AI for Mathematics

Jeremy Avigad

Department of Philosophy
Department of Mathematical Sciences
Hoskinson Center for Formal Mathematics
Carnegie Mellon University

Overview

Today, I will discuss:

- Interactive Proof Assistants and Formalization
- Automated Reasoning and Symbolic AI
- Machine Learning and Neural AI

I will refer to these collectively as “AI for Mathematics.”

I will also tell you about a remarkable programming language and proof assistant called *Lean*.

Overview

The digital library, Mathlib, has >160K theorems and >1.5M lines of code.

High-profile formalizations of contemporary research in Lean:

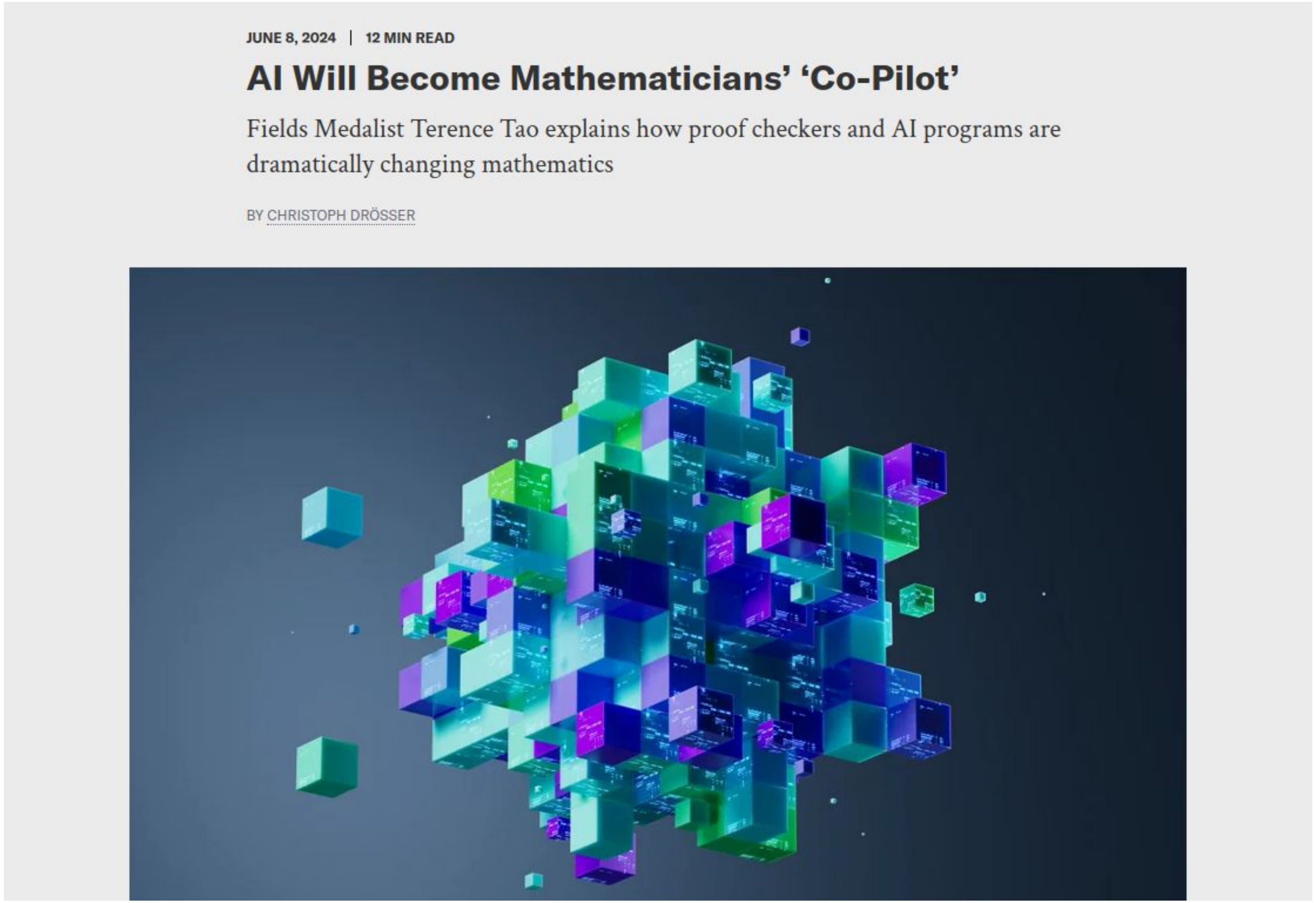
- the Polynomial Freiman–Ruzsa Project
- a strengthening of Carleson's Theorem

High-profile uses of automated reasoning for mathematics:

- Keller's conjecture
- refutation of the Kaplansky unit conjecture

High-profile advances in machine learning for mathematics:

- results in representation theory, knot theory, graph theory
- DeepMind's AlphaProof and AlphaGeometry



JUNE 8, 2024 | 12 MIN READ

AI Will Become Mathematicians' 'Co-Pilot'

Fields Medalist Terence Tao explains how proof checkers and AI programs are dramatically changing mathematics

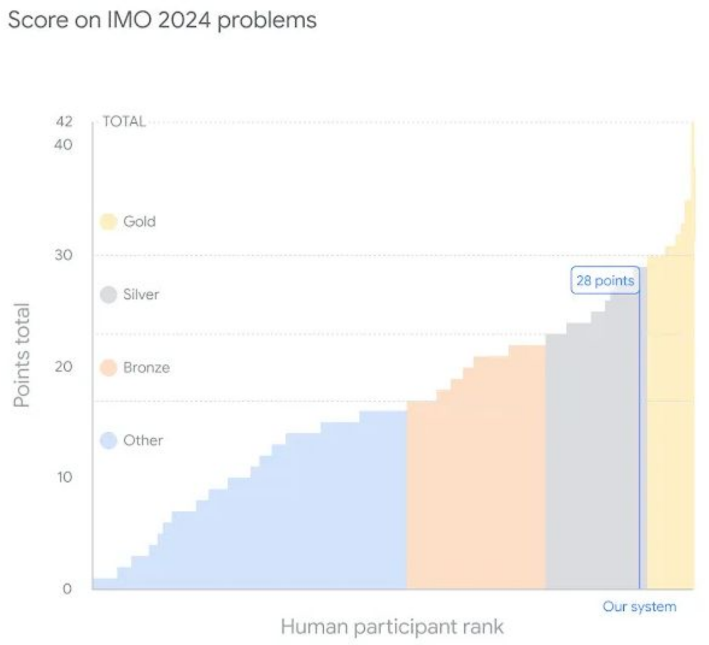
BY CHRISTOPH DRÖSSER

Tao’s Predictions

“I think in the future, instead of typing up our proofs, we would explain them to some GPT. And the GPT will try to formalize it in Lean as you go along. If everything checks out, the GPT will [essentially] say, ‘Here’s your paper in LaTeX; here’s your Lean proof. If you like, I can press this button and submit it to a journal for you.’”

Tao’s Predictions

“There could be collaborative projects where we don’t know how to prove the whole thing. But people have ideas on how to prove little pieces, and they formalize that and try to put them together. In the future, I could image a big theorem being proven by a combination of 20 people and a bunch of AIs each proving little things. And over time, they will get connected, and you can create some wonderful thing. That will be great.”



Graph showing performance of our AI system relative to human competitors at IMO 2024. We earned 28 out of 42 total points, achieving the same level as a silver medalist in the competition.

AlphaProof: a formal approach to reasoning

AlphaProof and AlphaGeometry

“It’s a fairly safe bet that if Google DeepMind can solve at least some hard I.M.O. problems, then a useful research tool can’t be all that far away.”

– Timothy Gowers, Rouse Ball Professor of Mathematics at the University of Cambridge

A.I. Is Coming for Mathematics, Too

For thousands of years, mathematicians have adapted to the latest advances in logic and reasoning. Are they ready for artificial intelligence?

Share full article

Move Over, Mathematicians, Here Comes AlphaProof

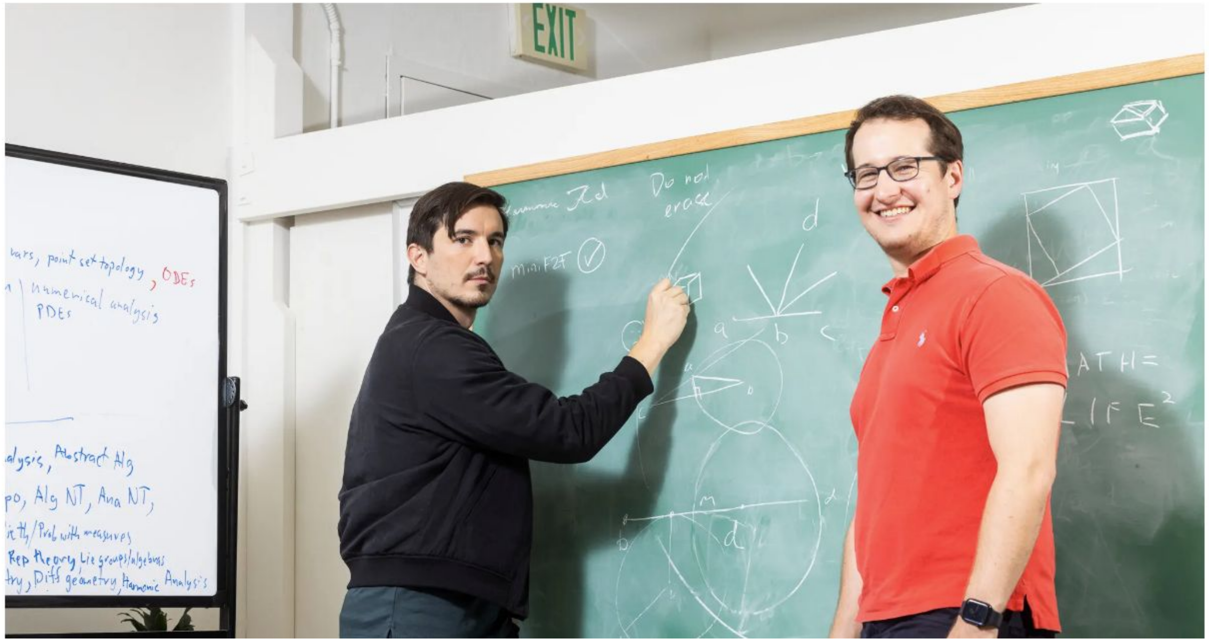
A.I. is getting good at math — and might soon make a worthy collaborator for humans.

Is Math the Path to Chatbots That Don't Make Stuff Up?

Chatbots like ChatGPT get stuff wrong. But researchers are building new A.I. systems that can verify their own math — and maybe more.

Listen to this article · 6:40 min Learn more

Share full article



Overview

These technologies will impact mathematics:

- verification of mathematical results and mathematical computation
- communication and collaboration
- mathematical reference and search
- exploration and discovery of new mathematics
- teaching and learning

I will discuss the technologies and their importance.

Interactive Theorem Provers

We have known since the early twentieth century that mathematics can be formalized:

- Mathematical statements can be expressed in formal languages, with precise grammar.
- Theorems can be proved from formal axioms, using prescribed rules of inference.

With the help of computational proof assistants, this can be carried out in practice.

In many systems, the formal proof can be extracted and verified independently.

Interactive Theorem Provers

Some proof assistants for mathematics:

- Mizar (1973, set theory)
- Isabelle (1986, simple type theory)
- Rocq (1989, dependent type theory)
- HOL Light (1994, simple type theory)
- Lean (2013, dependent type theory)

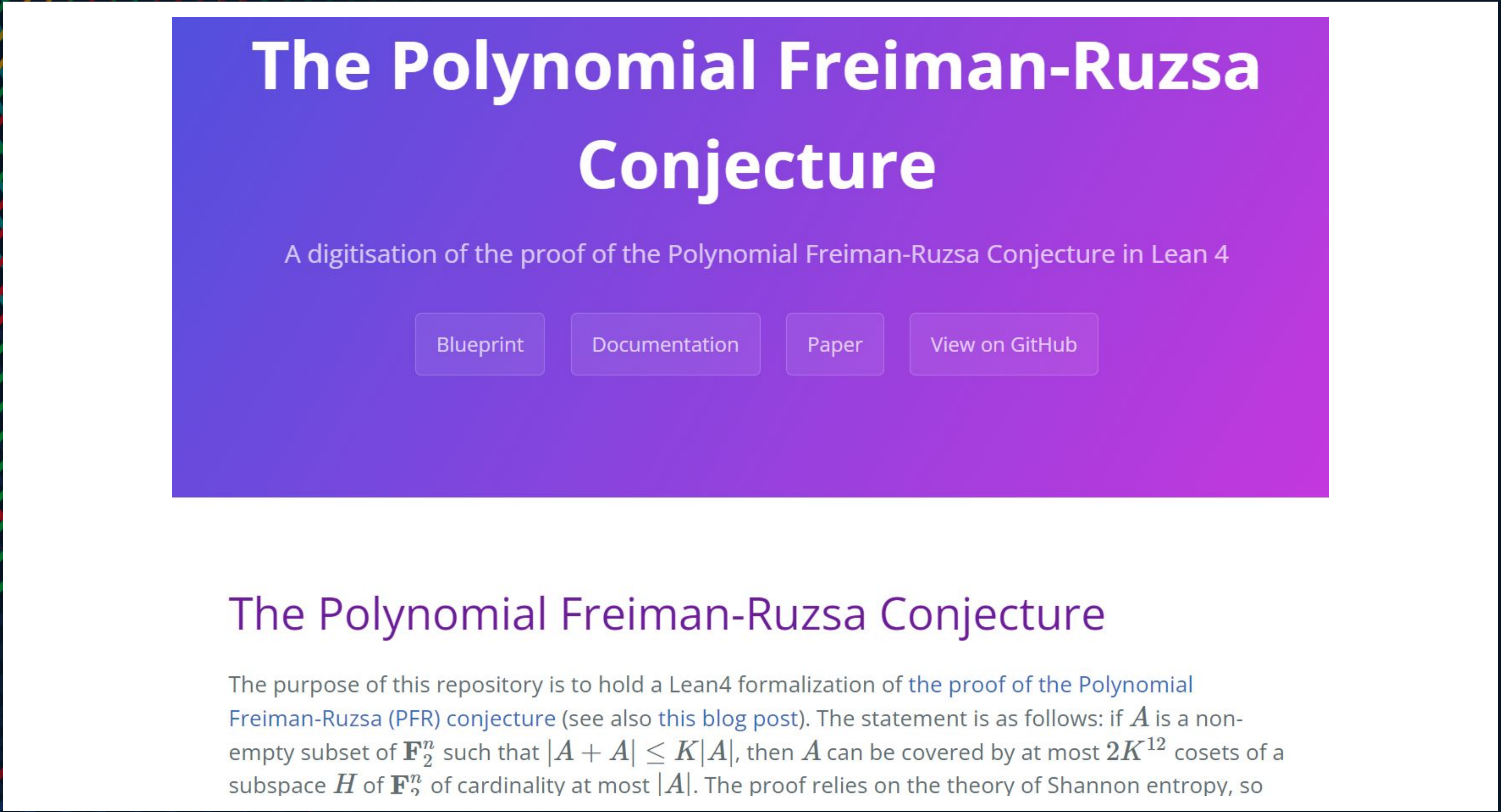
Proof assistants are now commonly used for hardware and software verification.

I will give a demonstration of Lean.

Lean and Mathlib

Few mathematicians were using formal methods in 2017. Since then:

- Mathlib has more than 1.6 million lines of code.
- The Lean Zulip channel more than members, about 850 active in any two-week period.
- There are have been a number of celebrated successes.
- There are interesting collaborative projects.
- There have been a number of articles in the general press.
- There are several meetings and workshops related to Lean.
- There is growing interest and enthusiasm in the mathematical community.

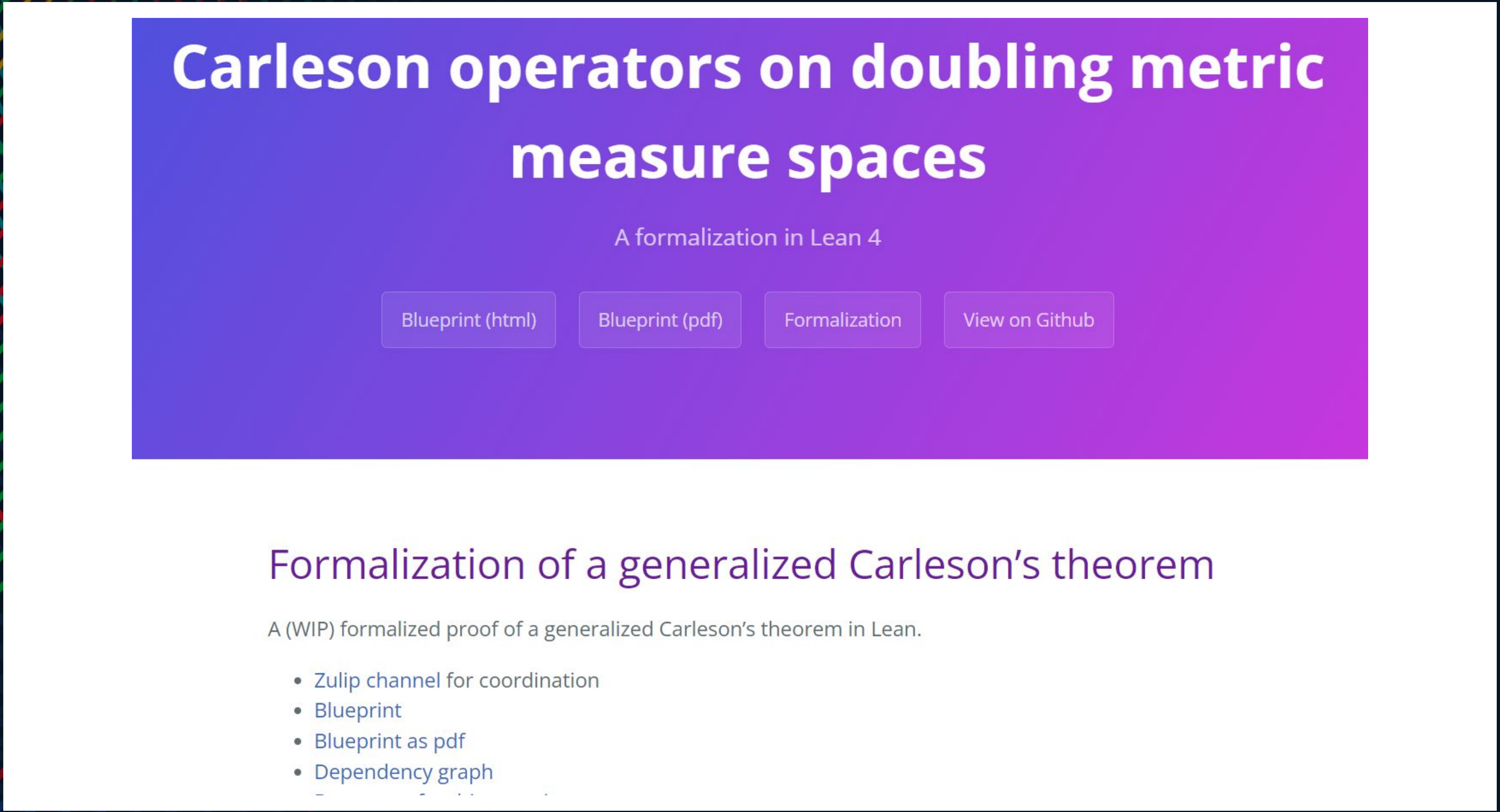


The Polynomial Freiman-Ruzsa Conjecture

A digitisation of the proof of the Polynomial Freiman-Ruzsa Conjecture in Lean 4

The Polynomial Freiman-Ruzsa Conjecture

The purpose of this repository is to hold a Lean4 formalization of the proof of the Polynomial Freiman-Ruzsa (PFR) conjecture (see also this blog post). The statement is as follows: if A is a non-empty subset of $\mathbf{F}_2^n$ such that $|A+A| \leq K|A|$, then A can be covered by at most $2K^{12}$ cosets of a subspace H of $\mathbf{F}_2^n$ of cardinality at most $|A|$. The proof relies on the theory of Shannon entropy, so



Carleson operators on doubling metric measure spaces

A formalization in Lean 4

Formalization of a generalized Carleson's theorem

A (WIP) formalized proof of a generalized Carleson's theorem in Lean.

- Zulip channel for coordination
- Blueprint
- Blueprint as pdf
- Dependency graph

Overview

I said I would discuss:

- Interactive Proof Assistants and Formalization
- Automated Reasoning and Symbolic AI
- Machine Learning and Neural AI

Automated Reasoning

By *automated reasoning*, I mean the use of computational, logic-based methods to carry out logical and mathematical inferences.

Some problems are *decidable*, and some aren’t.

Two types of procedures:

- **Decision procedures:** decide whether an inference is valid / true in an interpretation (and possibly produce a proof)
- **Search procedures:** search for a proof or justification

Decision Procedures

Early history:

- In 1915, Löwenheim proved the decidability of **monadic first-order logic**.
- Presburger presented his decision procedure for **linear integer arithmetic** in 1929.
- Tarski had a decision procedure for **real closed fields** in 1930.

Decision Procedures

Today:

- propositional logic (SAT solvers)
- *ground* equational reasoning
- linear integer and real arithmetic (equations and inequalities)
- real polynomial arithmetic (equations and inequalities)
- symbolic calculation (e.g. equations in an arbitrary ring)

Satisfiability modulo theories (SMT) solvers combine these.

Keller’s Conjecture

In 1930, Ott-Heinrich Keller conjectured that for every *n*, any tiling of *n*-dimensional space with unit *n*-dimensional cubes has to have at least two cubes that fully share a face.

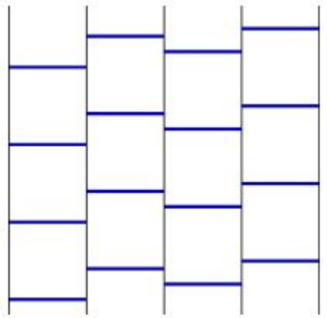


Figure 1: Two-dimensional tiling

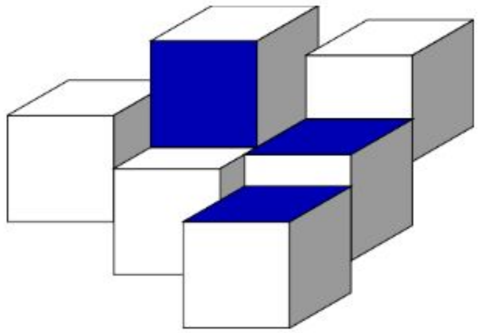


Figure 2: Three-dimensional tiling

Keller’s Conjecture

History:

- In 1940, Perron showed that the conjecture is **true** up to dimension 6.
- In 1992, Lagarias and Shor showed that the conjecture is **false** in dimensions 10 and up.
- In 2002, Mackey showed that it is **false** in dimensions 8 and 9.
- In 2020, Brakensiek, Heule, Mackey, and Narváez showed that it is **true** in dimension 7.

Keller’s Conjecture

In 1990, Corradi and Szabo showed that the conjecture is true if and only if there are no cliques of a certain size in certain associated graphs, now known as *Keller graphs*.

Brakensiek, Heule, Mackey, and Narváez used a SAT solver, with additional reductions and symmetry breaking, to establish this.

The images are taken from their web page.

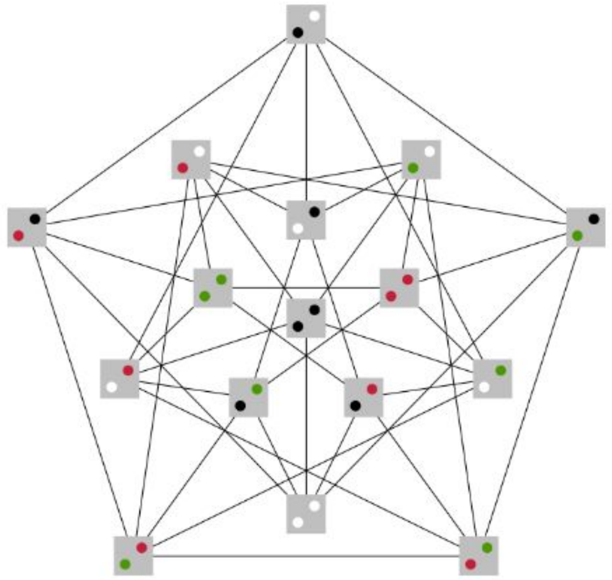


Figure 3: a Keller graph

Kaplansky's Unit Conjecture

In 1940, Higman conjectured that if K is a field and G is a torsion-free group, then the group ring K[G] has no nontrivial units.

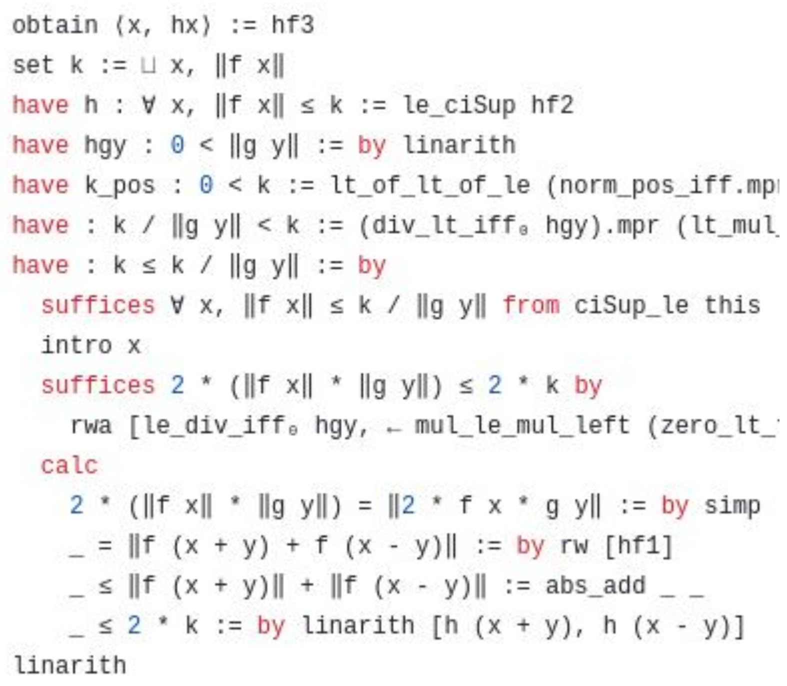
In 2021, Giles Gardam gave a counterexample with K = $\mathbf{F}_2$.

In a talk, Solving Semidecidable Problems in Group Theory, he explains how he used a SAT solver to find the counterexample.

He has since used a Gröbner basis algorithm to find a counterexample with K = $\mathbf{C}$.

Decision Procedures

Decision procedures like linear arithmetic are commonly used to help formalize mathematics.



```
obtain ⟨x, hx⟩ := hf3
set k := ⨆ x, ‖f x‖
have h : ∀ x, ‖f x‖ ≤ k := le_ciSup hf2
have hgy : 0 < ‖g y‖ := by linarith
have k_pos : 0 < k := lt_of_lt_of_le (norm_pos_iff.mp
have : k / ‖g y‖ < k := (div_lt_iff₀ hgy).mpr (lt_mul_
have : k ≤ k / ‖g y‖ := by
  suffices ∀ x, ‖f x‖ ≤ k / ‖g y‖ from ciSup_le this
  intro x
  suffices 2 * (‖f x‖ * ‖g y‖) ≤ 2 * k by
    rwa [le_div_iff₀ hgy, ← mul_le_mul_left (zero_lt_
  calc
    2 * (‖f x‖ * ‖g y‖) = ‖2 * f x * g y‖ := by simp
    _ = ‖f (x + y) + f (x - y)‖ := by rw [hf1]
    _ ≤ ‖f (x + y)‖ + ‖f (x - y)‖ := abs_add _ _
    _ ≤ 2 * k := by linarith [h (x + y), h (x - y)]
linarith
```

Automated Reasoning

Early history:

- Martin Davis implemented Presburger's decision procedure at the IAS in 1954.
- Allen Newell, Herbert Simon, and Cliff Shaw introduced the *Logic Theorist* in 1956.
- Henry Gelernter, J. R. Hansen, and Donald Loveland published an article on the *Geometry Machine* in 1960.
- Hao Wang implemented a prover for predicate logic in 1958.
- Davis and Hilary Putnam introduced the propositional resolution rule in 1960.
- John Alan Robinson introduced a unification algorithm in 1965.

Search Procedures

Larry Paulson, Jasmin Blanchette, and colleagues developed a powerful *sledgehammer* for the *Isabelle* proof assistant.

Given an inference to perform, it:

- selects about 200 potentially relevant facts from the mathematical library (from tens of thousands),
- sends the problem to an automated theorem prover,
- reconstructs a formally checked proof.

We are currently working on a similar tool for Lean.

Another tool, called *Aesop*, helps prove theorems automatically.

Overview

I said I would discuss:

- Interactive Proof Assistants and Formalization
- Automated Reasoning and Symbolic AI
- Machine Learning and Neural AI

Machine Learning for Mathematics

The last few years have brought exciting progress in machine learning, including deep learning and generative AI.

Two ways machine learning can be applied to mathematics:

- using formal representations of mathematics
- not using formal representations of mathematics
  - searching for interesting mathematical objects
  - detecting patterns in computational data.

Machine Learning for Mathematics

Direct applications:

- **Knot invariants** (sensitivity analysis): Lackenby and Juhász with DeepMind.
- **Kazhdan-Lustig polynomials** and **Bruhat intervals** (sensitivity analysis): Williamson with DeepMind.
- **Counterexamples in graph theory** (reinforcement learning): Wagner
- **Ribbons in knot theory** (reinforcement learning): Gukov, Halverson, Manolescu, and Ruehle
- **Approximating singularities** in solutions to PDEs (neural networks): Buckmaster, Gómez-Serrano, Lai, Wang
- Finding constructions of **large cap sets** (LLMs): Ellenberg with DeepMind

Finding Counterexamples in Graph Theory

For example, Adam Wagner used reinforcement learning to find counterexamples to dozens of conjectures in graph theory.

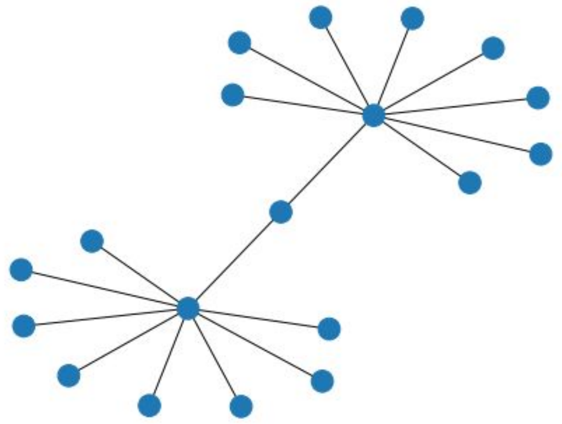
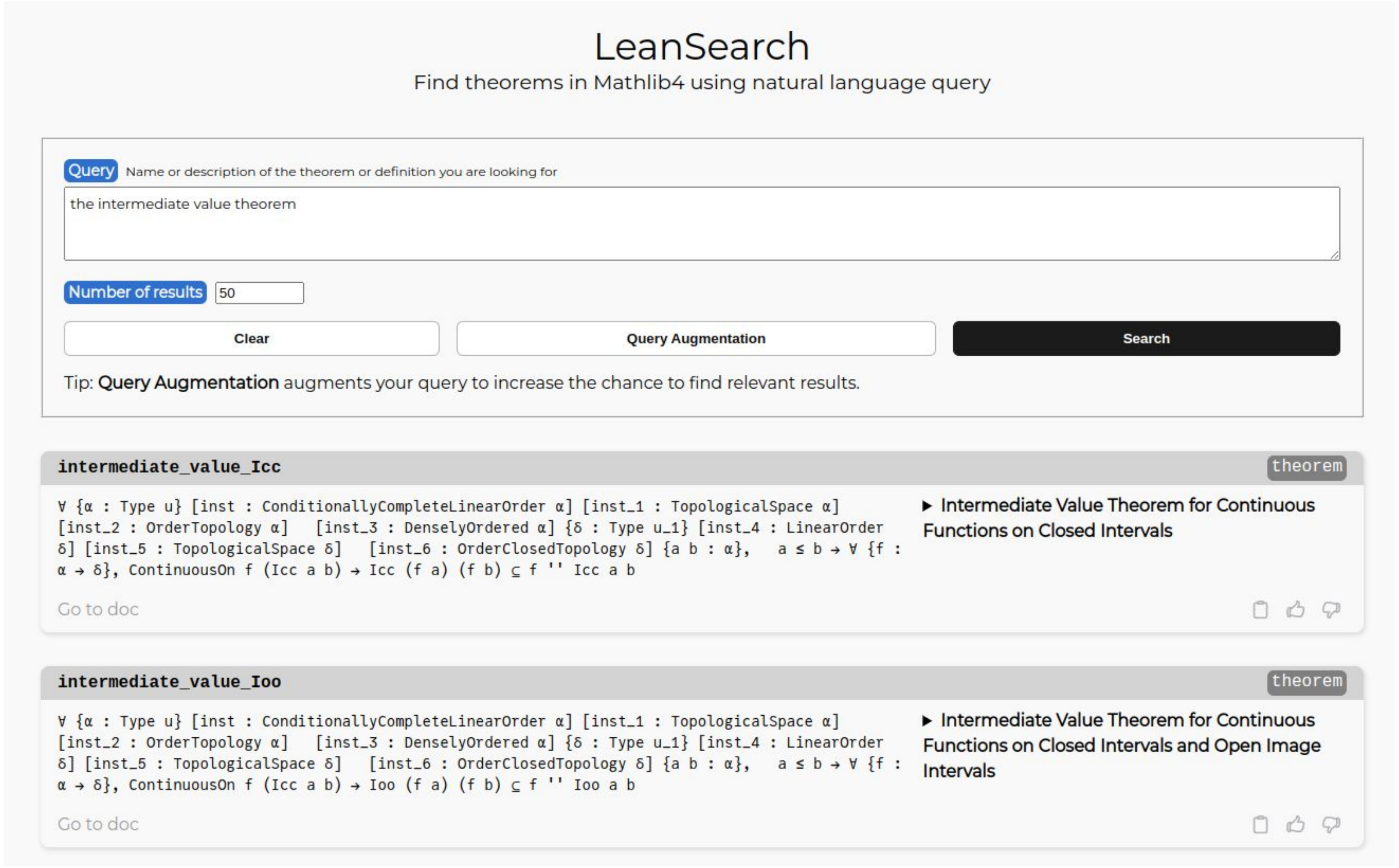


Figure 4: A graph on 19 vertices satisfying $\lambda_1 + \mu < \sqrt{n-1} + 1$.

Machine Learning and Formal Methods

Several efforts in machine learning for mathematics build on proof assistants like Lean:

- helping formalize theorems
  - search engines
  - premise selection
  - copilots
  - automated reasoning tools
- discovering proofs of theorems
  - Harmonic / Aristotle
  - DeepSeek
  - DeepMind / AlphaProof



LeanSearch

Find theorems in Mathlib4 using natural language query

Query Name or description of the theorem or definition you are looking for

the intermediate value theorem

Number of results 50

Tip: **Query Augmentation** augments your query to increase the chance to find relevant results.

intermediate_value_Icc theorem

```
∀ {α : Type u} [inst : ConditionallyCompleteLinearOrder α] [inst_1 : TopologicalSpace α]
[inst_2 : OrderTopology α]  [inst_3 : DenselyOrdered α] {δ : Type u_1} [inst_4 : LinearOrder
δ] [inst_5 : TopologicalSpace δ]  [inst_6 : OrderClosedTopology δ] {a b : α},  a ≤ b → ∀ {f :
α → δ}, ContinuousOn f (Icc a b) → Icc (f a) (f b) ⊆ f '' Icc a b
```

▶ Intermediate Value Theorem for Continuous Functions on Closed Intervals

Go to doc

intermediate_value_Ioo theorem

```
∀ {α : Type u} [inst : ConditionallyCompleteLinearOrder α] [inst_1 : TopologicalSpace α]
[inst_2 : OrderTopology α]  [inst_3 : DenselyOrdered α] {δ : Type u_1} [inst_4 : LinearOrder
δ] [inst_5 : TopologicalSpace δ]  [inst_6 : OrderClosedTopology δ] {a b : α},  a ≤ b → ∀ {f :
α → δ}, ContinuousOn f (Icc a b) → Ioo (f a) (f b) ⊆ f '' Ioo a b
```

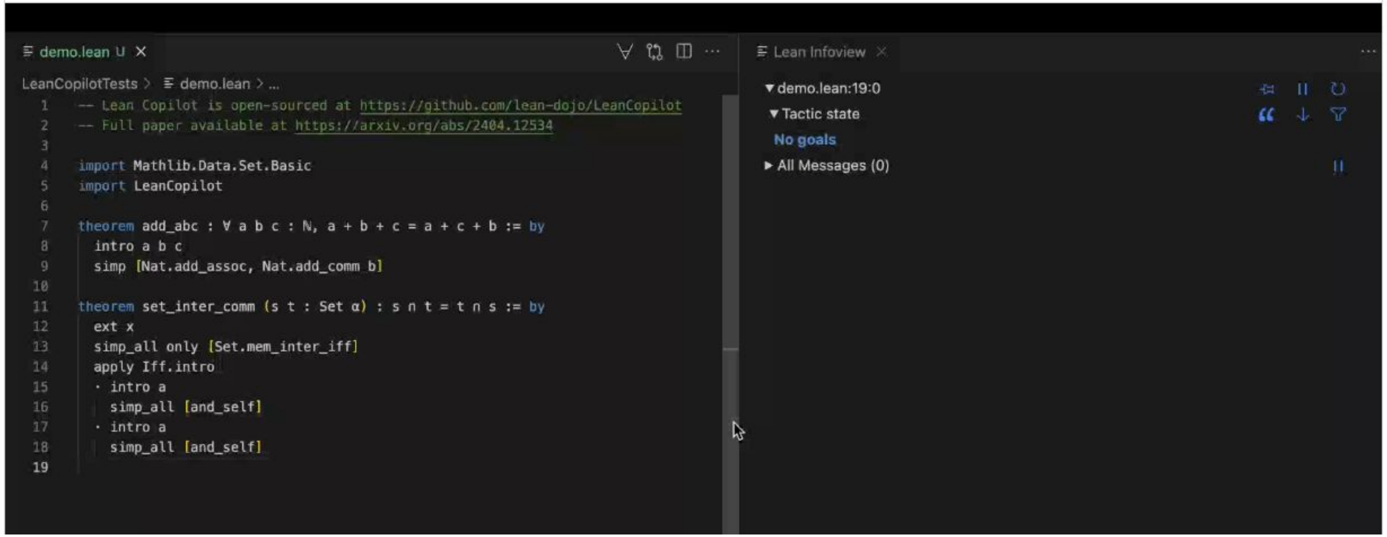
▶ Intermediate Value Theorem for Continuous Functions on Closed Intervals and Open Image Intervals

Go to doc

Lean Copilot: LLMs as Copilots for Theorem Proving in Lean

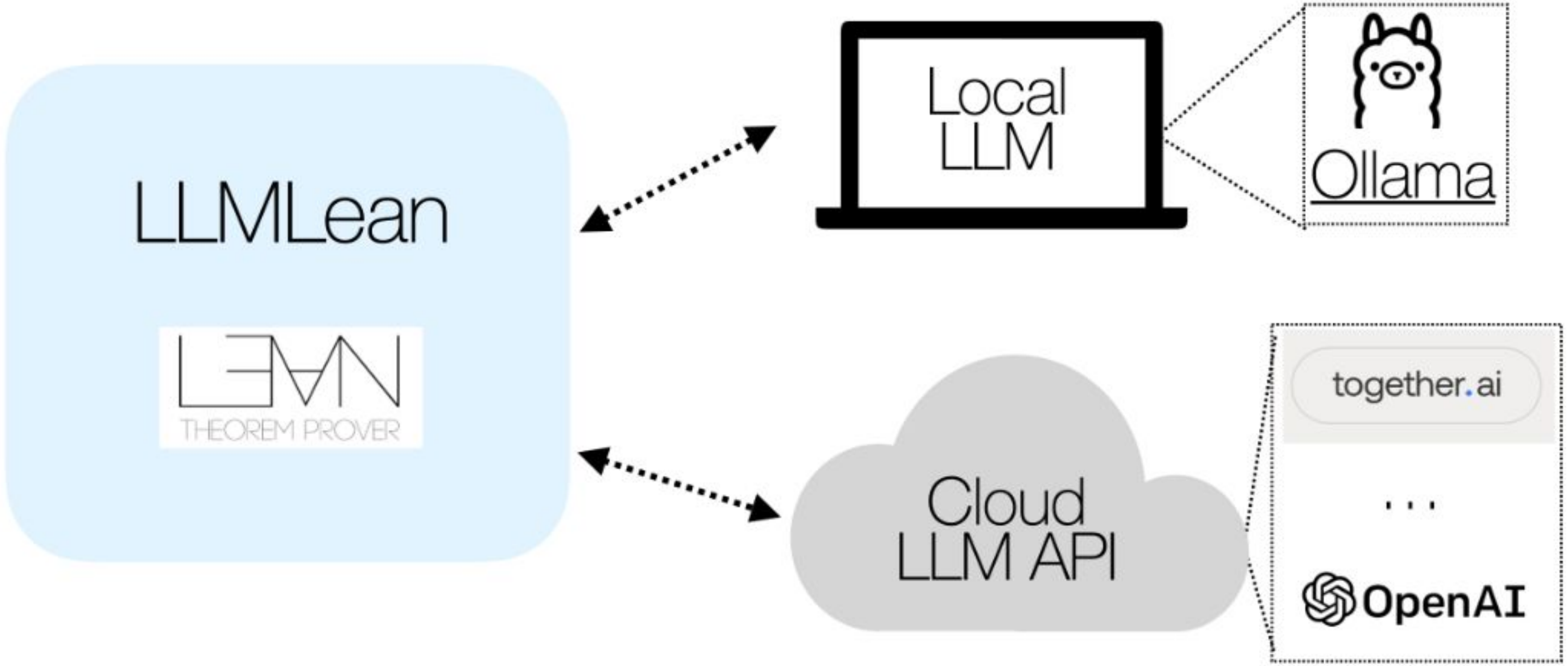
Lean Copilot allows large language models (LLMs) to be used in Lean for proof automation, e.g., suggesting tactics/premises and searching for proofs. You can use our built-in models from LeanDojo or bring your own models that run either locally (w/ or w/o GPUs) or on the cloud.

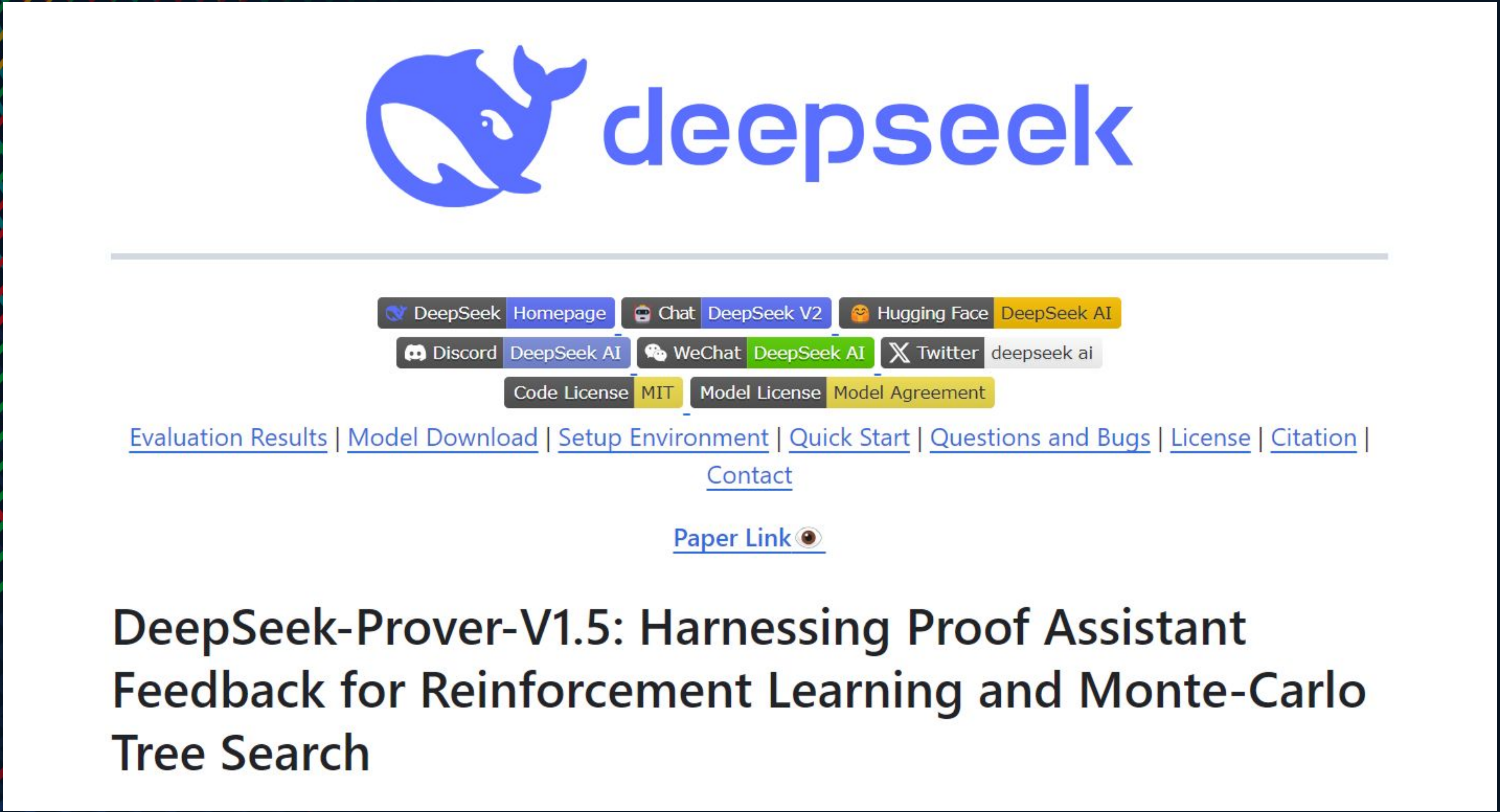
Lean.Copilot.Demo.mp4



```
-- Lean Copilot is open-sourced at https://github.com/lean-dojo/LeanCopilot
-- Full paper available at https://arxiv.org/abs/2404.12534

import Mathlib.Data.Set.Basic
import LeanCopilot

theorem add_abc : ∀ a b c : ℕ, a + b + c = a + c + b := by
  intro a b c
  simp [Nat.add_assoc, Nat.add_comm b]

theorem set_inter_comm (s t : Set α) : s ∩ t = t ∩ s := by
  ext x
  simp_all only [Set.mem_inter_iff]
  apply Iff.intro
  · intro a
    simp_all [and_self]
  · intro a
    simp_all [and_self]
```

No goals

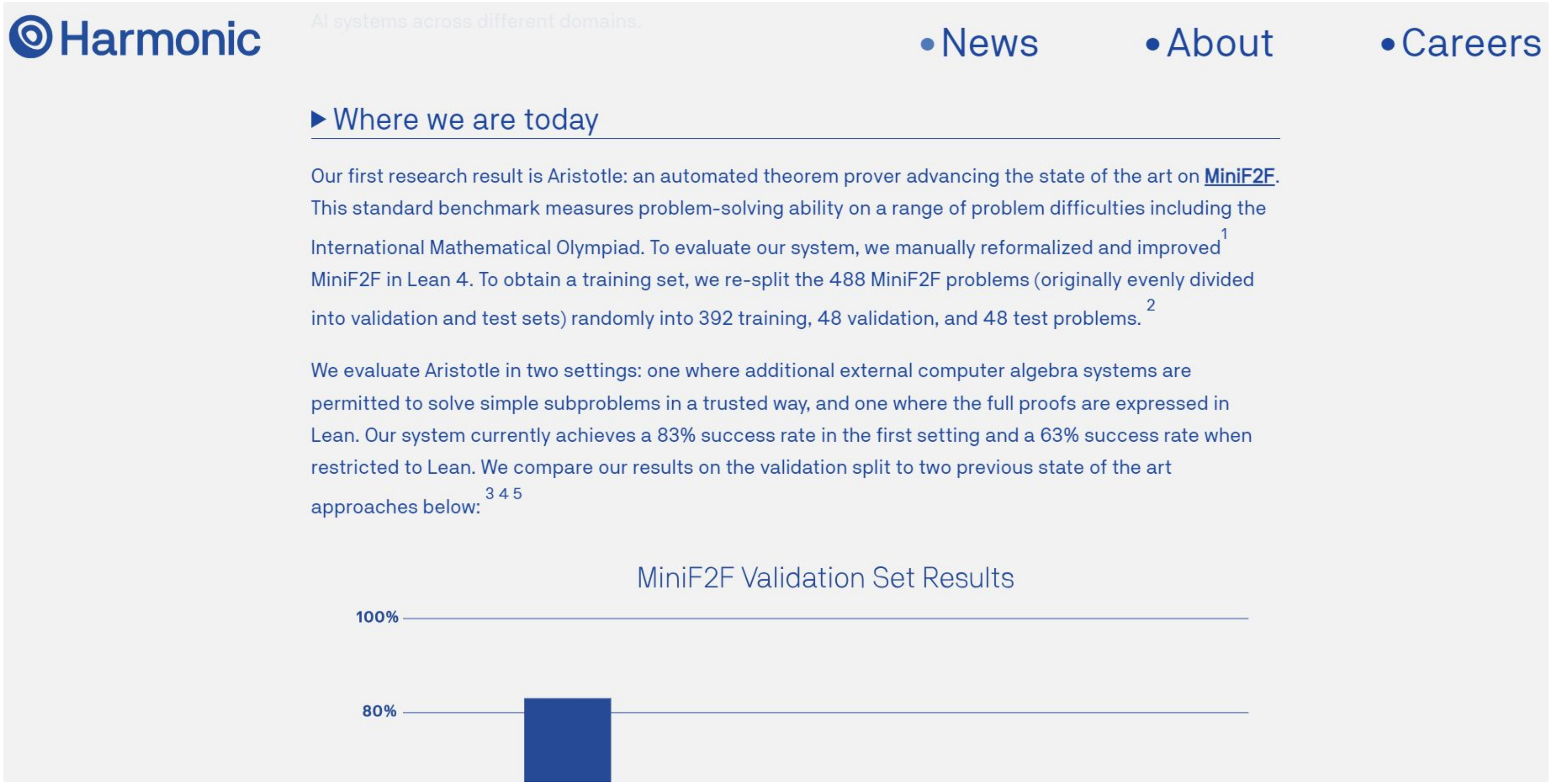

LLMLean
LEAN
THEOREM PROVER
Local
LLM
Ollama
Cloud
LLM API
together.ai
...
OpenAI



Evaluation Results | Model Download | Setup Environment | Quick Start | Questions and Bugs | License | Citation | Contact

Paper Link👁️

DeepSeek-Prover-V1.5: Harnessing Proof Assistant Feedback for Reinforcement Learning and Monte-Carlo Tree Search



Harmonic

AI systems across different domains.

• News • About • Careers

▶ Where we are today

Our first research result is Aristotle: an automated theorem prover advancing the state of the art on MiniF2F. This standard benchmark measures problem-solving ability on a range of problem difficulties including the International Mathematical Olympiad. To evaluate our system, we manually reformalized and improved[1] MiniF2F in Lean 4. To obtain a training set, we re-split the 488 MiniF2F problems (originally evenly divided into validation and test sets) randomly into 392 training, 48 validation, and 48 test problems.[2]

We evaluate Aristotle in two settings: one where additional external computer algebra systems are permitted to solve simple subproblems in a trusted way, and one where the full proofs are expressed in Lean. Our system currently achieves a 83% success rate in the first setting and a 63% success rate when restricted to Lean. We compare our results on the validation split to two previous state of the art approaches below:[3 4 5]

MiniF2F Validation Set Results

Machine Learning and Formal Methods

Machine learning and large language models raise a host of concerns:

- reliability
- explainability
- alignment.

Mathematical language, concepts, and rigor are needed to supplement these.

There is a growing awareness that a combination of neural and symbolic methods is the key to general intelligence, and that AI for mathematics is the frontier.

Machine Learning and Formal Methods

"We may hope that machines will eventually compete with men in all purely intellectual fields. But which are the best ones to start with? Even this is a difficult decision. Many people think that a very abstract activity, like the playing of chess, would be best. It can also be maintained that it is best to provide the machine with the best sense organs that money can buy, and then teach it to understand and speak English. This process could follow the normal teaching of a child. Things would be pointed out and named, etc. Again I do not know what the right answer is, but I think both approaches should be tried."

Alan Turing, "Computing Machinery and Intelligence," 1950

Final Thoughts

“Today we serve technology. We need to reverse the machine-centered point of view and turn it into a person-centered point of view: Technology should serve us.”

From *Things That Make Us Smart: Defending Human Attributes in the Age of the Machine,* by Donald A. Norman

The question is not “how can mathematicians use the technology?” but rather “what can technology do for mathematicians?”

Final Thoughts

As long as we continue to reason, deliberate, and communicate with one another, mathematical language and concepts will be essential.

Mathematics is fundamental to human endeavors from science and technology to public policy and finance.

In the face of technological change, we need to preserve mathematical values and maintain a mathematical outlook, and pass them on to the next generation.

The new technologies hold great promise, but they must enhance our ability to do mathematics, not replace it.