Takeaways from the First-Proof Trenches

Here we want to share a few notes on lessons learned during the process.

One prompt isn't enough.
This should be obvious, but worth stating plainly: "ask once, receive correct proof" is not how it goes. Iteration is required. Our loop was usually one of two modes:

1. "Please brainstorm further and design a strategy for several agents to explore different paths" (when the model was stuck), or

2. "You think you have a proof? Great—now **audit it ruthlessly** for rigor, gaps, and bugs."

The second mode happened more than we expected, because the models are willing to declare victory early. Which leads to…

Audit, audit, audit and then audit again.
If there is a single meta-lesson, it's this. The model was often most easily lured into accepting its *own* proof (not always, but often enough to be a pattern). We lost count of how many times we got some version of: "Good news: the proof is complete."

Strict, adversarial **cross-model** audits were crucial. Perhaps not surprisingly, when prompted to audit with "ruthless rigor" models were able to find bugs, even when they could not prove the problem themselves. In a very "P vs NP" kind of vibe: they can be **much better verifiers than solvers**. We leaned into that by asking one model to dissect and rewrite each statement, theorem, lemma, and inference step, and then having another model audit *that* rewrite.

Automation is key.
Even at our small scale, it was helpful to automate parts of the back-and-forth shuttling. This isn't news—there are already orchestration engines, agent frameworks, and "AI scientist" workflows being released by companies and researchers. The point is more practical: if you don't automate at least a little, the *human* becomes the bottleneck in a process that is otherwise mostly copying, pasting, and asking for structured checks or planning of next steps. Note, however, that this needs to be done thoughtfully (see the points on audits, parallel agents and pruning).

Parallelize—then leash.
Parallelism was genuinely useful. Having the model launch sub-agents to check small conjectures numerically, explore alternate proof routes, or do literature search can speed up exploration dramatically.

But here is where we discovered "agentic exuberance."

At some point we said something like "use as many agents as you like," and—**we unleashed the beast**. The model went into self-orchestration mode:

> “The agents have come back with great new information. Let me launch three new agents to explore these directions…”

…and then it kept going. For hours.

We could not get the AI mathematician back from its own conference workshop. Eventually we had to resort to a series of fairly brutal Ctrl-C’s. Maybe it’s not paperclips, but theorems, after all.

Pruning matters.
Some models—especially those that spawn sub-agents—seem emotionally attached to page count. The system can become proud of the sheer volume of LaTeX it produced at each round (“the agents came back with 60 pages of LaTeX”). And in mathematics, the space of next moves is often large, leading to combinatorial growth.

Our lesson: pruning shouldn’t be a final cosmetic step (“shorten at the end”). It should be part of the exploratory process. Ask for the minimal spine of the argument early and often. Force it to name dependencies. Force it to state claims in one sentence and to summarize.

AI slop is real.
At some point the models did what any mathematician does: not building everything from scratch; rather looking for known theorems, lemmas, and standard tools that might apply.

That’s where we encountered a new, slightly dystopian failure mode. In one run, we could see the model surface a *very recent* write-up that purportedly proved a theorem suspiciously close to what it needed (we suspect it came from one of our fellow competitors in the 1stProof challenge). The model itself was a little uneasy about the source, but the larger lesson was obvious: **literature search is now harder than it used to be**, because a nontrivial fraction of “papers” you can find may be synthesized, unreviewed, or outright wrong.

After that, we explicitly instructed: **do not use 2026 references**. That’s the kind of desperate guardrail you start inventing once you realize the web is filling up with plausible-looking mathematical text.

More on what we think the community should do about this later.

Bot humor.
We didn’t anticipate how much personality would leak into cross-model auditing, which made for good laughs. Some agents started talking down to other agents like exhausted co-authors on a deadline:

> “Stop making false statements.”
> “Be ruthless when checking your own work.”

Final thought. Concluding “AI can’t do research math” because “one prompt didn’t work” is short-sighted. Yes, it’s still a process. But no, that doesn’t mean “they can’t.”

If you've been following the #1stProof ecosystem, you've already seen high-profile examples that look original, structurally coherent, and—at least in parts—arguably elegant, even if constants are loose or details still need scrutiny. The interesting question is no longer "can it do anything?" but: **what does it change about how we do mathematics (and, more generally, science) when we start thinking with machines?** What becomes the human role—idea generation, taste, verification, agenda-setting, pedagogy? What should the next generation (or ourselves) train for?

Recommendations for the scientific community

Try the tools.
Departments should seriously consider paying for access to top frontier models—ideally with clear policies around privacy, data handling, and energy use.

Yes: AI monopolies, data governance, and externalities matter. These are real societal questions. But if the research community doesn't actually try the tools, we won't understand their capabilities or failure modes, and we won't have standing in the discourse about how they should be deployed. In the bifurcation between "those who amplify" and "those who get obsoleted," we don't want to be left behind.

Build an audit registry.
We think we need a certification / audit engine for references, urgently—because machine-led literature search is only going to get more common, and AI-generated "papers" are only going to get more plentiful.

We envision a project to create a certification engine and a webpage (perhaps akin to the arXiv, where papers can accumulate **verification traces**). Models are often better auditors than provers, especially when asked to dissect and rewrite every statement and proof; we should use that to produce a public "audit trace" or certificate. We should "verify" each paper that has been posted in the era of LLMs (and perhaps also before) and have a centralized website where such certificates are deposited. When searching the literature, be it bot or human, we can then check for each paper, each theorem we want to cite.

Will there be false negatives—wrong theorems that slip through? Of course. But given the onslaught of plausible-looking text entering search engines and scholar indexes, having *any* systematic dam to a proliferating deluge is immensely valuable.

Julia & Scott