Kimina Prover

Tackling Competition Level Mathematics through Reinforcement Learning

Mantas Bakšys and Jonas Bayer
Project Numina

Who we are

Project Numina

Non-Profit, AI4Math, Open-Source

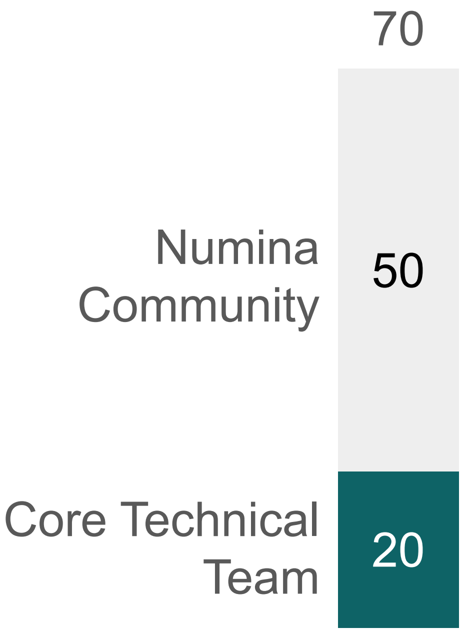
Open scientific collaboration initiated by Jia Li, Yann Fleureau, Guillaume Lample, Stan Polu & Hélène Evain

Open Data
High Quality Datasets

AI for Formal Reasoning
Developing Open Models

Human-AI Collaboration
Tools and Platforms

Contributors and Funding



Taskforces:

Dataset
Autoformalization
Model
Reinforcement Learning
Platform

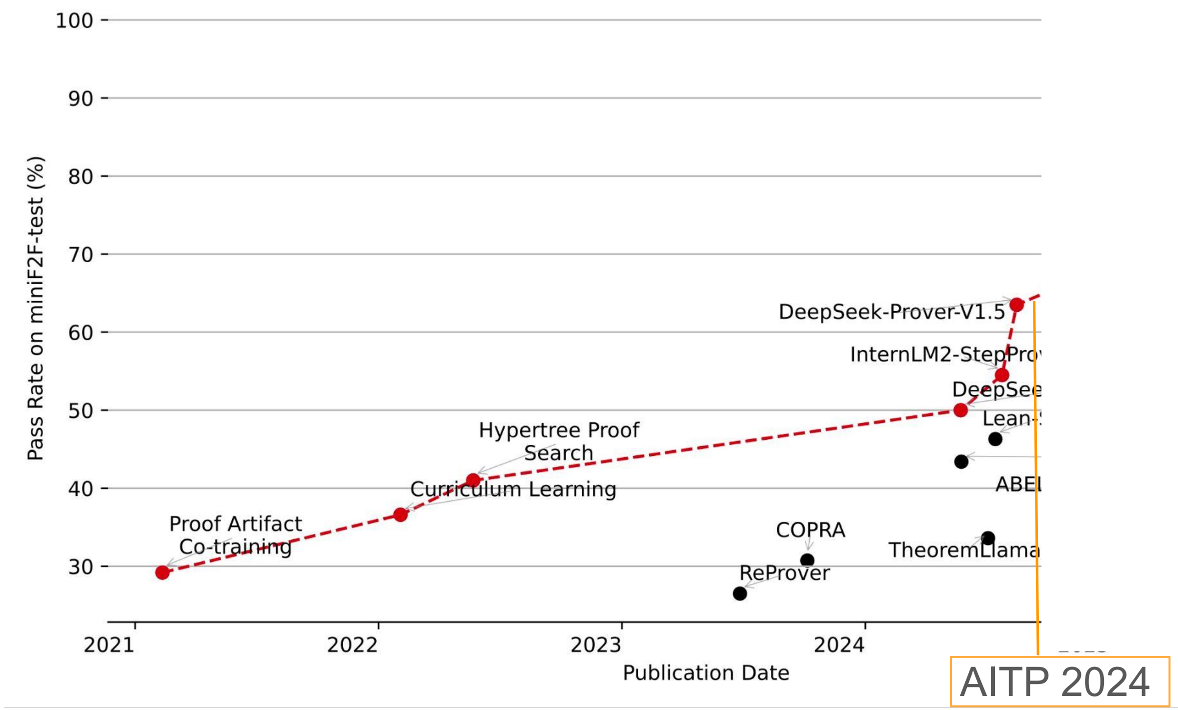
Agenda

Project Numina

- Context: Benchmarks and Previous Approaches
- KiminaProver
  - Chain of Thought Pattern
  - Infrastructure
  - Results
  - Extension features
- Outlook

miniF2F – from Challenge…

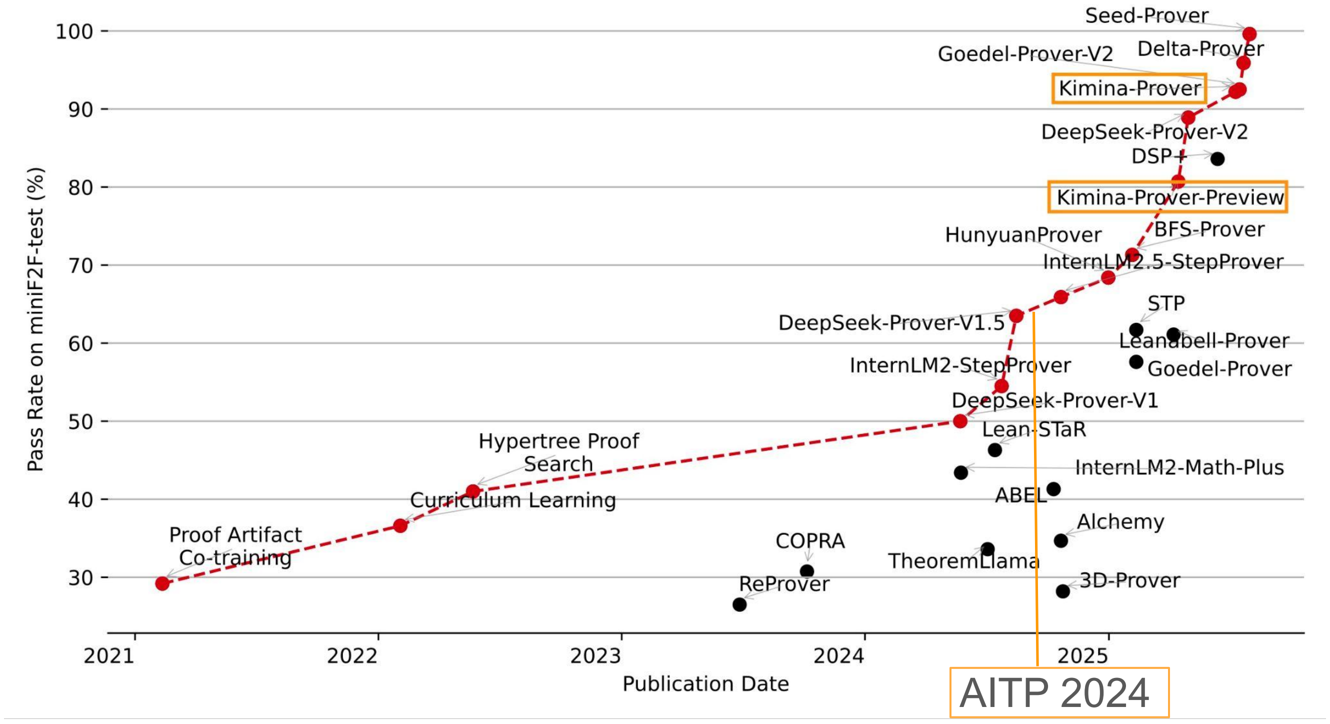
High-school level competition mathematics in Lean4



Source of diagram: SeedProver Paper

miniF2F – from Challenge... to Saturation

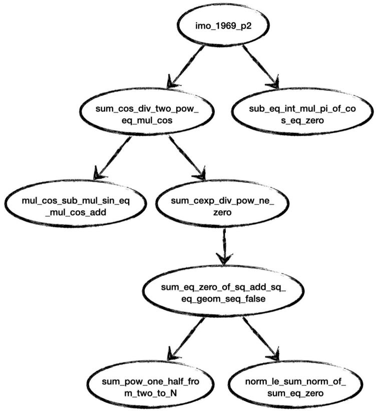
High-school level competition mathematics in Lean4



Source of diagram: SeedProver Paper

IMO as a Test for Formal Theorem Proving

- Algebra and Number Theory IMO problems are easy to state but hard to prove in Interactive Theorem Provers.
- Requires sophisticated multi-step reasoning.
- New problems every year, avoids contamination risks.



Natural Language Statement

1969/2.
Let $a_1, a_2, \cdots, a_n$ be real constants, x a real variable, and

$$f(x) = \cos(a_1 + x) + \frac{1}{2}\cos(a_2 + x) + \frac{1}{4}\cos(a_3 + x) + \cdots + \frac{1}{2^{n-1}}\cos(a_n + x).$$

Given that $f(x_1) = f(x_2) = 0$, prove that $x_2 - x_1 = m\pi$ for some integer m.

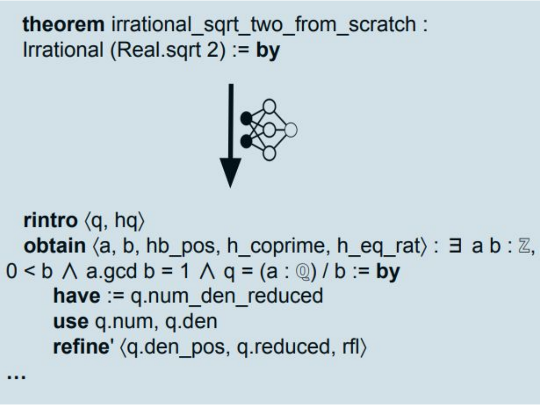
Lean 4 Statement

```
theorem imo_1969_p2 (m n : ℝ) (k : ℕ) (a : ℕ → ℝ) (y : ℝ → ℝ) (h₀ : 0 < k)
  (h₁ : ∀ x, y x = ∑ i in Finset.range k, Real.cos (a i + x) / 2 ^ i) (h₂ : y m = 0)
  (h₃ : y n = 0) : ∃ t : ℤ, m - n = t * Real.pi := by
```

Generated proof spans 520 lines and uses 7 sub-lemmas

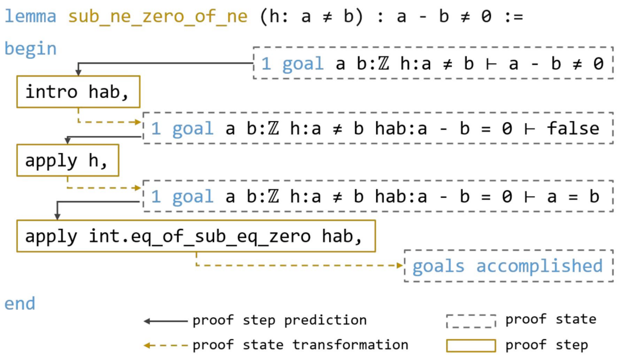
Previous Approaches & Limitations

Whole Proof Generation



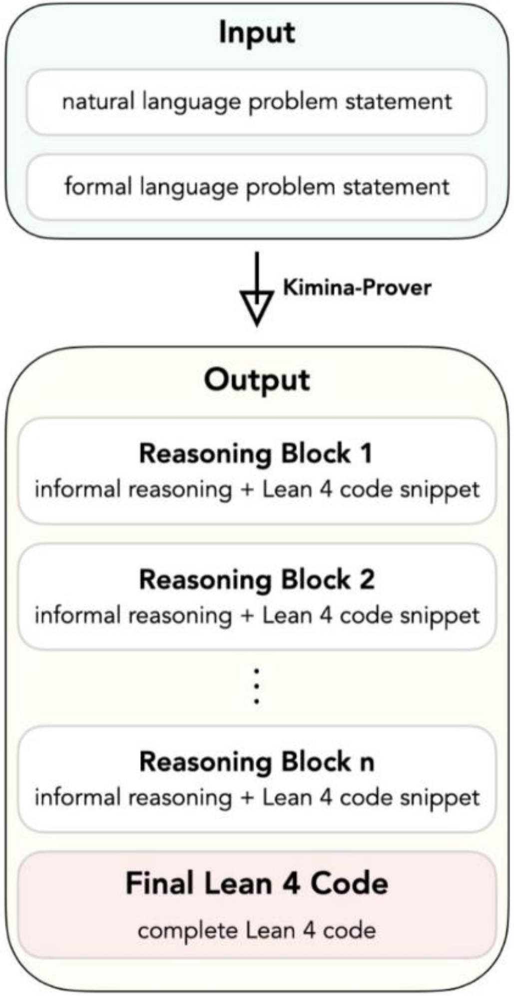
Standard LLMs struggle with deep, non-linear formal reasoning.

Step Prover



Complex search algorithm
Less inference efficient

→ Limited performance scaling with model size.

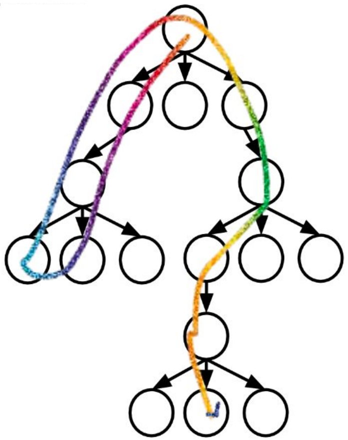


Idea: Chain of (Formal) Thought Reasoning

Bring o1-style reasoning to Formal Maths:

- Reasoning CoT that captures step-prover behaviour
- Whole-proof model inference efficiency

Flattened Search Tree



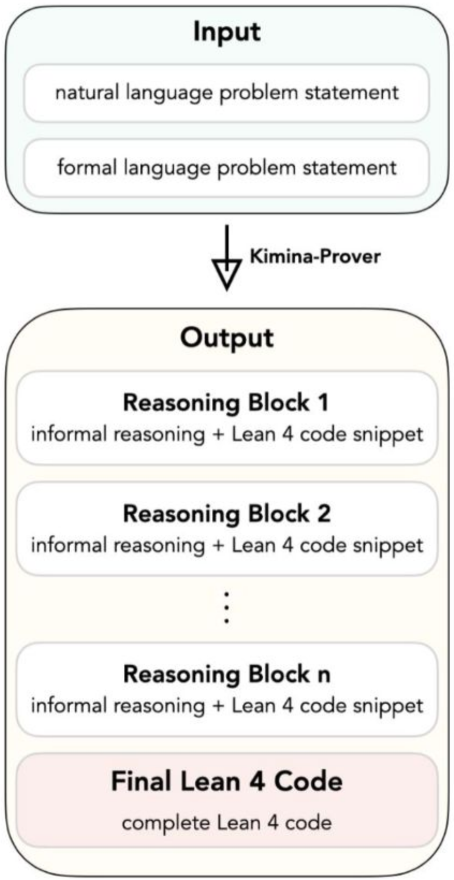
RL

Activation Data

Need: Samples solutions in our output format

- Special tags <think>, ```lean ``` code markers
  → intersperse informal and formal
- Transform 20K samples to our format by few-shot prompting Claude Sonnet 3.7
  → used for fine-tuning
- Further fine-tune with informal math reasoning data
  → includes more types of reasoning

→ Obtain model able to mix informal and formal



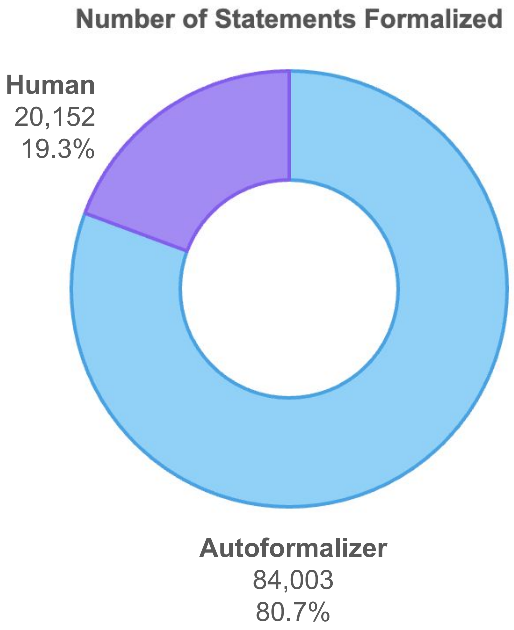
Problem Dataset

Need: Lots of problems to do RL with

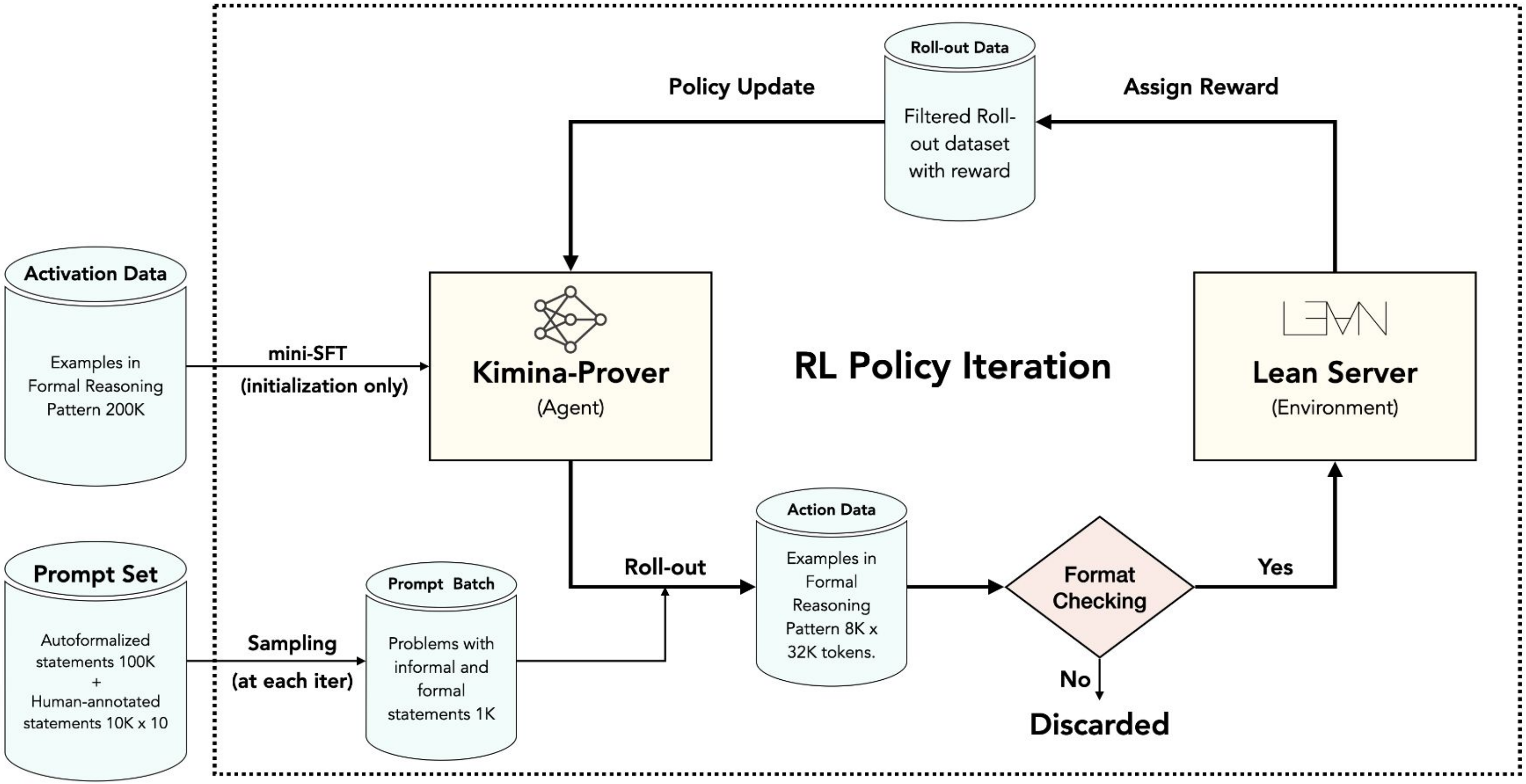
- Manual formalization by expert human annotators
- Statement-autoformalization for scaling
  Challenge: lack of direct reward
  → use LLM as a judge

→ Largest public Lean statement dataset

Open-sourced on HuggingFace

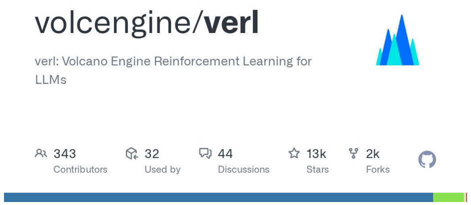


The Kimina Prover RL Pipeline



Infrastructure: verl

- RL-framework for LLMs
- Implements “Hybridflow” approach:
  RL as a dataflow, combining **control** and **computation** flows.
- Integration with FSDP, Megatron-LM, vLLM and SGLang.

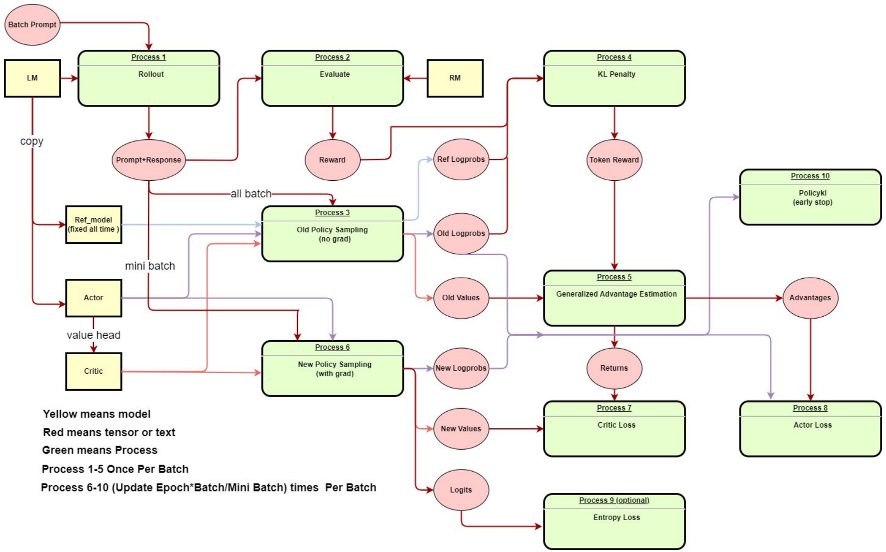


SGL

vLLM
Easy, fast, and cheap LLM serving for everyone

NVIDIA

MEGATRON-LM



Infrastructure: Lean Server

Challenge: Verify lots of Lean proofs quickly and efficiently, and reliably in a distributed setting

- Supports parallel Lean REPL processes.
- Reuses imports across multiple requests (LRU cache).

Mode	Total Verification Time (mm:ss)	Average Verification Time (s/it)
Cached	05:50	3.65
Non-Cached	08:14	5.14

Table 2: Performance comparison of cached vs. non-cached verification on a MacBook Pro M2 with 32GB RAM and 10 CPUs on the first 100 samples from the Goedel-LM/Lean-workbook-proofs dataset. Caching leads to significantly faster verification times.

# CPUs	Total Verification Time (mm:ss)	Average Iterations Rate (it/s)
8	20:11	0.83
16	09:57	1.67
32	05:54	2.82
60	03:51	4.33

Table 1: Performance scaling of proof verification with different CPU configurations (60-core Intel Xeon CPU @ 3.10GHz) on the first 1000 samples from the Goedel-LM/Lean-workbook-proofs dataset. Increasing the number of CPUs consistently translates into higher average iterations rates.

Kimina Prover Preview: Changing the ATP Landscape

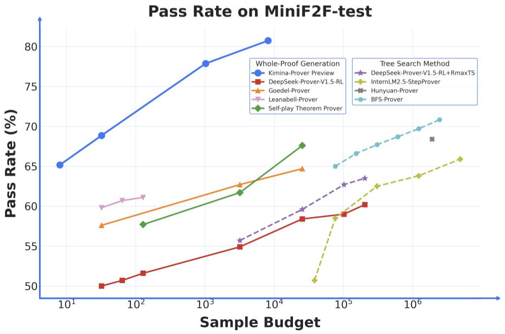
KIMINA-PROVER PREVIEW: TOWARDS LARGE FORMAL REASONING MODELS WITH REINFORCEMENT LEARNING

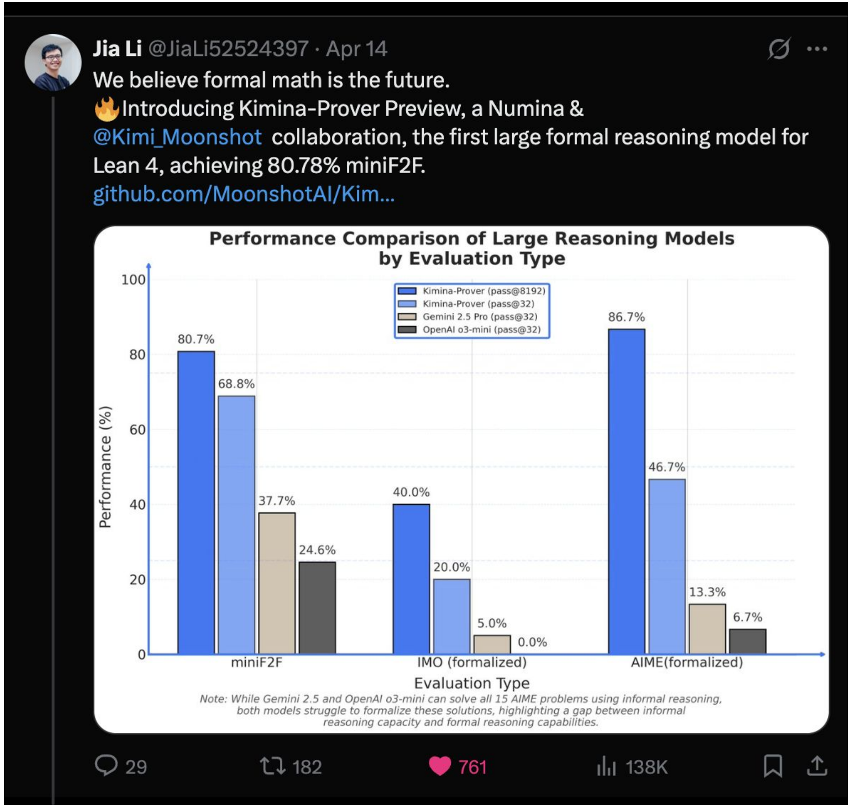
TECHNICAL REPORT OF KIMINA-PROVER PREVIEW

Numina & Kimi Team

ABSTRACT

We introduce Kimina-Prover Preview, a large language model that pioneers a novel reasoning-driven exploration paradigm for formal theorem proving, as showcased in this preview release. Trained with a large-scale reinforcement learning pipeline from Qwen2.5-72B, Kimina-Prover demonstrates strong performance in Lean 4 proof generation by employing a structured reasoning pattern we term *formal reasoning pattern*. This approach allows the model to emulate human problem-solving strategies in Lean, iteratively generating and refining proof steps. Kimina-Prover sets a new state-of-the-art on the miniF2F benchmark, reaching 80.7% with pass@8192. Beyond improved benchmark performance, our work yields several key insights: (1) Kimina-Prover exhibits high sample efficiency, delivering strong results even with minimal sampling (pass@1) and scaling effectively with computational budget, stemming from its unique reasoning pattern and RL training; (2) we demonstrate clear performance scaling with model size, a trend previously unobserved for neural theorem provers in formal mathematics; (3) the learned reasoning style, distinct from traditional search algorithms, shows potential to bridge the gap between formal verification and informal mathematical intuition. We open source distilled versions with 1.5B and 7B parameters of Kimina-Prover[1].





Emergent Human-like Proof Style

The model learns complex reasoning patterns beyond simple step generation:

- Exploring multiple paths, reflection, refining thinking.
- Generating structured, decomposed proofs (e.g., using have statements)
- Reasoning length correlates with proof complexity

```
theorem imo_1962_p2 (x : ℝ) (h₀ : 0 ≤ 3 - x) (h₁ : 0 ≤ x + 1)
    (h₂ : 1 / 2 < Real.sqrt (3 - x) - Real.sqrt (x + 1)) : -1 ≤ x ∧ x < 1 - Real.sqrt 31 / 8 := by
  have h3 : -1 ≤ x := by
    nlinarith
  have h4 : Real.sqrt (3 - x) - Real.sqrt (x + 1) > 1 / 2 := by linarith
  have h5 : Real.sqrt (x + 1) ≥ 0 := Real.sqrt_nonneg (x + 1)
  have h6 : (7 / 4 - 2 * x) > Real.sqrt (x + 1) := by
    nlinarith [Real.sq_sqrt (show (0 : ℝ) ≤ 3 - x by linarith), Real.sq_sqrt (show (0 : ℝ) ≤ x +
    1 by linarith),
      Real.sqrt_nonneg (3 - x), Real.sqrt_nonneg (x + 1)]
  have h7 : (7 / 4 - 2 * x) ^ 2 > (Real.sqrt (x + 1)) ^ 2 := by
    nlinarith [h6, Real.sqrt_nonneg (x + 1)]
  have h8 : (7 / 4 - 2 * x) ^ 2 > x + 1 := by
    have h10 : (Real.sqrt (x + 1)) ^ 2 = x + 1 := by
      rw [Real.sq_sqrt]
      linarith
    nlinarith [h7, h10]
  have h9 : 64 * x ^ 2 - 128 * x + 33 > 0 := by
    nlinarith [h8]
  have h10 : x < 1 - Real.sqrt 31 / 8 := by
    by_contra hx
    push_neg at hx
    have h12 : Real.sqrt 31 > 0 := by
      apply Real.sqrt_pos.mpr
      norm_num
    nlinarith [sq_nonneg (x - (1 - Real.sqrt 31 / 8)), sq_nonneg (Real.sqrt 31), Real.sq_sqrt
    (show (0 : ℝ) ≤ 31 by norm_num),
      h12]
  exact ⟨h3, h10⟩
```

Listing 3: Lean 4 proof of IMO-1962-P2 found by Kimina-Prover.

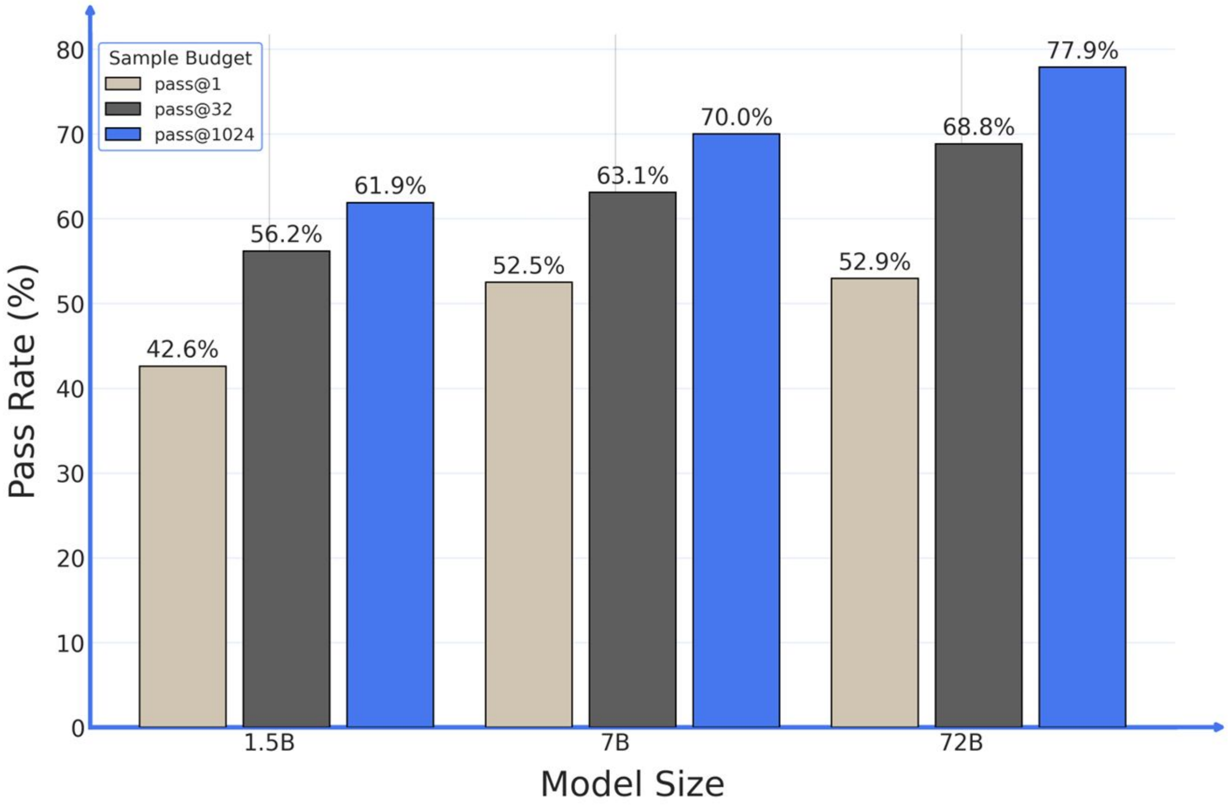
```
theorem imo_1962_p2 (x : R)
  (h0 : 0 <= 3 - x)
  (h1 : 0 <= x + 1)
  (h2 : 1/2 < sqrt(3 - x) - sqrt(x + 1)) :
  -1 <= x and x < 1 - sqrt(31)/8 := by {
    constructor
    linarith
    rw [← sub_pos]
    field_simp [Real.sqrt_lt] at h2 |-
    apply lt_of_le_of_lt
    rw [mul_comm]
    rw [sub_eq_add_neg]
    apply lt_of_le_of_lt
    rw [← lt_sub_iff_add_lt]
    ring_nf
    rw [← lt_sub_iff_add_lt]
    linarith [Real.sq_sqrt (by linarith : 0 <= 1 + x)]
    rw [Real.sqrt_lt (by norm_num)]
    rw [Real.sqrt_lt] <;> nlinarith
    norm_num at this
  }
```

Listing 4: Lean 4 proof of IMO-1962-P2 found by BFS-Prover.

Performance Scaling with Model Size

Observation:
Clear performance improvement as model size increases (1.5B -> 7B -> 72B parameters).

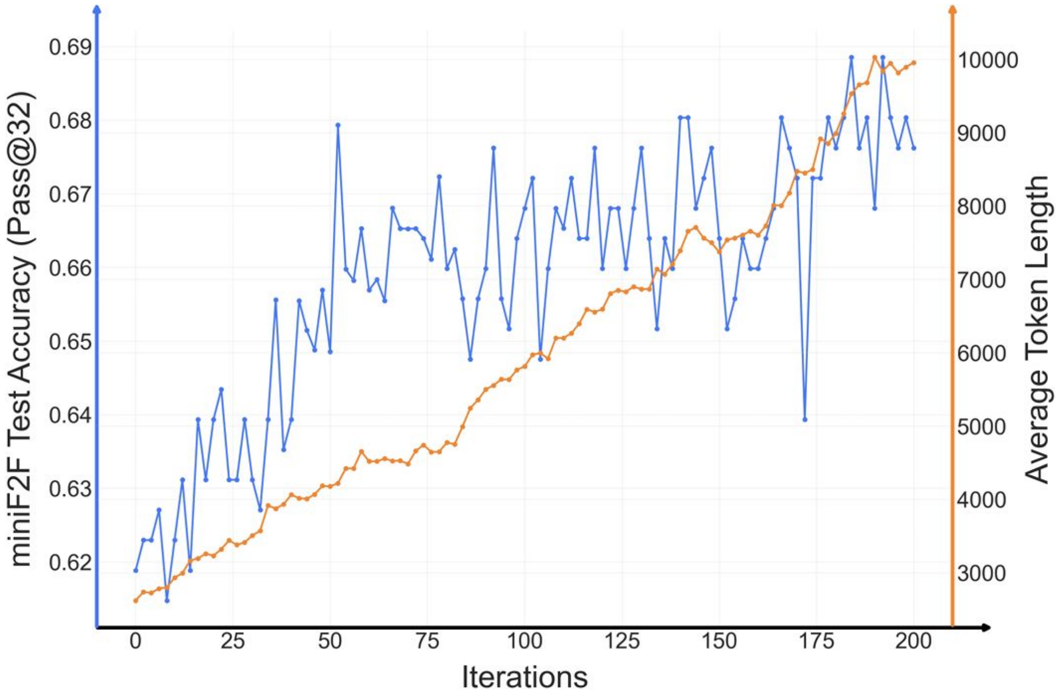
- 72B model significantly outperforms 7B, especially with larger sample budgets (+7.87% at pass@1024).
- This scaling trend ***was not clearly observed*** in previous neural theorem provers for formal math.



Test Time Scaling

Observation:

- Performance (pass@32) improves as the model learns to generate longer, more complex outputs
- Formal reasoning length scaling is more volatile than informal math but ultimately successful. Potential applications to other data-limited domains.



Multi-Turn Interactions

Idea: Extend the CoT approach to allow multiple LLM-Lean interactions

- If Lean rejects the proof a new prompt is generated which includes the error message(s).
- Continue with repairing the proof attempt using the Lean feedback.

Cold-Start Data (again):

- To get the pipeline running, we sample cold-start data in our error fixing format using Claude Sonnet 4.
- We pair up syntactically similar failing and successful proof attempts and generate a dataset to teach the model the error-fixing pattern.

Error Fixing Improvements

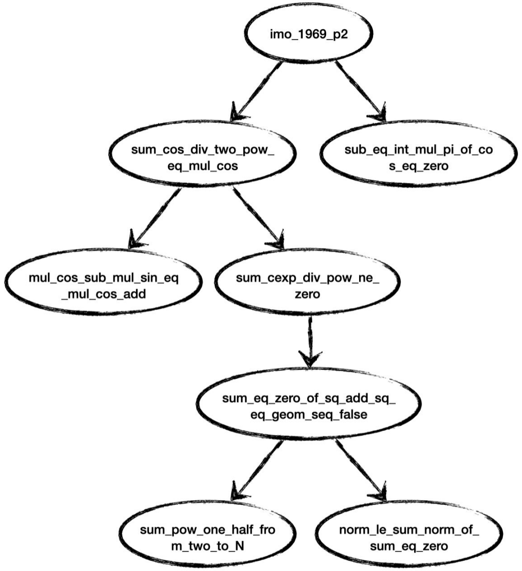
	16+16 attempt-and-fix	32×1 brute-force	32+32 attempt-and-fix
kimina-prover	35.6	28.8	44.1

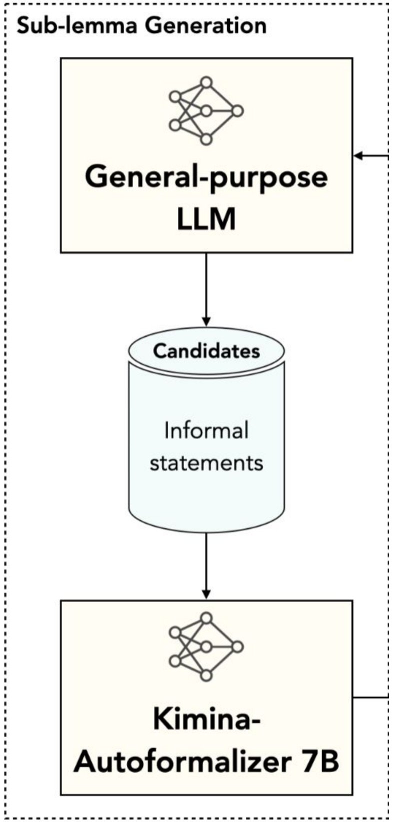
Table 2: *Performance comparison between error-fixing and brute-force strategies on a selected subset of 59 MiniF2F-Test problems with the lowest win rates. Under equal sample budgets, the error-fixing strategies (16+16) outperform the brute-force baseline (32×1), demonstrating improved sample efficiency.*

Lemma-Enabled Reasoning Pattern

Complex problems require breaking down the proofs into smaller steps:

- Two step-pipeline combining a general purpose LLM with Kimina-Autoformalizer 7B to generate sub-lemmas.
- Equip the model with the ability to identify and utilize useful lemmas provided in the input.



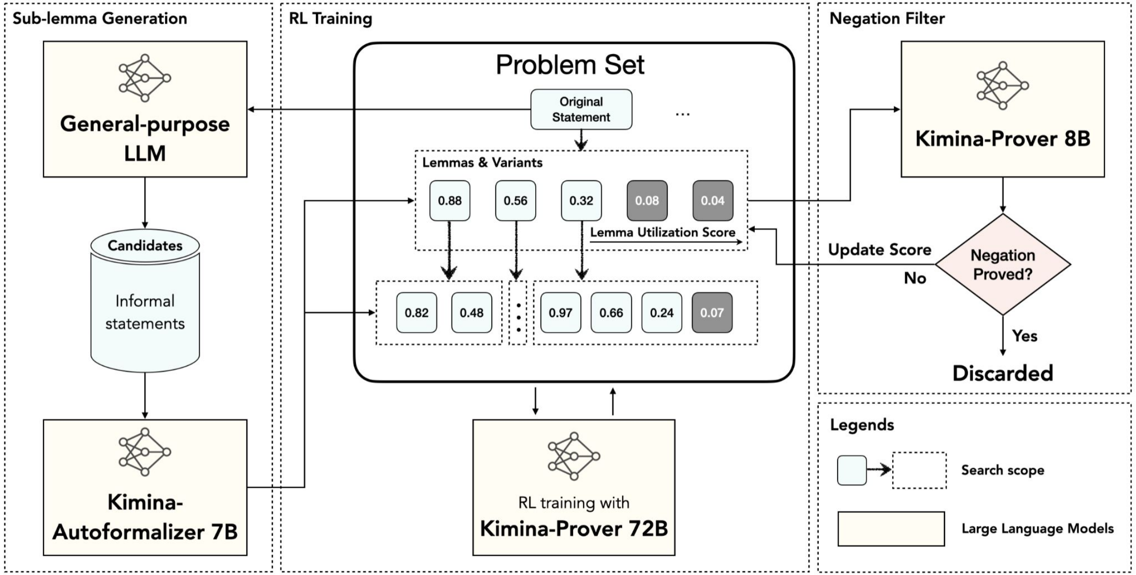


Test Time Reinforcement Learning Search

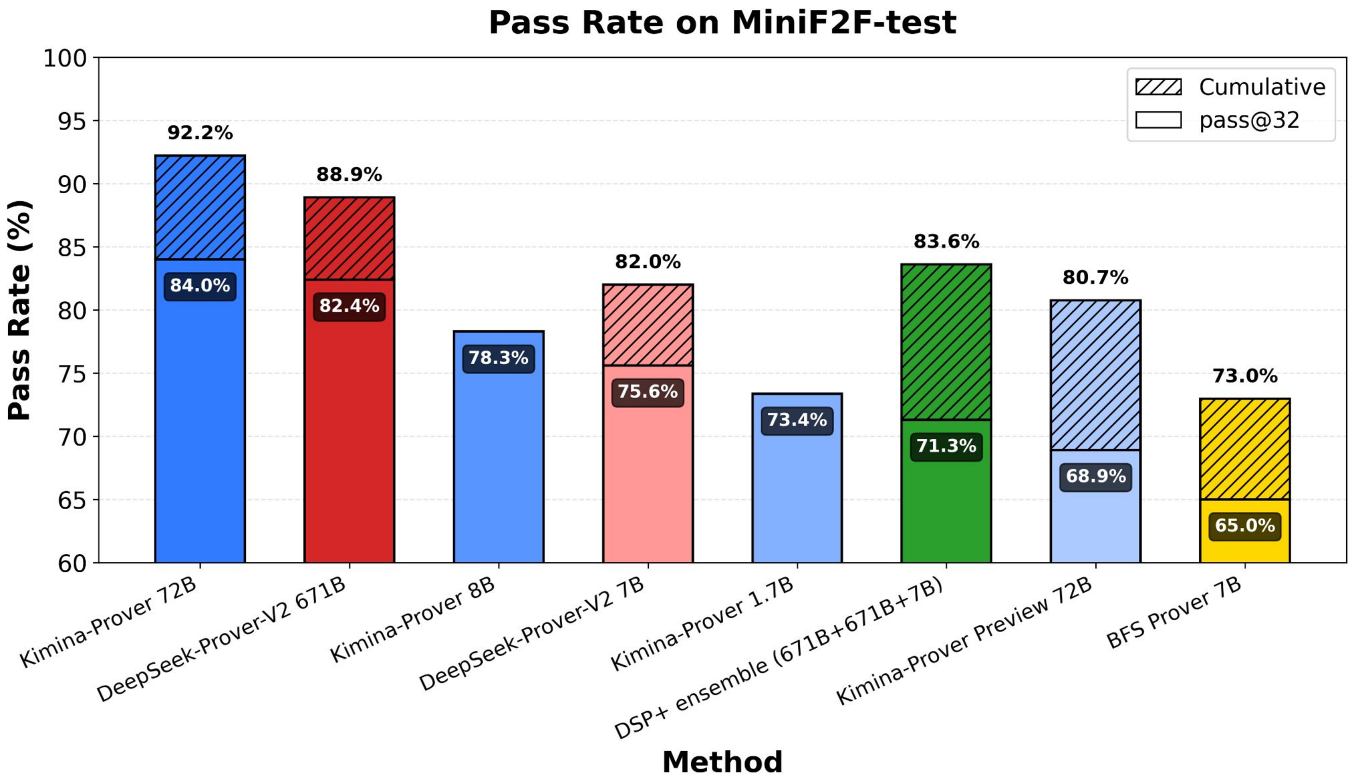
A trainable agentic proving framework that enables the model to recursively:

- discover,
- combine and,
- apply

lemmas to construct complex proofs, building on a novel **lemma-enabled pattern.**



Kimina Prover Release



IMO Releases

ByteDance Seed Prover Achieves Silver Medal Score in IMO 2025

Aristotle Achieves Gold Medal-Level Performance at the International Mathematical Olympiad, iOS App Beta Launch 07.28.2025

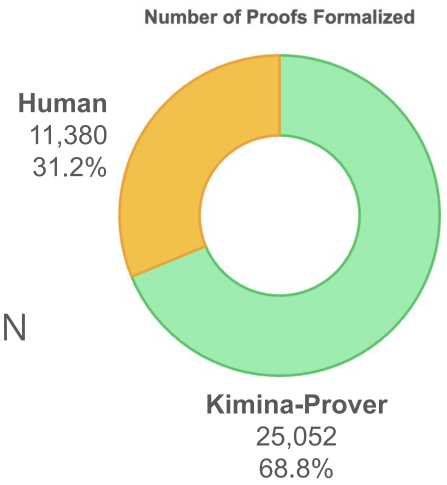
DeepMind and OpenAI claim gold in International Mathematical Olympiad

Two AI models have achieved gold medal standard for the first time in a prestigious competition for young mathematicians – and their developers claim these AIs could soon crack tough scientific problems

By Alex Wilkins

22 July 2025

Open-Source Releases



Dataset NuminaMath-LEAN

huggingface.co/datasets/AI-MO/NuminaMath-LEAN

Training Pipeline KiminaProver-RL

github.com/project-numina/kimina-prover-rl

Infrastructure: Lean Server & Client

github.com/project-numina/kimina-lean-server
pypi.org/project/leanclient

Kimina Prover Demo

demo.projectnumina.ai

Future Directions

- More than one approach successful for IMO-level mathematics:
  - ➔ Pure natural language & formal successful – what's to come?
- Key challenge: Staying up to date with Lean/Mathlib updates
  - ➔ How to provide reliable infrastructure for proof assistant users?
- New benchmarks and tasks needed?
  - ➔ PutnamBench, RLMEval
  - ➔ End-to-end development of mathematical theories

Thank You!

Project Numina