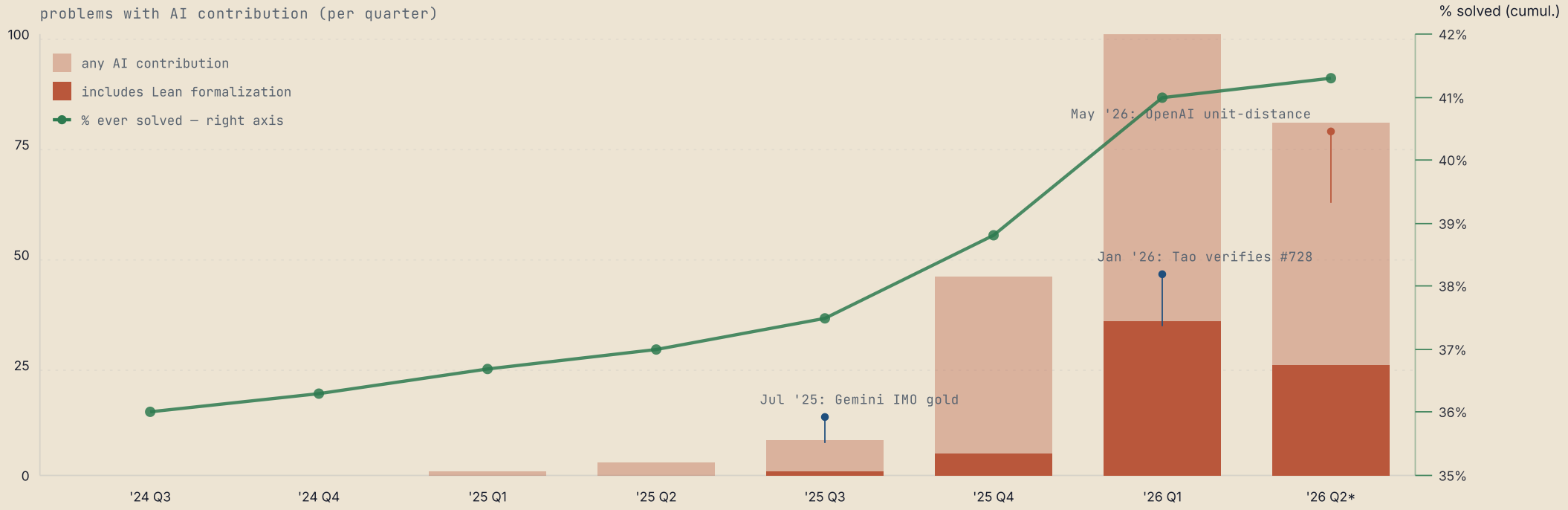
Smart Scaffolding
An AI Pipeline for Economic Proofs

Pedro Aldighieri · Northwestern University · pedro.aldighieri@u.northwestern.edu

AI contributions to Erdős problems

Per-quarter count · Tao-Bloom Erdős registry (N=1,179) · green line: share of all problems ever solved, right axis (estimated).



* Q2 partial: Apr + first half of May

Source: github.com/teorth/erdosproblems wiki, "AI contributions" page (accessed May 21, 2026). Registry total: 1,179 problems (Mar 2026).

A short timeline of AI progress in mathematics

2023	**GPT-3.5/4 cannot count the r's in "strawberry."**
JUL 2024	**First AI IMO Silver** — 4 of 6 competition problems solved at medal level. Google DeepMind · AlphaProof / AlphaGeometry 2.
JUL 2025	**AI achieves IMO Gold** — both Google DeepMind (Gemini) and OpenAI scored 35/42. AI reaches gold-medal level at a major math olympiad; only 67 of 630 human contestants did.
JAN 2026	**Terence Tao independently verifies** an AI proof of Erdős problem #728. First autonomous AI entry on the Tao–Bloom open-problems registry.
MAY 2026	**OpenAI resolves the unit-distance problem** — open since the 1950s.

Timothy Gowers (Fields Medal '98) · companion paper to OpenAI unit-distance proof

"If a human had written the paper and submitted it to the Annals of Mathematics and I had been asked for a quick opinion, I would have recommended acceptance without any hesitation. No previous AI-generated proof has come close to that."

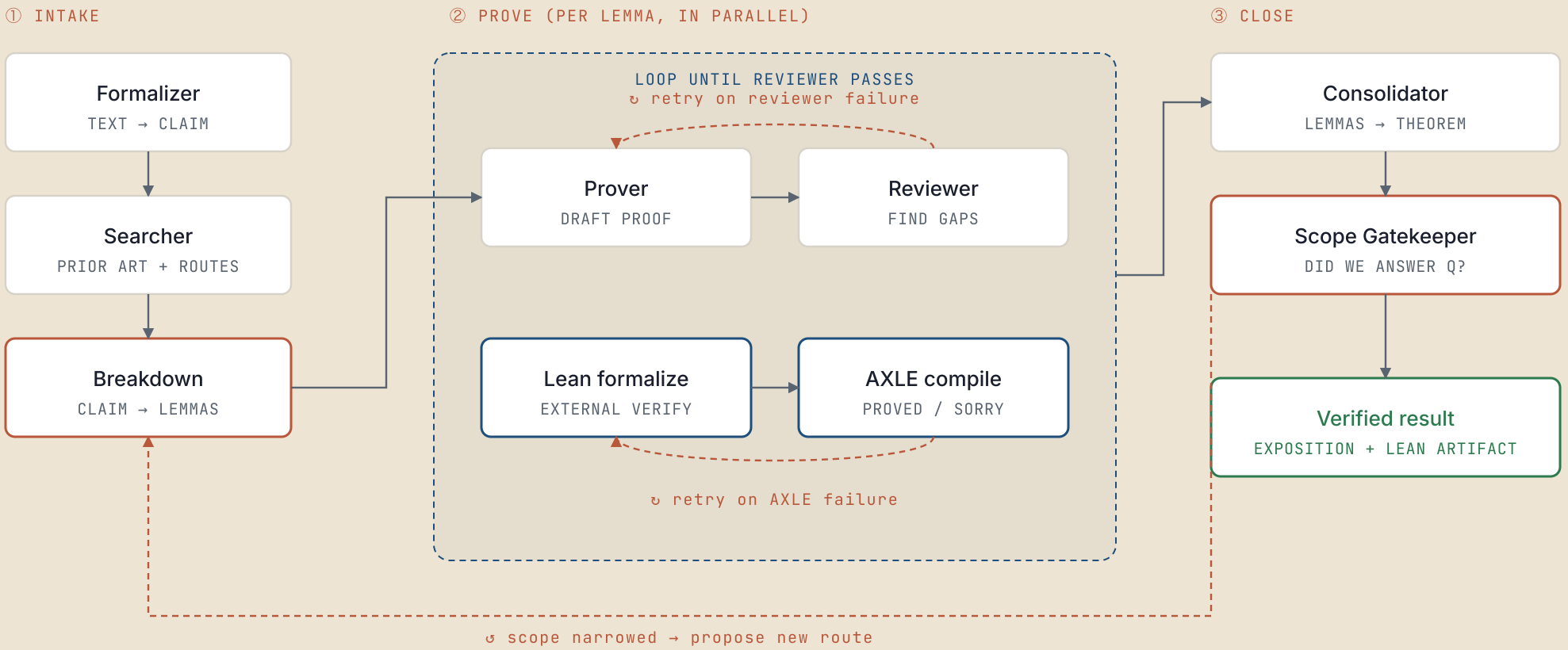
MathPipeProver

- This presentation introduces MathPipeProver, a scaffolding to automate proofs.
- Given recent progress in math, the idea was to apply LLMs to econ theory proofs.
- I've been developing it over the past months, working alongside Piotr Dworczak.
- It's still early days, but the results so far are encouraging.

What is a "scaffolding"?

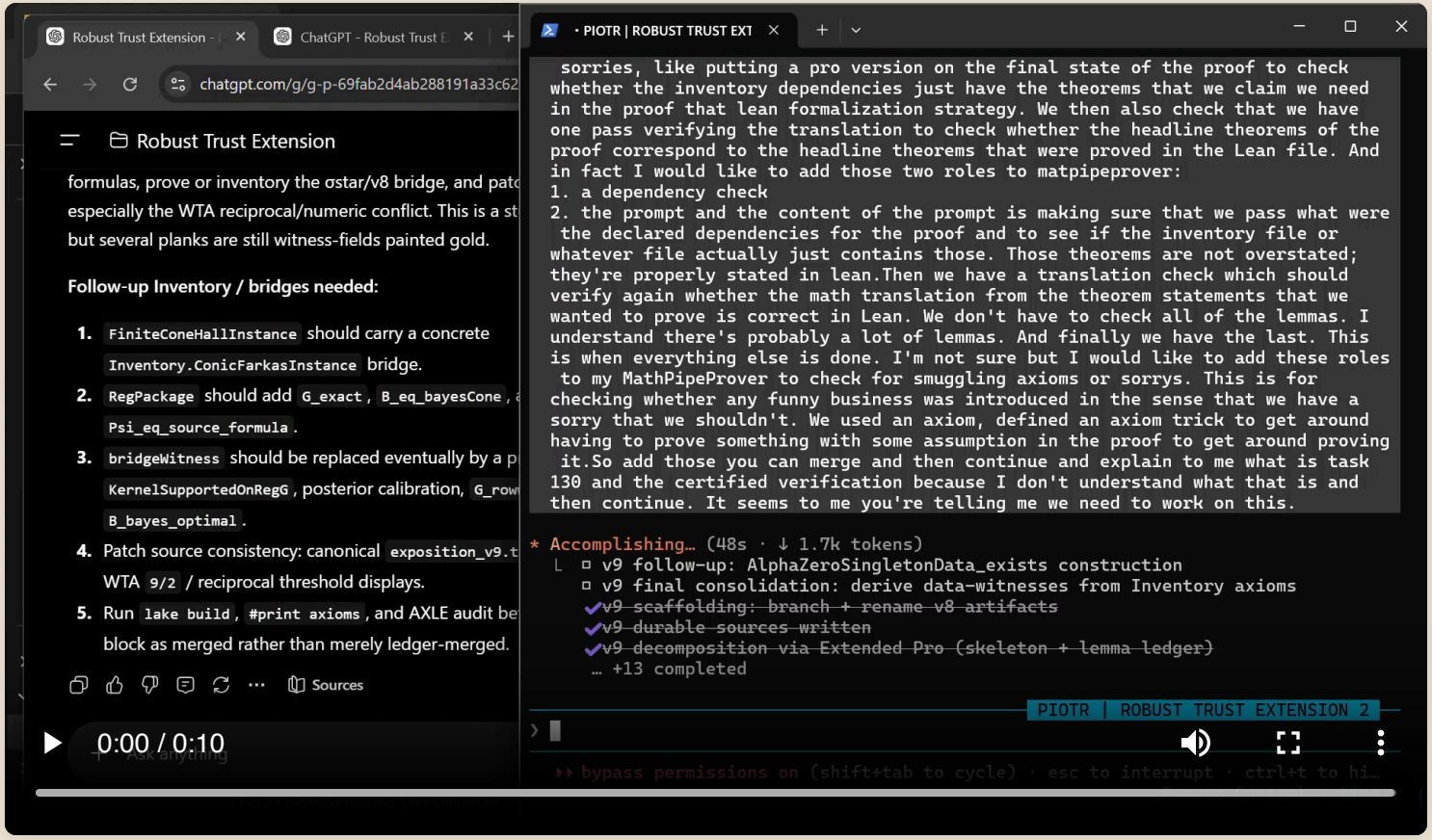
- A pipeline of prompts plus an orchestrator that moves proof state between them.
- Automates the in/out, instead copy-pasting between chats and judging each reply by hand.
- The roles divide labor along familiar lines: formalizer, literature searcher, prover, reviewer...
- The most important thing are not the prompts themselves, it's the conceptual division of labor.
- In particular, having a separate reviewer verify every step of the proof key.

The workflow



MathPipeProver in action

The orchestrator, Claude Code, driving a prover–reviewer loop, live in ChatGPT Extended Pro.



Case 1: Robust Trust

Extending an existence theorem from Dworczak–Smolin (2026) from finite to infinite menus.

165
GIT COMMITS IN THE PROOF REPO

577
GPT EXTENDED PRO SESSIONS

9
DISTINCT STRATEGY BRANCHES

- Four full pipeline passes. Most strategy branches did not close — most runs ended in honest "this route is dead" reports rather than fabricated proofs.
- Only produced strong conditional versions of the theorem.
- Lean formalized some routes that seems unintelligible, at great expense.

The interesting failure mode is the absence of a failure mode you'd expect. Across 577 sessions and dozens of dead routes, the pipeline did not produce a single plausible-but-wrong proof that survived the reviewer.

What we have tried so far

A portfolio view across the real git-backed attempts, not every exploratory folder.

9	600+	900+	2
REPOSITORY PROJECTS	GIT COMMITS	GPT PRO SESSIONS	PROMISING LEAN RUNS

- Mixed record: some closed results, some useful dead ends, some still in flight.
- One serious miss: a false statement slipped through a review of a submitted proof, and was caught only after focused re-checking.
- The Lean module is recent: two attempts look successful, but the current workflow is costly to run.

What is Lean, and how can a machine check a proof?

- A proof is a finite sequence of inference rules applied to axioms to reach a valid conclusion.
- Lean acts as a mathematical compiler: it treats a theorem as a specification and mechanically verifies that the proof satisfies it.
- Proofs are written using high-level commands (e.g., `induction`, `rw`), which Lean automatically compiles into formal logic.
- Lean's standard library (Mathlib) contains over a million formalized lemmas, allowing standard background math to be cited directly.
- This provides an objective, mechanical check on logical validity—exactly the ground-truth signal that an LLM cannot generate for itself.

What that looks like in practice

Every statement needs proof

```
-- 0+n=n is obvious — but Lean needs a proof.
theorem zero_add (n : ℕ) : 0 + n = n := by
  induction n with
  | zero      ⇒ rfl
  | succ n ih ⇒ rw [Nat.add_succ, ih]
```

Peano arthmetic axioms give us that `m + 0 = m`, but not that `0 + n = n`. `induction n with` gives two goals: `zero` (base) and `succ n ih` (step), where `n` is the predecessor and `ih : 0+n=n` is the induction hypothesis. `rfl` closes the base: 0+0=0 holds by definition of +. In the step: `Nat.add_succ` rewrites 0+(n+1) as (0+n)+1; then `ih` rewrites 0+n to n. Every step must be a named, proved fact.

How `MathPipeProver` uses Lean

- Post-consolidator Lean module: nine roles translate a verified English proof into checked Lean.
- Verification backend: AXLE by AxiomMath, a Lean compiler as a hosted API.
- AXLE repair-proofs auto-closes `sorry` stubs using proof tactics.
- AXLE disprove runs counterexample search on every "proved" lemma.

What you need to try this yourself

- **Claude Code** ($20/month) as the orchestrator — long-running session, self-loops, ability to remote-drive other tools and shell.
- **ChatGPT Pro** subscription, from $100/month. Pro's Extended mode is currently the strongest reasoning configuration for mathematics.
- The orchestrator drives your browser rather than calling the API directly. You pay your existing subscription, not per-token rates.
- **AXLE API key** — free tier — if you want the Lean module. Sign up at `axle.axiommath.ai`.

Repository: `github.com/p-aldighieri/MathPipeProver` · Contributions welcome.

Where does that leave us?

- Capabilities are moving fast; no plateau in sight.
- Formal proofs are self-contained, and correctness can be checked externally.
- That makes formal reasoning an excellent target for RLVR.
- Near-term possibility: systems become superhuman at proving formal statements.
- What is the scarce human input: valuable problems, search, writing?

Thank you

Feel free to email me with questions, suggestions, or proofs you would like to see attempted.

- pedro.aldighieri@u.northwestern.edu