights, and navigating through the “thick jungles” that are **beyond our cognitive abilities**

Software

Lean in Software Verification: The Story of SampCert

Lean is a programming language, and is used in **many software verification projects**

You can write code and reason about it simultaneously

You can prove that your code has the properties you expect

> *"Testing can show the presence of bugs, but not their absence"* – E. Dijkstra

Differential Privacy

A mathematical framework that ensures the **privacy of individuals** in a dataset by adding controlled **random noise** to the data

Discrete sampling algorithms, like the *Discrete Gaussian Sampler*, are used to add carefully calibrated noise to data

What may go wrong if a buggy sampler is used?

Privacy Violations: leakage of sensitive information

Incorrect Results: distorted analysis results

The 2017 Gödel Prize was awarded to Dwork et al for the development of diff. privacy

SampCert

A project led by **Jean-Baptiste Tristan** at AWS

An **open-source** Lean library of formally **verified differential privacy primitives**

Tristan's implementation is not only verified, but it is also **twice as fast as the previous one**

He managed to implement **aggressive optimizations** because Lean served as a guide, ensuring that **no bugs** were introduced

SampCert would not exist without Mathlib

SampCert is software, but its verification relies heavily on Mathlib

The verification of code addressing practical problems in data privacy depends on the formalization of mathematical concepts, from **Fourier analysis** to **number theory** and **topology**

Many more open-source projects

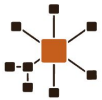
Cedar, a policy language and evaluation engine

LNSym, a symbolic simulator for Armv8 native-code programs: cryptographic machine-code programs

TenCert, a tensor compiler, verified StableHLO and NKI

NFA2SAT, a verified string solver

Many more at the **Lean Project Registry**: reservoir.lean-lang.org



CSLib is a new industry/academia effort on formalizing computer science in Lean

Summary

Machine-checkable proofs enable you to **code, refactor, and optimize without fear**

AI

Lean Enables Verified AI for Mathematics and Code

LLMs are powerful tools, but they are prone to **hallucinations**

In math, a **small mistake can invalidate the whole proof**

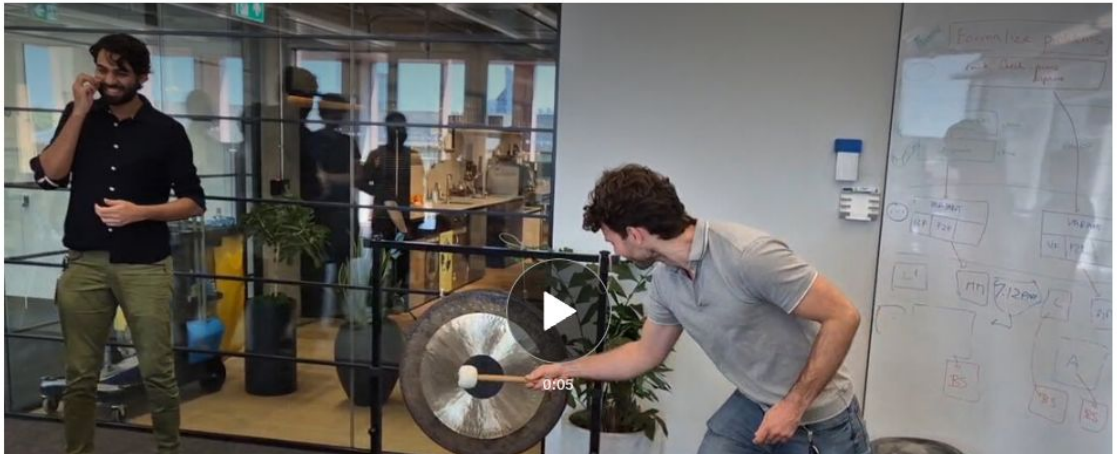
Imagine manually checking an AI-generated proof with the size and complexity of FLT

- The informal proof is **over 200 pages**
- Buzzard estimates a formal proof will require more than **1M LoC** on top of Mathlib

Machine-checkable proofs are the antidote to hallucinations

Move Over, Mathematicians, Here Comes AlphaProof

A.I. is getting good at math — and might soon make a worthy collaborator for humans.



AlphaProof & the International Math Olympiad

Determine all real numbers α such that, for every positive integer n, the integer

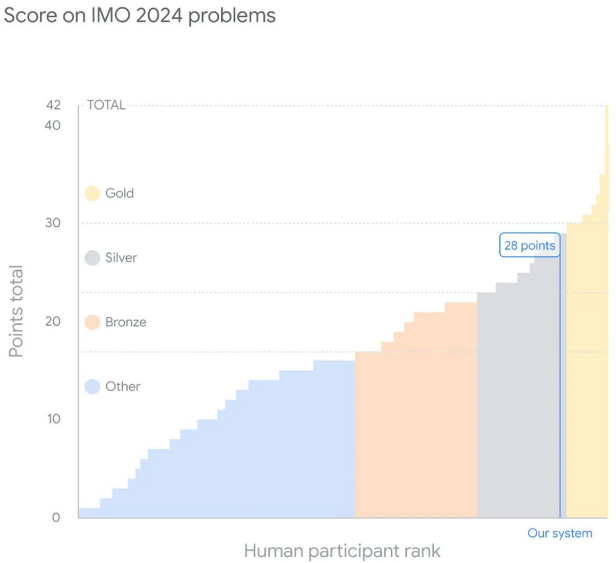
$$\lfloor \alpha \rfloor + \lfloor 2\alpha \rfloor + \cdots + \lfloor n\alpha \rfloor$$

is a multiple of n. (Note that $\lfloor z \rfloor$ denotes the greatest integer less than or equal to z. For example, $\lfloor -\pi \rfloor = -4$ and $\lfloor 2 \rfloor = \lfloor 2.9 \rfloor = 2$.)

Solution: α is an even integer.

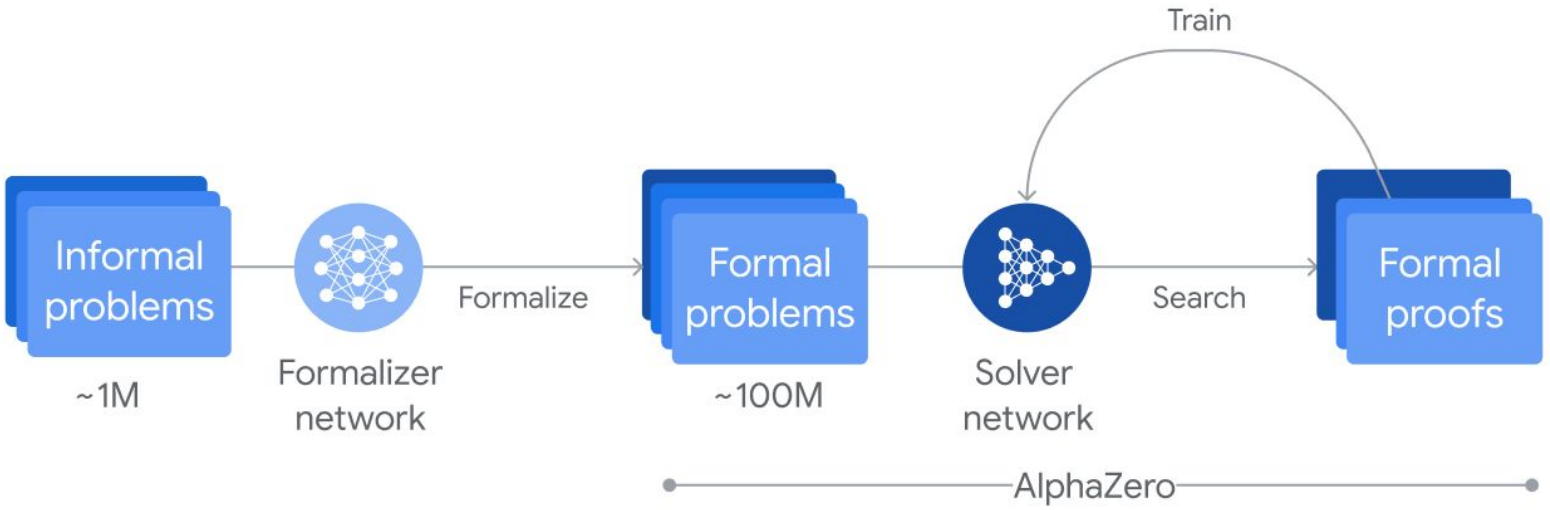
```
open scoped BigOperators

theorem imo_2024_p1 :
    {(α : ℝ) | ∀ (n : ℕ), 0 < n → (n : ℤ) | (∑ i in Finset.Icc 1 n, ⌊i * α⌋)}
    = {α : ℝ | ∃ k : ℤ, Even k ∧ α = k} := by
  rw [(Set.Subset.antisymm_iff ), (Set.subset_def), ]
  /- We introduce a variable that will be used
     in the second part of the proof (the hard direction)
```



deepmind.google/discover/blog/ai-solves-imo-problems-at-silver-medal-level

Learning à la DeepMind



deepmind.google/discover/blog/ai-solves-imo-problems-at-silver-medal-level

IMO 2025: 3 Gold, 1 Silver

OpenAI (informal)

DeepMind (informal)

Advanced version of Gemini with Deep Think officially achieves gold-medal standard at the International Mathematical Olympiad

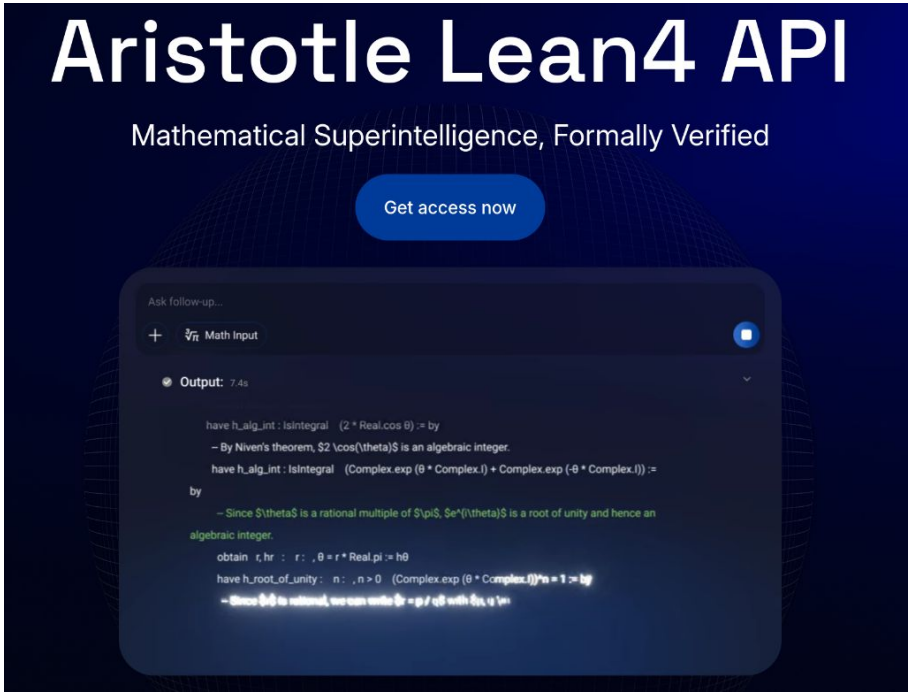
Harmonic (Lean)

Aristotle Achieves Gold Medal-Level Performance at the International Mathematical Olympiad, iOS App Beta Launch

ByteDance (Lean)

ByteDance Seed Prover Achieves Silver Medal Score in IMO 2025

IMO-Level Proof Search as a Service



AlphaProof Interest Form

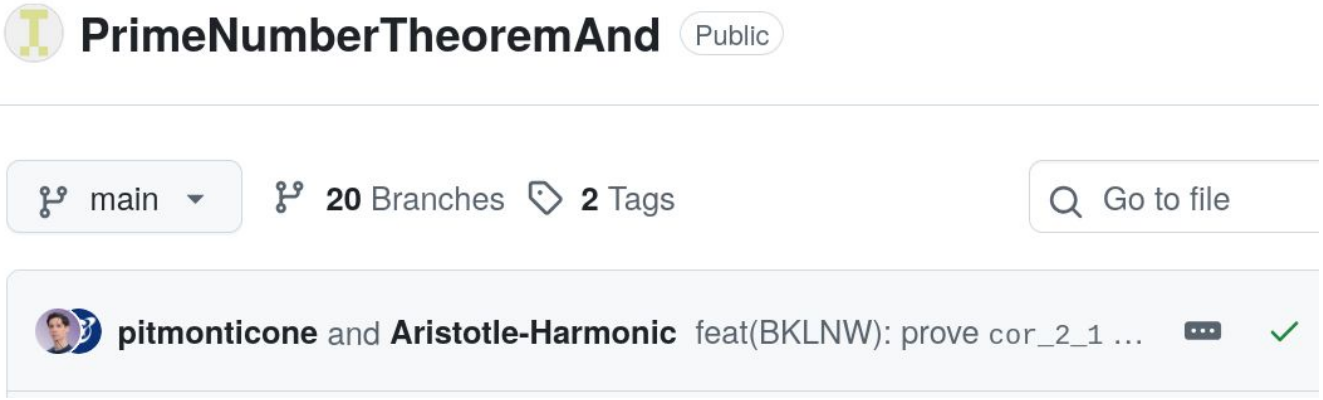
Thanks for your interest in AlphaProof. To deliver a reliable user experience, we are carefully managing our compute resources to maintain system stability. We will be reviewing applications and prioritizing access over time as we scale the system. We appreciate your patience and understanding.

To help you get the most from this process, we encourage you to first explore the AlphaProof Blog Post, the AlphaProof Nature Paper, and the **AlphaProof Tool Demos [1][2]**. Understanding AlphaProof beforehand will empower you to complete this form more effectively and confidently.

aristotle.harmonic.fun

deepmind.google/alphaproof

IMO-Level Proof Search as a Service



Putnam Bench as the Next Frontier Already Being Solved

Lean (out of 672)

#	Model	num-solved	compute
1	Aleph Prover (Logical Intelligence)	668	Avg $68, Limit $1400 per problem
2	Aleph Prover (Logical Intelligence)	637	Avg $54, Limit $400 per problem
3	Seed-Prover 1.5 (ByteDance)	581	10 H20 days per problem
4	Aleph Prover (Logical Intelligence)	500	Avg $23, Limit $100 per problem
5	Hilbert	462	avg pass@1840
6	Seed-Prover (ByteDance)	329	MEDIUM
7	Ax-Prover (Axiomatic AI)💙	91	pass@1, avg. 100 tool calls
8	Goedel-Prover-V2💚	86	pass@184
9	DeepSeek-Prover-V2💚	47	pass@1024
10	GPT-5 (ReAct, 10 turns)	28	pass@1, 10 tool calls

💚 fully open-sourced

💙 partially open-sourced

trishullab.github.io/PutnamBench/leaderboard.html

Autoformalization

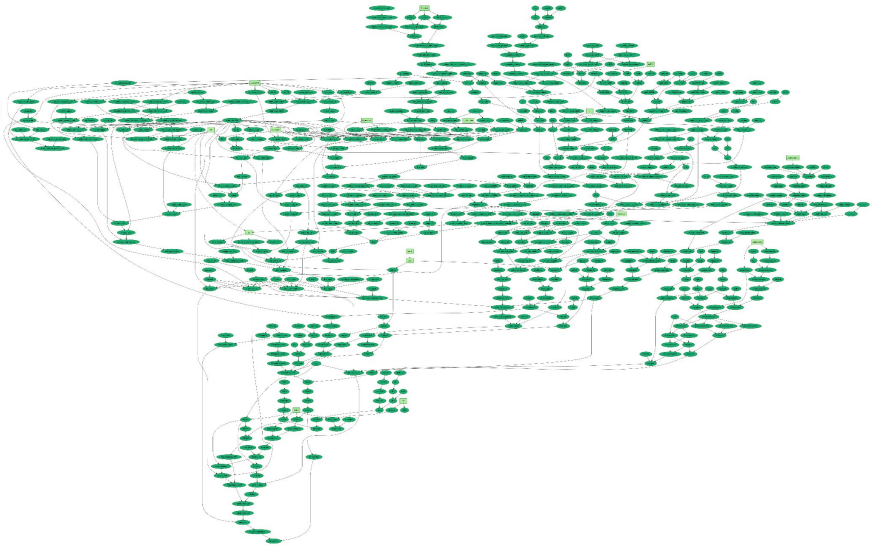
Auguste Poiroux

Dear all,

We, at Math, Inc., are excited to introduce Gauss, a new autoformalization agent designed to assist mathematicians in their Lean formalization.

Using Gauss, and building over the recent medium PNT progress by Kontorovich et al., we formalized the **strong Prime Number Theorem** in Lean. Over the course of three weeks, Gauss produced ~25k lines of Lean code and over 1,000 theorems and definitions. To put these numbers in perspective, our previous iteration, three months ago, produced ~3.5k lines of Lean code and 100 theorems and definitions to formalize the abc conjecture almost always.

Our goal is to make Gauss accessible and easy to use. If you are interested, you can register for early access to Gauss here. We are happy to discuss about Gauss and Math, Inc. in the discussion thread.



github.com/math-inc/strongpnt

Vibe Proving

Forbidden Sidon subsets of perfect difference sets, featuring a human-assisted proof

Boris Alexeev ChatGPT* Lean† Dustin G. Mixon‡§

Abstract

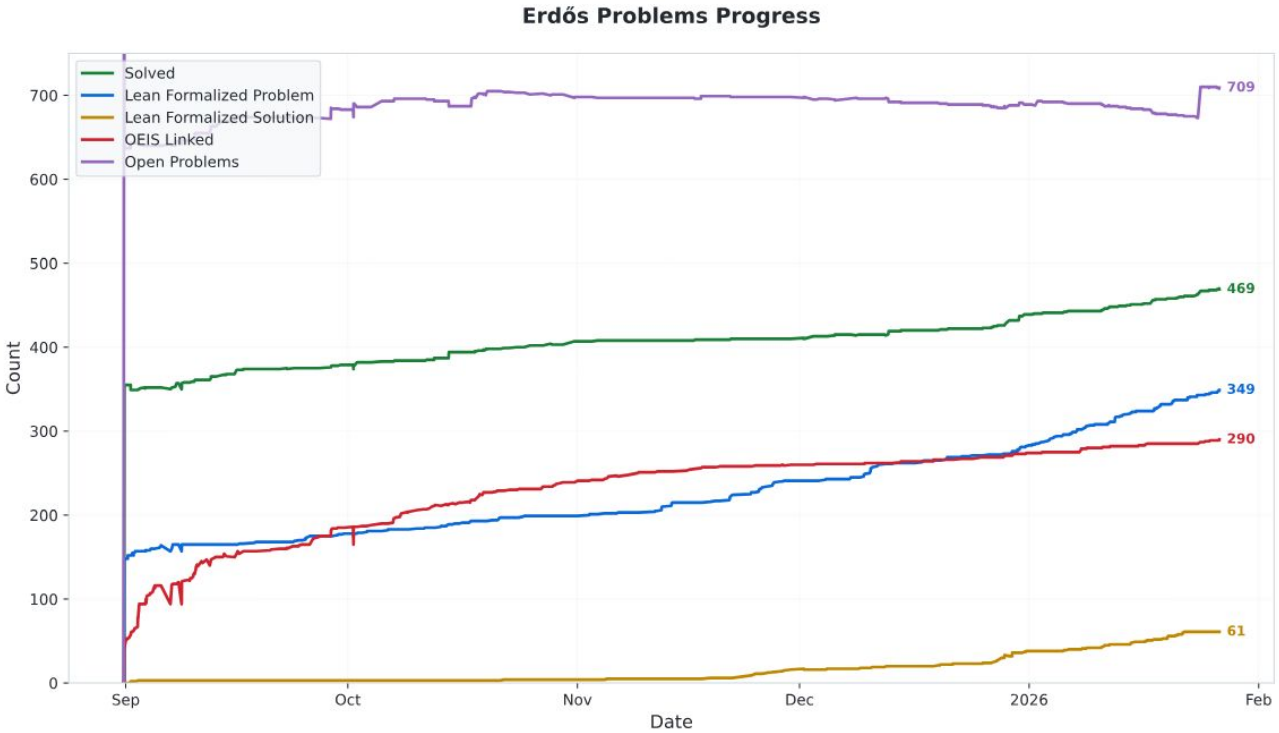
We resolve a $1000 Erdős prize problem, complete with formal verification generated by a large language model.

In over a dozen papers, beginning in 1976 and spanning two decades, Paul Erdős repeatedly posed one of his "favourite" conjectures: every finite Sidon set can be extended to a finite perfect difference set. We establish that $\{1, 2, 4, 8, 13\}$ is a counterexample to this conjecture.

During the preparation of this paper, we discovered that although this problem was presumed to be open for half a century, Marshall Hall, Jr. published a different counterexample three decades *before* Erdős first posed the problem. With a healthy skepticism of this apparent oversight, and out of an abundance of caution, we used ChatGPT to vibe code a Lean proof of both Hall's and our counterexamples.

borisalexeev.com/pdf/erdos707.pdf

Vibe Proving



github.com/teorth/erdosproblems

Vibe Proving

Problem	AI tools used	Date	Outcome
[11]	ChatGPT, Aristotle	24 Jan, 2026	🔴 Incorrect claim made
[42]	GPT-5.2, GPT-5.2 Pro, Codex	19 Jan, 2026	🟡 Partial result (Lean)
[51]	ChatGPT (free version)	11 Jan, 2026	🔴 Incorrect proof found
[64]	AlphaEvolve	3 Nov, 2025	No counterexample found
[124]	Aristotle	29 Nov, 2025	🟡 Partial result (Lean)
[205]	Aristotle, ChatGPT 5.2 Thinking	10 Jan, 2026	🟢 Full solution (Lean)

github.com/teorth/erdosproblems

Vibe Proving

1(d). Solutions generated by humans in collaboration with AI

Problem	Human collaborators	AI tools used	Date	Outcome
[347]	ebarschkis, Wouter van Doorn, Bartosz Naskrecki, Terence Tao	GPT Codex, Aristotle	25 Oct, 2025 - 21 Jan, 2026	🟢 Full solution (Lean)
[367]	Boris Alexeev, Wouter van Doorn, Terence Tao	Gemini Deepthink, Aristotle	20-22 Nov, 2025	🟡 Partial result
[401]	Boris Alexeev, Kevin Barreto, Leeham, Nat Sothanaphan	Aristotle, ChatGPT-5.2 Pro	10-11 Jan, 2026	🟢 Counterexample to alternate formulation; full solution to revised formulation (Lean)
[460]	Przemek Chojecki	GPT 5.2	In progress	🟡 Reduction to a simpler problem
[659]	Benjamin Grayzel	Gemini 3.0	13 Jan, 2026	🟢 Full solution (Lean)
[684]	Quanyu Tang	ChatGPT 5.2 Thinking	19 Jan, 2026	🟡 New partial result
[848]	Mehtaab Sawhney, Mark Sellke	GPT-5	12 Oct-20 Nov, 2025	🟢 Full solution
[1026]	Boris Alexeev, Stijn Cambie, Terence Tao, Lawrence Wu	ChatGPT, Aristotle, AlphaEvolve, Gemini	8 Dec, 2025	🟢 Full solution (Lean)
[1038]	Junnosuke Koizumi, jspier, Nat Sothanaphan, Terence Tao	ChatGPT, AlphaEvolve	In progress	🟡 Partial result
[1141]	Quanyu Tang	ChatGPT 5.2 Thinking, ChatGPT 5.2 Pro	25 Jan, 2026	🟡 Partial result of variant problem

github.com/teorth/erdosproblems

Autonomous Exploration

MATHEMATICAL EXPLORATION AND DISCOVERY AT SCALE

BOGDAN GEORGIEV, JAVIER GÓMEZ-SERRANO, TERENCE TAO, AND ADAM ZSOLT WAGNER

ABSTRACT. `AlphaEvolve` [223] is a generic evolutionary coding agent that combines the generative capabilities of LLMs with automated evaluation in an iterative evolutionary framework that proposes, tests, and refines algorithmic solutions to challenging scientific and practical problems. In this paper we showcase `AlphaEvolve` as a tool for autonomously discovering novel mathematical constructions and advancing our understanding of long-standing open problems.

1.5. **Building a pipeline of several AI tools.** Even more strikingly, for the finite field Kakeya problem (cf. Problem 6.1), `AlphaEvolve` discovered an interesting general construction. When we fed this programmatic solution to the agent called `Deep Think` [148], it successfully derived a proof of its correctness and a closed-form formula for its size. This proof was then fully formalized in the Lean proof assistant using another AI tool, `AlphaProof` [147]. This workflow, combining pattern discovery (`AlphaEvolve`), symbolic proof generation (`Deep Think`), and formal verification (`AlphaProof`), serves as a concrete example of how specialized AI systems can be integrated. It suggests a future potential methodology where a combination of AI tools can assist in the process of moving from an empirically observed pattern (suggested by the model) to a formally verified mathematical result, fully automated or semi-automated.

arxiv.org/abs/2511.02864

Summary

Machine-checkable proofs enable **AI that does not hallucinate**

AI can find proofs, translations, and new results, in collaboration with humans or by itself

Wrap-Up

Lean enables decentralized collaboration

Lean is Extensible

Users extend Lean using Lean itself

Lean is implemented in Lean

You can make it your own

You can create your own moves

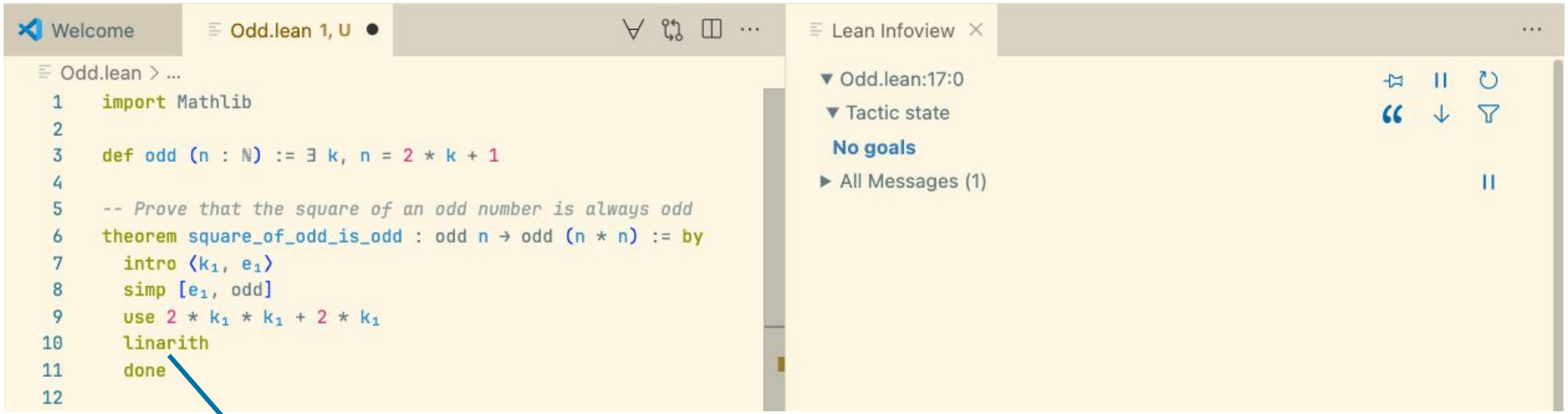
Machine-Checkable Proofs

You don’t need to trust me to use my proofs

You don’t need to trust my automation to use it

Code without fear

Lean is a game where we can implement your own moves



The `linarith` “move” was implemented by the Mathlib community in Lean!

Focused Research Organizations

A new type of nonprofit startup for science developed by Convergent Research

Conclusion

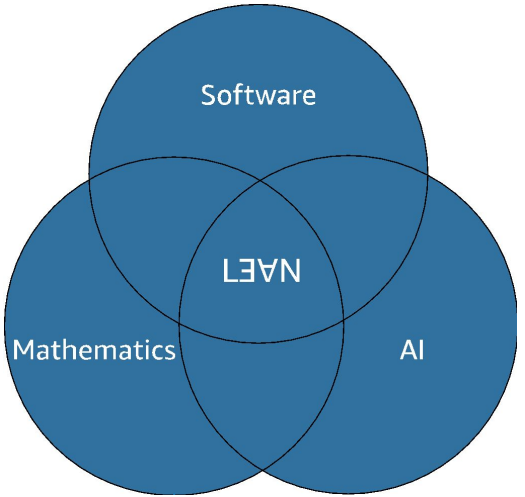
Lean is an **efficient programming language** and **proof assistant**

Machine-checkable proofs eliminate the trust bottleneck

Lean enables **decentralized collaboration**

Lean is very **extensible**

The Mathlib community is changing how math is done



It is not just about proving but also understanding complex objects and proofs, getting new insights, and navigating through the “thick jungles” that are beyond our cognitive abilities

Thank You

lean-lang.org
lean-fro.org
Community chat: leanprover.zulipchat.com
Mastodon: @leanprover@functional.cafe