Verified Collaboration:
How Lean is Transforming Mathematics, Programming, and AI

Leo de Moura
Senior Principal Applied Scientist, AWS
Chief Architect, Lean FRO

October 11, 2024

Breaking the Cycle of Uncertainty: Math, Software, and AI You Can Trust

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.

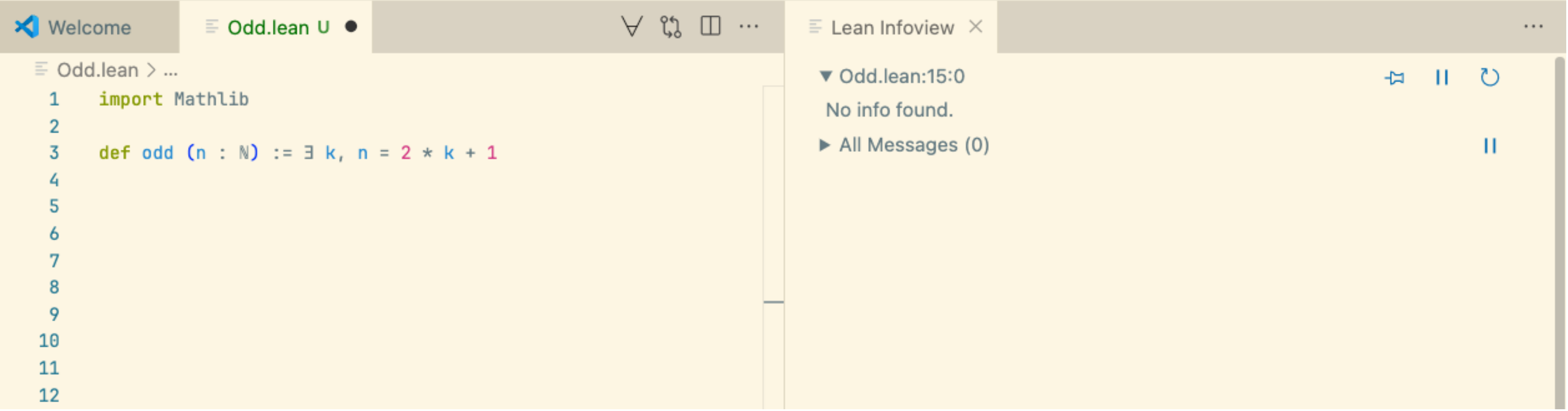
Lean is an open-source **programming language** and **proof assistant** that is transforming how we approach mathematics, software verification, and AI.

Lean provides **machine-checkable proofs**.

Lean addresses the “**trust bottleneck**”.

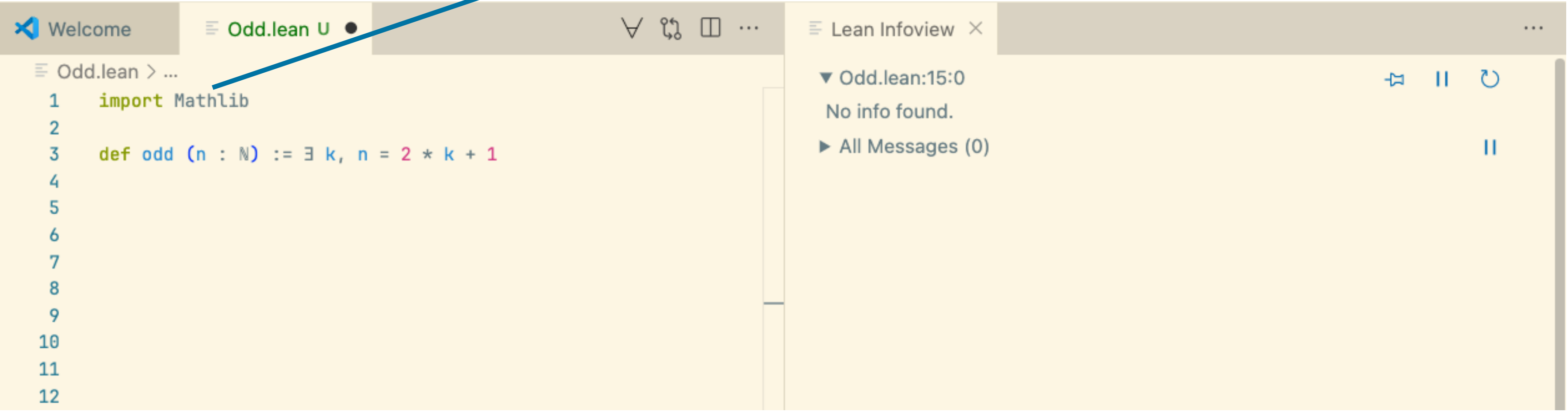
Lean opens up new possibilities for **collaboration**.

A small example

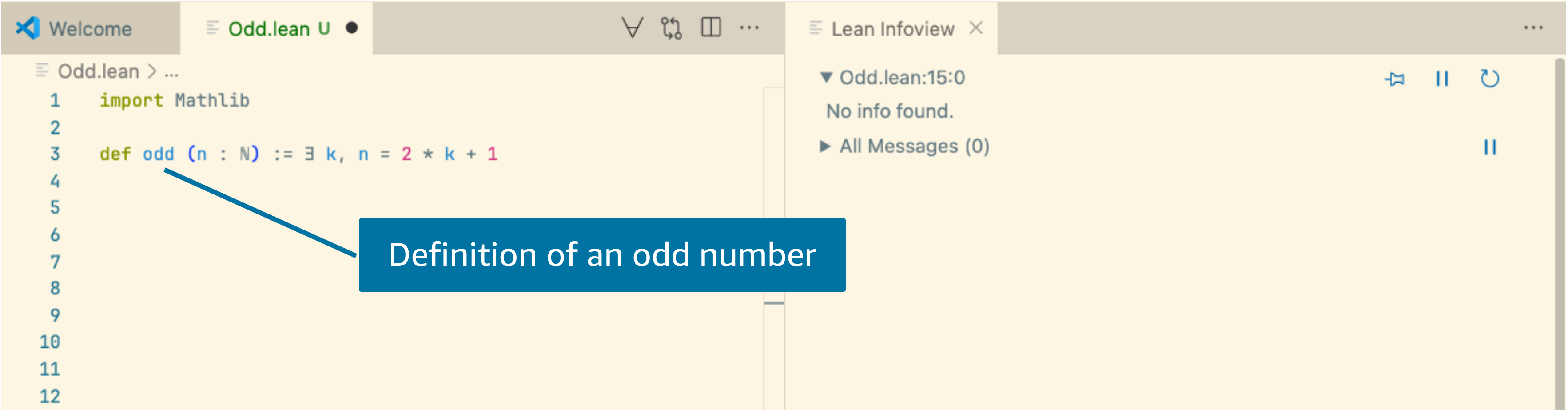


A small example

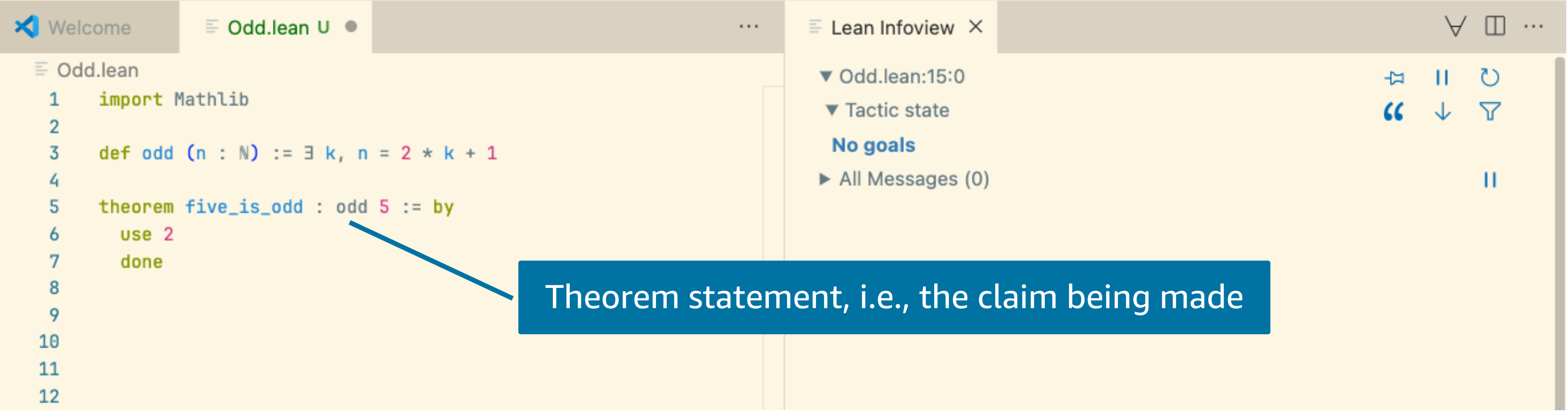
Mathlib is the Lean Mathematical library



A small example

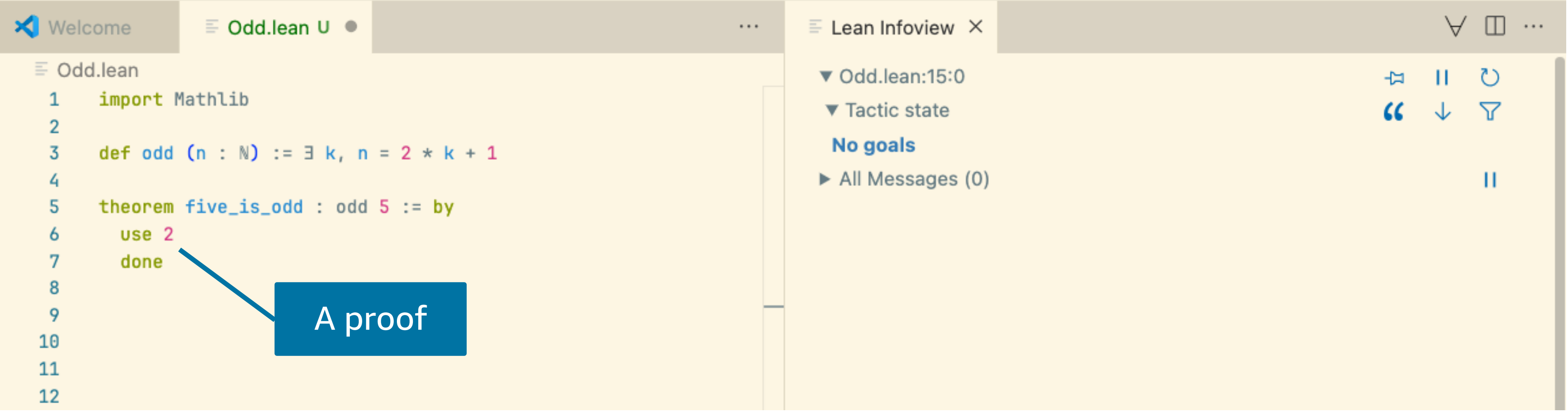


Our first theorem

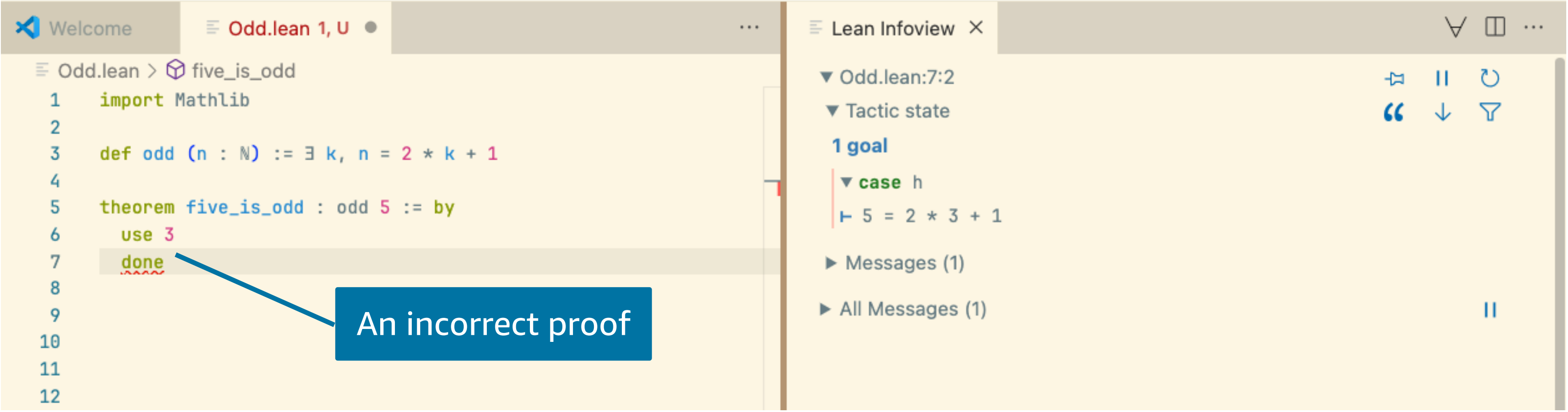


Theorem statement, i.e., the claim being made

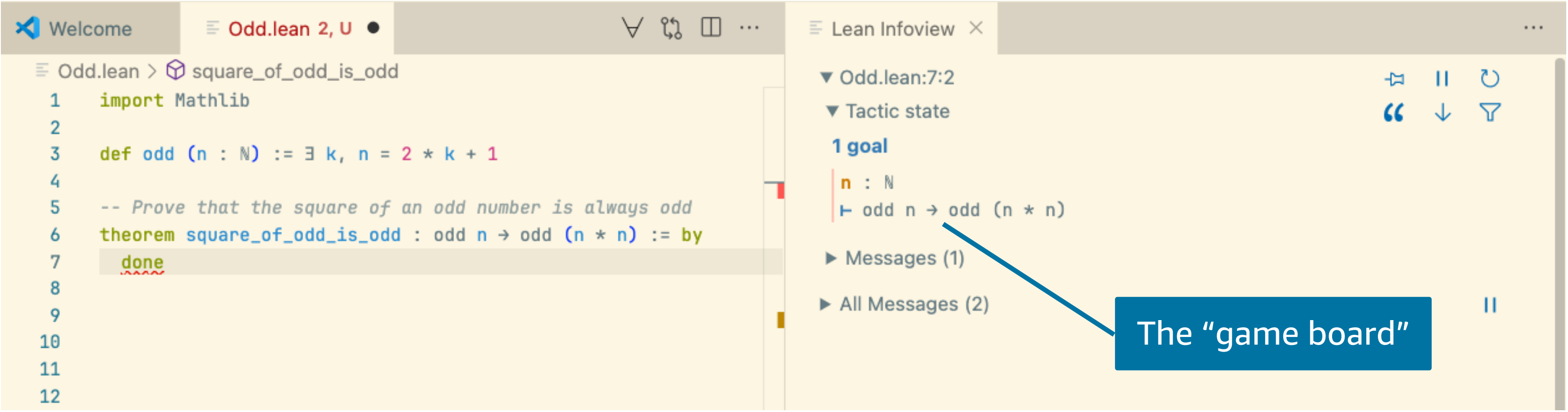
Our first theorem



Our first theorem

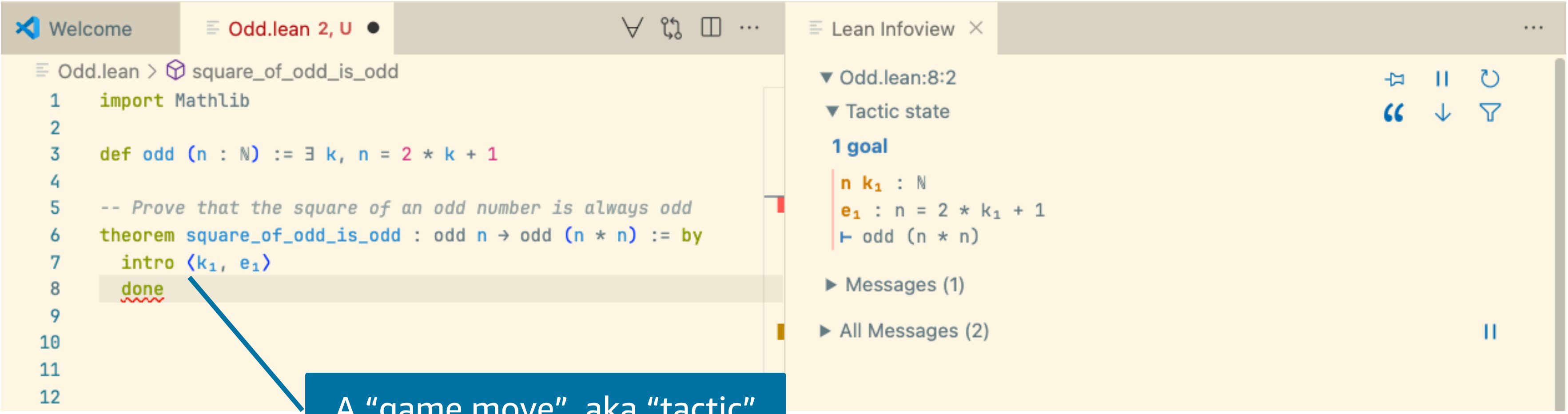


Theorem proving in Lean is an interactive game



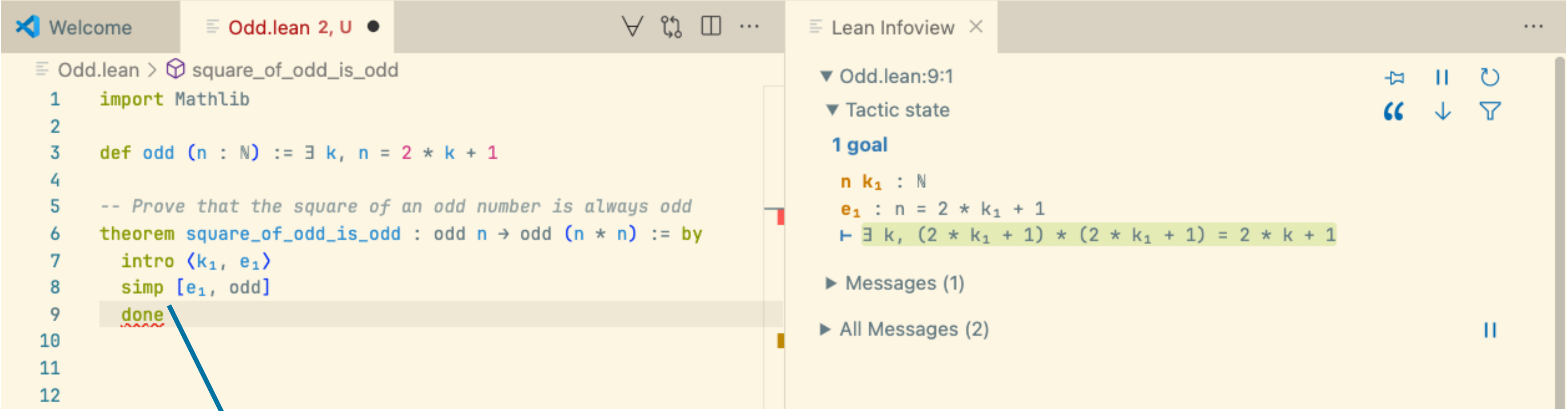
“You have written my favorite computer game”, Kevin Buzzard

Theorem proving in Lean is an interactive game



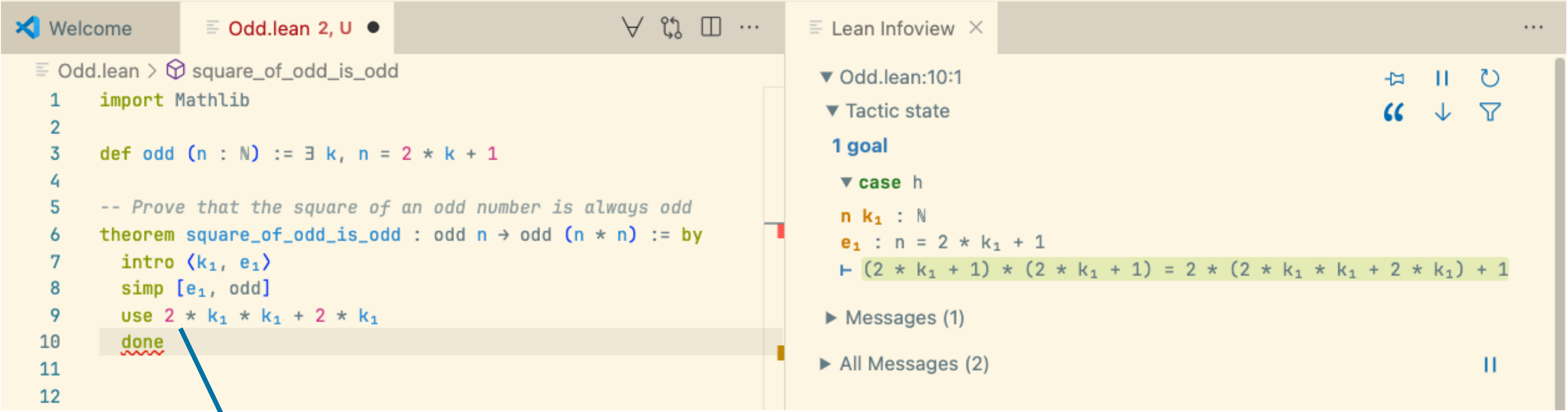
A “game move”, aka “tactic”

Theorem proving in Lean is an interactive game



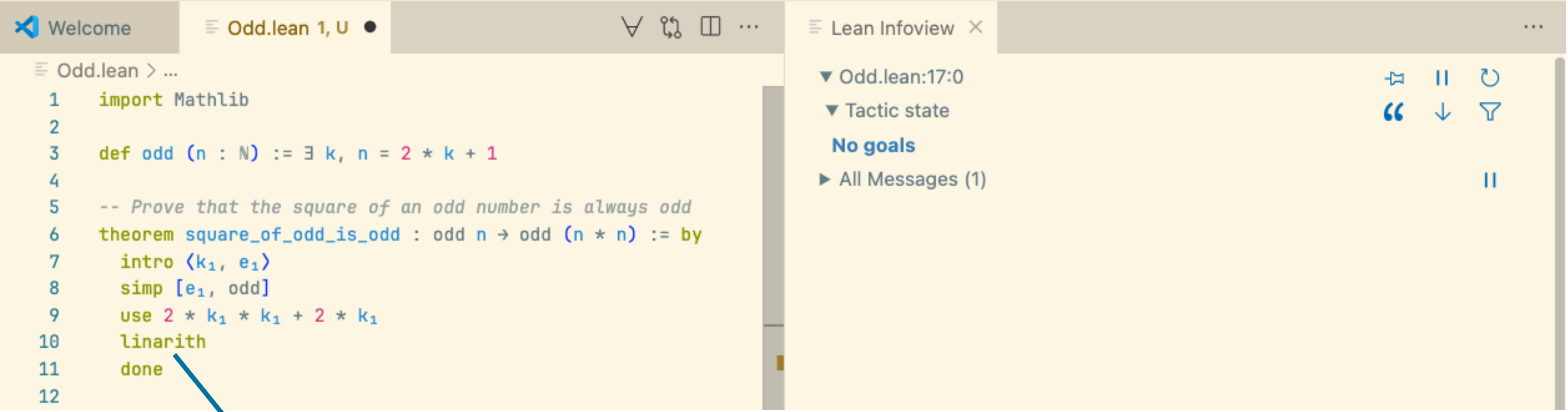
The “game move” `simp`, the simplifier, is one of the most popular moves in our game

Theorem proving in Lean is an interactive game



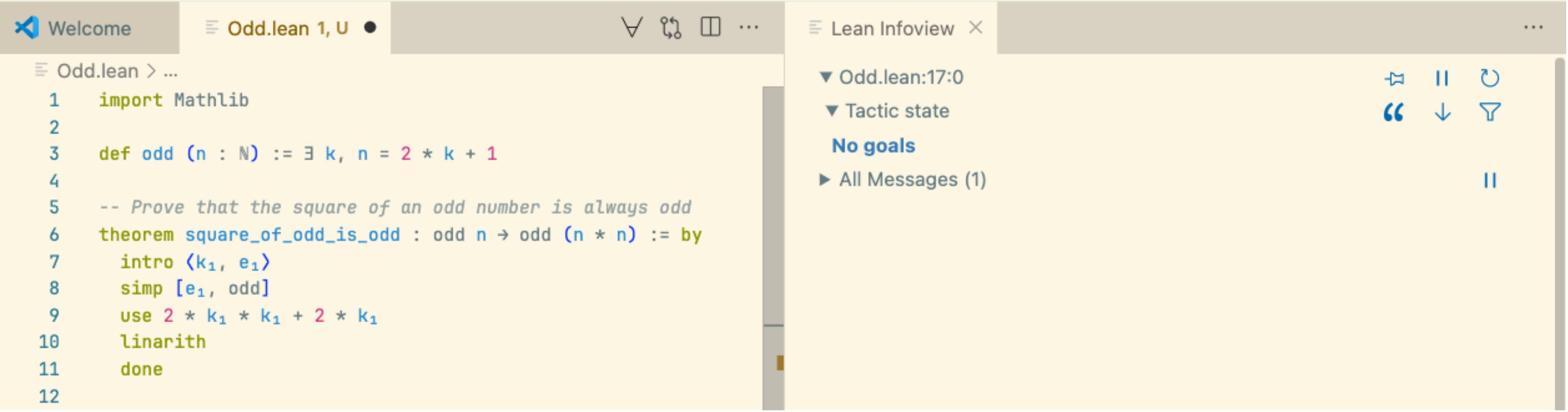
The "game move" `use` is the standard way of proving statements about existentials

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

Mathlib

The Lean Mathematical Library supports a wide range of projects.

It is an open-source **collaborative project** with over 500 contributors and 1.5M LoC.

“I’m investing time now so that somebody in the future can have that amazing experience”,

Heather Macbeth

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.

Mathematics

Software

AI

Mathematics

Preamble: the Perfectoid Spaces Project

Kevin Buzzard, Patrick Massot, Johan Commelin

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

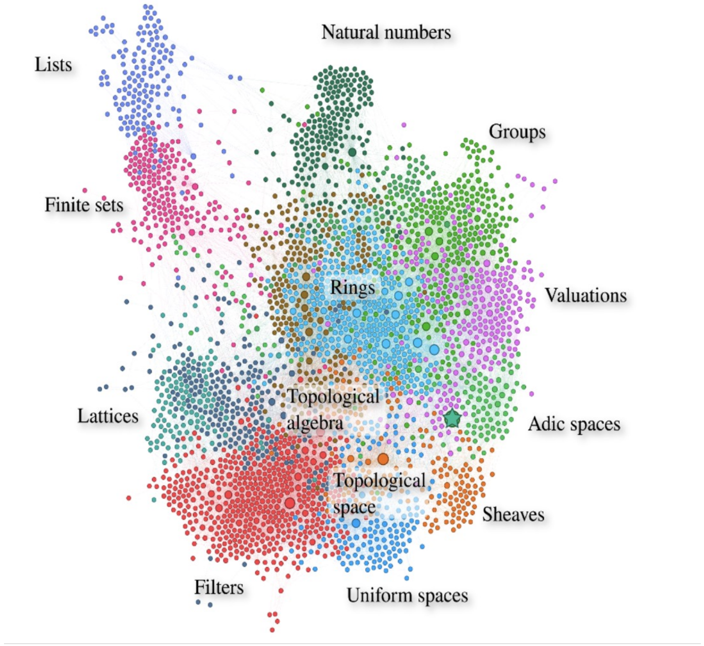
They translated Peter Scholze's definition into a form a computer can understand.

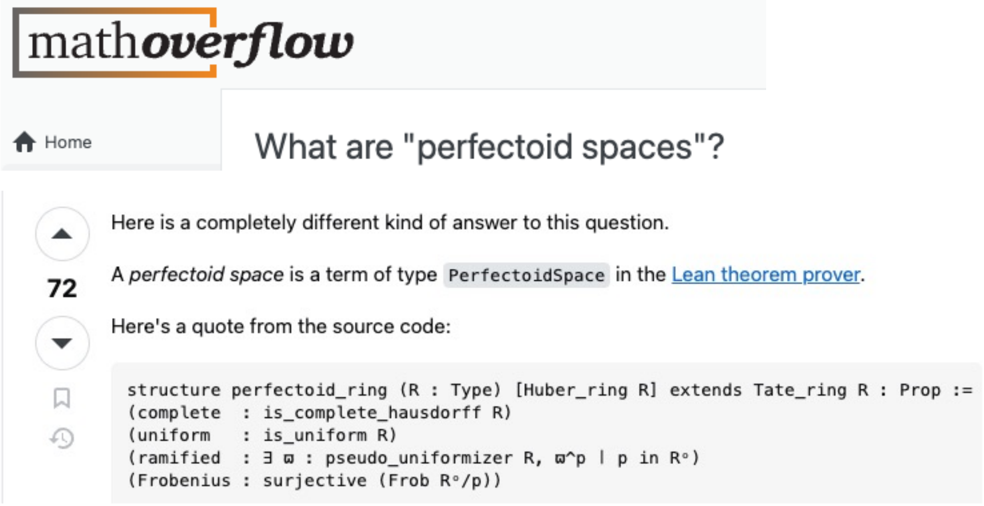
It not only achieved its goals but also demonstrated to the math community that **formal objects can be visualized and inspected with computer assistance**.

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

Preamble: the Perfectoid Spaces Project (cont.)

Kevin Buzzard, Patrick Massot, Johan Commelin





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_ring R : Prop :=
(complete  : is_complete_hausdorff R)
(uniform   : is_uniform R)
(ramified  : ∃ ϖ : pseudo_uniformizer R, ϖ^p ∣ p in R°)
(Frobenius : surjective (Frob R°/p))
```

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"*,
>
> Peter Scholze

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

Achieving the Unthinkable

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

Only the Beginning

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

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

Fermat’s Last Theorem for regular primes, Brasca et al., 2021-2023

Unit Fractions, Bloom and Mehta, 2022

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

Polynomial Freiman-Ruzsa Conjecture (PFR), Tao and Dillies, 2023

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

Carleson Project, van Doorn, 2024-ongoing

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

What did we learn?

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

The power of the **community**.

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

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.

AWS Clean Rooms Differential Privacy

Protect the privacy of your users with mathematically backed controls in a few steps

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**: https://reservoir.lean-lang.org/

What did we learn?

Machine-checkable proofs enable you to **code 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.

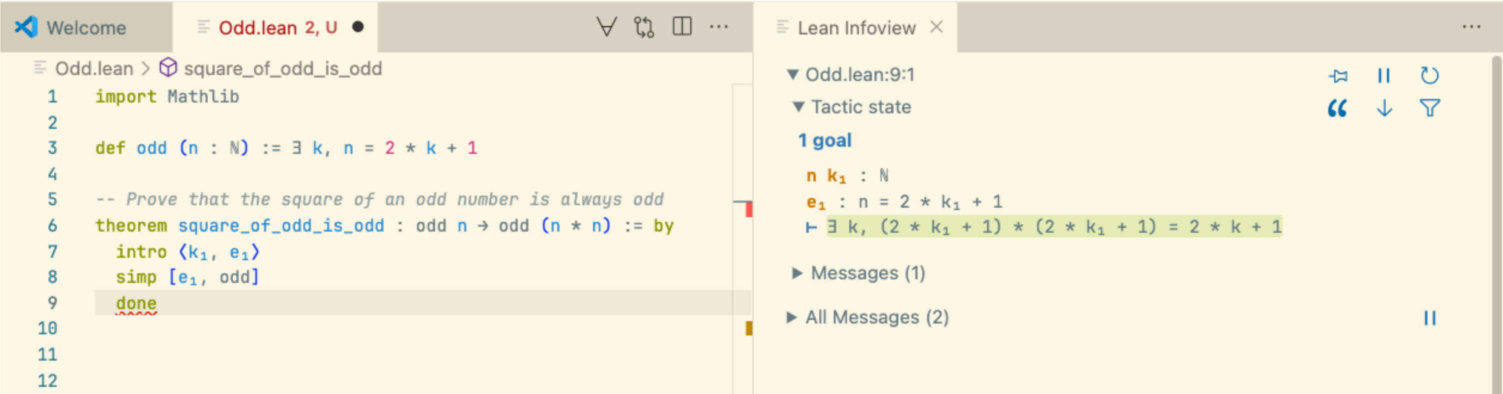
Machine-checkable proofs are the antidote to hallucinations.

Synthetic Data Generation

LLMs require **vast amounts of data** for training.

Lean mathematical libraries provide valuable, **correct-by-construction training data**.

Tools like lean-training-data, by **Kim Morrison**, extract data that includes the “game board” before and after each “move”.



Synthetic Data Generation

LLMs require **vast amounts of data** for training.

Lean mathematical libraries provide valuable, **correct-by-construction training data**.

Tools like lean-training-data, by **Kim Morrison**, extract data that includes the “game board” before and after each “move”.

We can search for alternative proofs and automatically verify their correctness.

AILean, a project led by **Soonho Kong** at AWS, uses Lean to generate **new synthetic theorems** that are correct by construction.

AI Proof Assistants

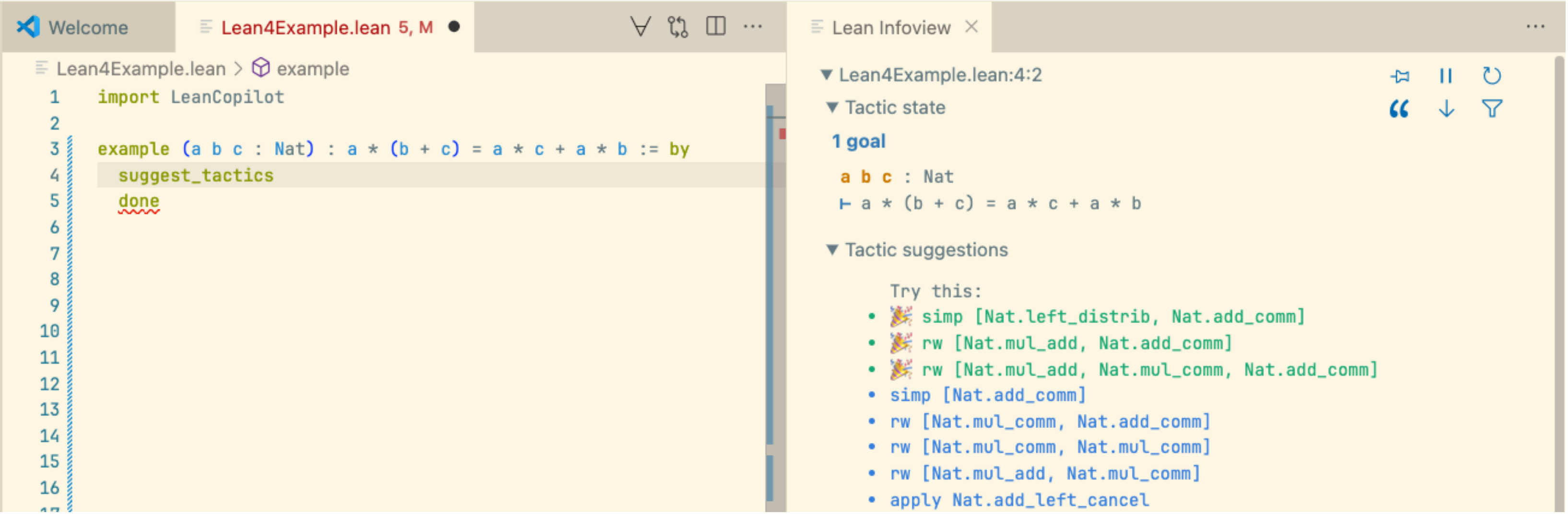
Several groups are developing AI that suggests the **next move**(s) in Lean's interactive proof game.

LeanDojo is an open-source project from Caltech, and everything (model, datasets, code) is open.

OpenAI and Meta AI have also developed AI assistants for Lean.

AI Proof Assistants

LeanCopilot is part of the LeanDojo project at Caltech. It uses the move (aka tactic) suggestion feature available in the Lean IDE.



Move Over, Mathematicians, Here Comes AlphaProof

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

Share full article | 47

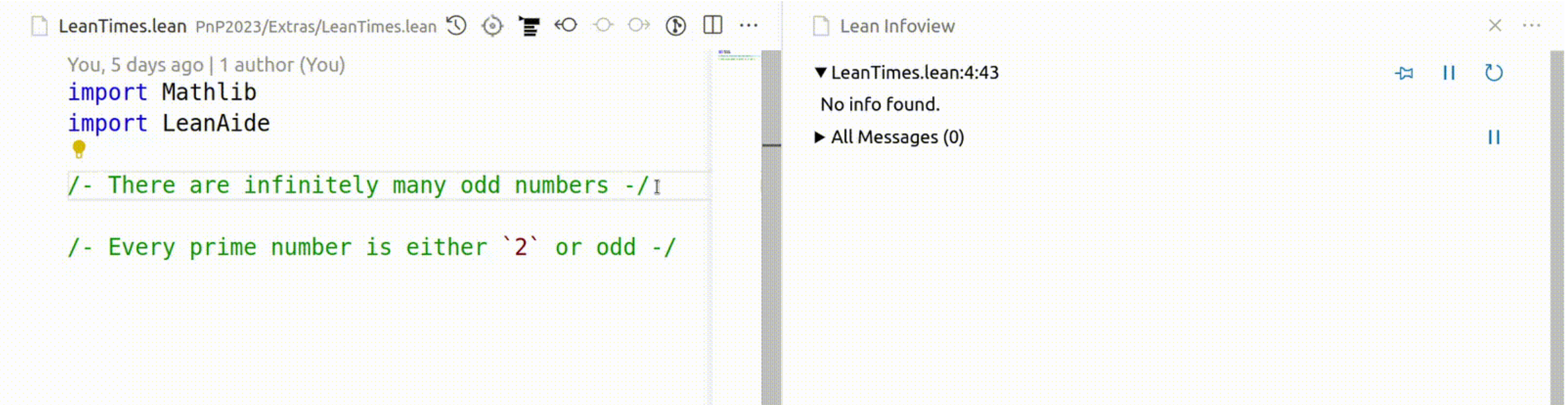
Ringing the gong at Google Deepmind's London headquarters, a ritual to celebrate each A.I. milestone, including its recent triumph of reasoning at the International Mathematical Olympiad. Google Deepmind

Auto-formalization

The process of converting natural language into a formal language like Lean.

It is much **easier to learn to read Lean than to write it**.

LeanAide is one of the auto-formalization tools available for Lean.



Specification-Oriented Programming

What if we could describe complex systems in plain language, and AI turned them into formal, provable code?

Specification-Oriented Programming

What if we could describe complex systems in plain language, and AI turned them into formal, provable code?

You describe what you want in natural language or pseudo-code.

AI auto-formalizes it in Lean.

You review the result and collaborate until it matches your intent.

AI synthesizes efficient, machine-checkable code and proofs.

What did we learn?

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

LLMs enable **auto-formalization**.

Lean can generate **synthetic correct by-construction datasets**.

Machine learning opens doors to **new proof search engines**.

Before we 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

Welcome | Odd.lean 1, U

Odd.lean > ...

```lean
import Mathlib

def odd (n : ℕ) := ∃ k, n = 2 * k + 1

-- Prove that the square of an odd number is always odd
theorem square_of_odd_is_odd : odd n → odd (n * n) := by
  intro ⟨k₁, e₁⟩
  simp [e₁, odd]
  use 2 * k₁ * k₁ + 2 * k₁
  linarith
  done
```

Lean Infoview

▼ Odd.lean:17:0

▼ Tactic state

No goals

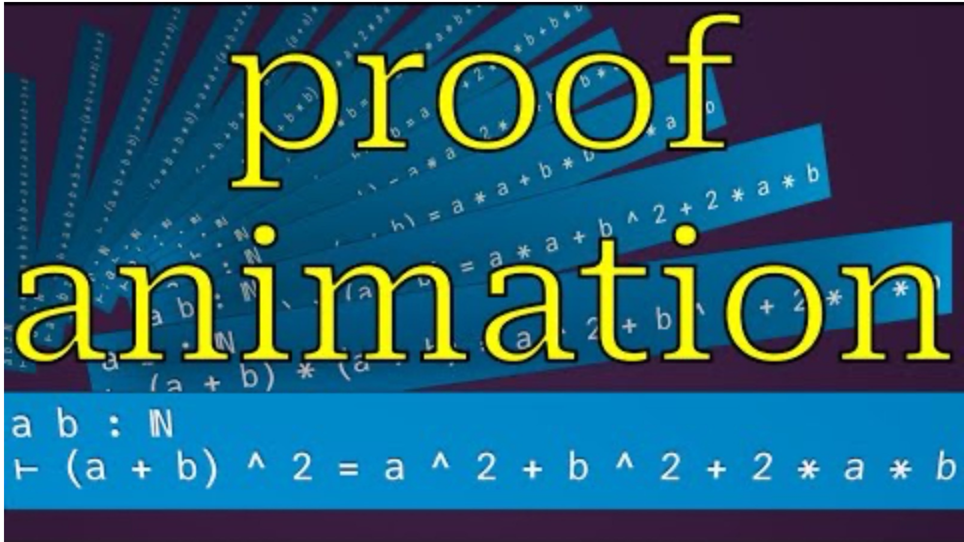
▶ All Messages (1)

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

You can use Lean to introspect its internal data

The tool lean-training-data is implemented in Lean itself. **It is a Lean package**.

A similar approach can be used to automatically generate proof animations.



Lean FRO: Shaping the Future of Lean Development

The Lean Focused Research Organization (FRO) is a non-profit dedicated to Lean's development.

Founded in **August 2023**, the organization has 14 members.

Its mission is to enhance critical areas: **scalability**, **usability**, **documentation**, and **proof automation**.

It must reach **self-sustainability in August 2028** and become the **Lean Foundation**.

Philanthropic support is gratefully acknowledged from the **Simons Foundation**, the **Alfred P. Sloan Foundation**, **Richard Merkin**, and **Founders Pledge**.

How can I contribute?

Help building Mathlib.

Want to explore new ways to collaborate and use machine assistance? Contribute to Prof. Tao's new project.

Want to engage with the vibrant Lean community? Join our Zulip channel.

Interested in verified tensor compilers? Contribute to the verified StableHLO project.

Want to contribute to a large formalization project? Join the FLT formalization project.

Start your own open-source Lean project! Your package will be available on our registry Reservoir.

Conclusion

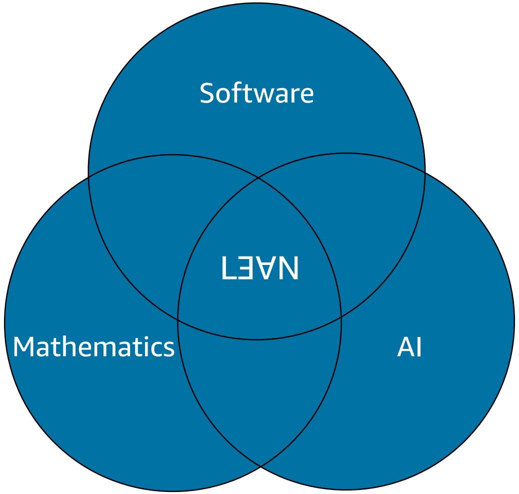
Lean is an **efficient programming language** and **proof assistant**.

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

https://leanprover.zulipchat.com/
x: @leanprover
LinkedIn: Lean FRO
Mastodon: @leanprover@functional.cafe
#leanlang, #leanprover

https://www.lean-lang.org/