

AI Looping in Mathematics

Alex Townsend*

October 1, 2026

AI agents can now loop through the math literature. They can extract precise conjectures from papers, books, and conference proceedings, attempt them one by one, and return with a long list of problems they believe they have solved. Not every conjecture will fall this way. Some will need more compute or an ingenious human idea, but enough may fall that we should think about what happens next.

It’s a pretty simple workflow. An AI agent finds a conjecture in a paper, uses the surrounding text to reconstruct the relevant definitions and assumptions, and checks whether the problem is still open. It then attempts a proof or searches for a counterexample. If it finds something promising, another agent checks the argument, and a formal system such as Lean may verify a formalized proof. The process is then repeated for another paper, with the eventual aim of covering all publicly available mathematical research. Later passes reuse earlier attempts rather than starting from scratch (Figure 1).

And the scale of this loop is not as absurd as it first sounds. Let’s say we can assemble a corpus of about three million mathematical books and papers. Consider having 10,000 agents work in parallel and giving each publication ten agent-hours. (Any conjectures left unresolved in that first pass can receive more compute on a second loop.) That is 30 million agent-hours in total, or about 18 weeks of wall-clock time at full utilization. My back-of-the-envelope estimate, which makes many assumptions about falling future inference costs, puts such a first pass at roughly \$50 million in January 2027.¹ Later

loops could be cheaper because much of the extraction and context gathering need not be repeated, and a significant fraction of the problems might already be solved.

This prospect suggests two responses. First, we should think more carefully about how we recognize mathematical progress—that’s a whole other topic [2]. Second, I think we, the mathematical community, should help set up the loop because we know the context for a conjecture, the implicit motivations, and the consequences. We also need to turn the resulting proofs into human understanding, which motivates the “NLA, explained” project described below [6].

I don’t expect the transition to a new *modus operandi* to be painless. But I’d rather we help shape our subject’s future than simply react to it. With mathathons starting to pop up [3], I believe looping is going to happen with or without us. I think it will be done better with us.

I imagine each mathematical field crowdsourcing the effort and asking the public for help with the compute, much as astronomy invited public participation nearly twenty years ago. This could be a wonderful period for mathematics in which the conjecture-proving part of mathematics becomes accessible to hobbyists.

I think it makes most sense to run these looping projects in each specialized field independently, with experts involved. I encourage you to start a GitHub repository in your area of expertise and welcome others, including the general public, to have a go.

*The author is an Associate Professor of Mathematics at Cornell University. His email address is townsend@cornell.edu.

¹Three million publications times ten agent-hours each at

an assumed \$1.50 per agent-hour is \$45 million.

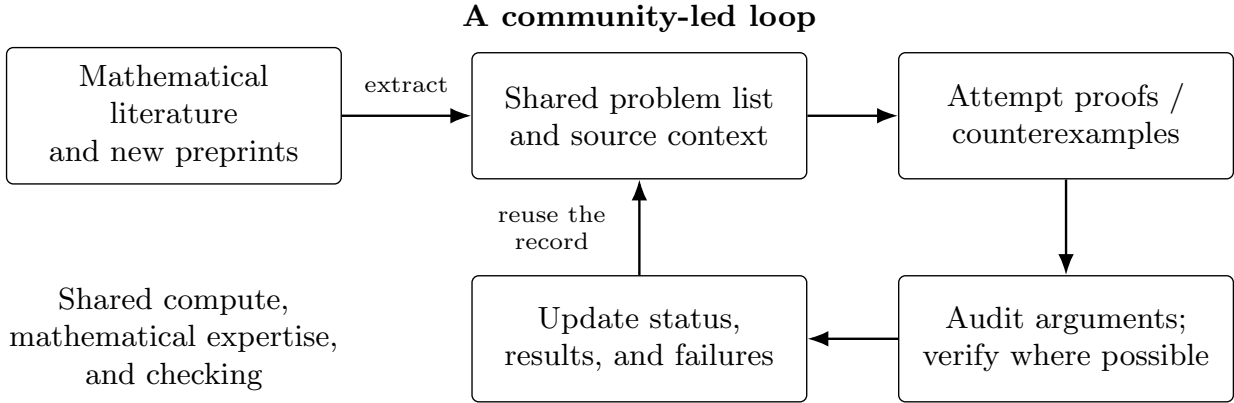


Figure 1: A possible mathematical loop. Later passes reuse the record of earlier attempts.

1 A small experiment

On the 11th of September 2026, I ran a small-scale version of this looping experiment. I released a GitHub repository containing about 200 problems in numerical linear algebra and related matrix mathematics, drawn from published sources [5]. This happens to be one of my areas of expertise, and I was interested to know what fraction of problems AI agents could now solve. As of September 28, 2026, the repository listed 106 problems as solved and 70 others as partially resolved. Of the 106, 64 had Lean-verified solutions; the remaining 42 had published results or independently audited arguments, including AI-agent audits. Section 3 explains the further checks for Lean verification (see also Figure 2). My goal is to eventually have all the solutions verified by Lean. In addition, I have also contacted 96 people by email for their expert opinion, and to confirm correct references and historical information. The pace of progress in the repository was unlike anything I’ve ever been involved with. Most contributors were researchers in numerical linear algebra or related fields, but we also wanted to get undergraduate and graduate students involved. So, together with colleagues Matthew Colbrook, George Stepaniants and Françoise Tisseur, I launched “NLA, explained,” a competition to turn some of these AI-generated arguments into explanations that mathematicians can understand [6].

There is also a much larger example of looping. In the Find, Attempt, and Recommend project, or FAR, Zheng and coauthors searched 5,245 combinatorics papers, extracted 6,453 candidate conjectures or open problems, and filtered them to 4,717 apparently well-posed and still-open questions [7]. Their pipeline surfaced 598 potential resolutions using GPT-5.5.

2 Mathematics without long-standing conjectures

The possibility of looping raises a concern. Traditionally, conjectures have given a community a shared research direction. Some people build techniques and collaborations around trying to solve conjectures, and proof attempts can go well beyond the original question. If many such questions were settled in a short period, the loss of that direction could be disorienting, even if the mathematical answers themselves are welcome.

However, mathematics can continue to flourish without so many long-standing conjectures. Some conjectures will remain too hard for AI agents, and some mathematicians may choose to work on these very challenging remaining problems. Progress will be slow but highly valued. There will also be plenty of work translating AI proofs into the realm of human understanding: identifying the central ideas, explain-

ing them to each other, advertising the techniques to other mathematicians and scientists, and formulating new problems that we find interesting or that have real-world applications. This is the motivation behind NLA, explained [6]. We also have teaching and mentoring as significant parts of our job to foster the minds of the young. That’s plenty to do, and we can be proud of doing it.

3 What I have learned about looping

To set up a productive GitHub repository for looping through the literature, a few details need a bit of sorting out.

Extraction. Extracting a precise conjecture from the literature is not always a trivial preliminary step. Sometimes the definitions were made precise in other work, the question is slightly misstated, or the original source is difficult to find. A problem may also have been solved since the source appeared. This is a big reason why looping needs to happen in each subfield, with experts participating. Each entry should preserve the source statement and its assumptions, along with the evidence that the question is still open.

Encouraging broad participation. Once you have a collection of open problems, the next step is to let the community know about it. I did that for my repo with a digest post that goes out to members of my community. One could also post on social media. It helps to explain not just how to submit an answer, but how someone with less background could make a useful contribution.

Auditing every submission. Once contributions start coming in, the owners of the repository need to carefully audit the submissions. In my repository, agents check entries in the solved and partially solved lists after updates, and other agents look for issues as new work is added. These checking bots have already found mistakes. When they are unsure about a conjecture’s history, I have also emailed people who know the literature. These exchanges help double-check the statement of a conjecture and its history. Of course, a bot audit is not a guarantee,

so I believe an argument should eventually be Lean-verified. These checks and verifications use compute too and could also be crowdsourced to a small group of trusted individuals.

Lean verification. For Lean verification, I have been using a back-and-forth procedure (Figure 2). One agent translates the mathematics into Lean. A second, independent agent translates the Lean statement back into ordinary mathematics. A third compares that reconstruction with the original statement in the repository and looks for altered assumptions, missing cases, or a changed conclusion. I do this because Lean might verify a statement that is subtly different from the original conjecture. The back-translation covers the theorem statement and its definitions and assumptions, rather than every step of the proof. To reduce the risk of simply reproducing the original prose, the second agent should work in a fresh context containing only the formal statement and its supporting definitions, with the original prose and explanatory comments withheld. It should identify the Lean declarations supporting its reconstruction. This reduces the risk but does not guarantee fidelity.

I separately ask agents to compare the actual Lean proof with the informal argument. A different valid proof can settle the same conjecture, but any change of strategy should be recorded so that the informal explanation does not misdescribe the formal proof. Back-translation is a readability aid: a reader familiar with Lean can instead compare the formal statement and proof directly with the original mathematics. Another agent independently reruns the Lean verification, checking for unproved placeholders or unexpected axioms as well as a successful build. Only after the statement audit and formal verification pass do I label the result Lean-verified. Ideally, an expert also checks the Lean statement. One could also add a “human-reviewed and Lean-verified” status after the argument appears in a peer-reviewed publication. My repository welcomes people to continue to work on the conjectures and write up the proofs in peer-reviewed manuscripts.

Difficult-to-read arguments. A challenge is that AI agents sometimes write arguments that are difficult to read or use ideas from literature outside

A first-pass audit of the formal statement

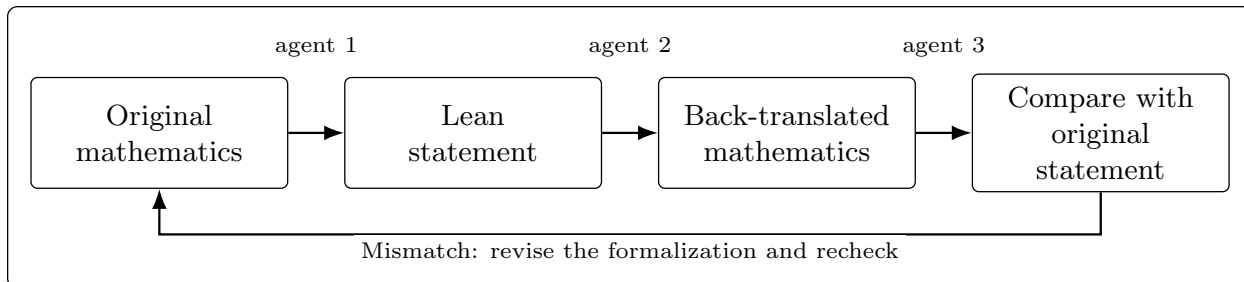


Figure 2: Initial procedure for auditing the translation of a conjecture into Lean. The process is not flawless; ideally a human expert checks the Lean statement.

your specialty. There is a mathematical opportunity here too: understanding why an argument works can reveal that it applies somewhere else. This is part of what we could encourage excellent undergraduates and graduate students to get involved in.

Giving credit to contributors. One issue I haven’t quite sorted out is how to give appropriate credit to individuals who spend a lot of time trying to solve a conjecture. GitHub provides a public record of who contributed what and when. But how this record translates into recognition in hiring and promotion remains less clear. So far, I have mainly been saying “thank you,” and hoping that others might appreciate their work too.

namics [9], which is closer to sharing the token burden.

Mathematics has its own precedent too. The Polymath project developed a new proof of the density Hales–Jewett theorem through open online collaboration [10, 11]. I see AI-assisted looping as extending our existing ambitions to collaborate widely and be a welcoming academic field. Thomas Hales’ Formal Abstracts project offers a complementary precedent: it aims to express the main results and definitions of mathematical publications in a computer-readable form grounded in Lean, without requiring formal proofs [12]. Such a corpus would remove part of the extraction step in Figure 1.

4 Other crowdsourcing efforts

Nearly twenty years ago, Galaxy Zoo invited members of the public to classify images of galaxies. Its initial research paper reported more than forty million classifications from about one hundred thousand participants and compared the results with those of professional astronomers [1]. The contributions were checked and became a scientific resource.

Astronomy is not the only place to look. In structural biology, players of the online game Foldit produced models that helped researchers determine the crystal structure of a retroviral protein in 2011 [8]. Folding@home offers a different model: volunteers donate computing time to simulations of protein dy-

5 Conclusion

If AI systems are about to loop through mathematics, I would rather the mathematical community help build the loop than simply watch it happen. If most currently catalogued problems are settled, I expect more of our effort to go into deciding what to ask next. In pure mathematics, that could mean developing concepts and connections that organize the answers and suggest new conjectures. In applied mathematics, it could mean choosing useful models and assumptions, and deciding which questions capture the phenomena we want to understand. Such open-ended work would make formulating the right question an even more important part of mathematical

research. I think mathematics can flourish without a stock of long-standing conjectures; our task is to help make that future one in which more people can take part.

References

- [1] C. J. Lintott et al. Galaxy Zoo: Morphologies derived from visual inspection of galaxies from the Sloan Digital Sky Survey. *Monthly Notices of the Royal Astronomical Society*, 389(3):1179–1189, 2008. <https://arxiv.org/abs/0804.4483>.
- [2] W. P. Thurston. On proof and progress in mathematics. *Bulletin of the American Mathematical Society*, 30(2):161–177, 1994. <https://arxiv.org/abs/math/9404236>.
- [3] Mathathon organizers. Caltech Mathathon. Event website, 2026. Accessed September 21, 2026. <https://mathathonchallenge.com/>.
- [4] A. Townsend. Open problems in numerical linear algebra. GitHub repository, 2026. Accessed September 28, 2026; version 80c0e3e. See also the resolution archive and contribution rules. <https://github.com/ajt60gaibb/OpenProblemsInNLA/tree/80c0e3e638b2f26dcb3a00353651fc3d2215dd65>.
- [5] *NLA, explained*. Proof digests and explanation competition. 2026. Accessed September 20, 2026. <https://nla-explained.com/>.
- [6] Z. Zheng, S. Zhang, J. Avigad, P. Tetali, and S. Welleck. The problem is the problem: Towards scalable mathematical discovery. arXiv:2608.16977, version 1, 2026. <https://arxiv.org/abs/2608.16977v1>.
- [7] F. Khatib, F. DiMaio, Foldit Contenders Group, Foldit Void Crushers Group, et al. Crystal structure of a monomeric retroviral protease solved by protein folding game players. *Nature Structural & Molecular Biology*, 18(10):1175–1177, 2011. <https://doi.org/10.1038/nsmb.2119>.
- [8] Folding@home. About. Project website. Accessed September 20, 2026. <https://foldingathome.org/about>.
- [9] T. Gowers. Is massively collaborative mathematics possible? Gowers’s Weblog, January 27, 2009. <https://gowers.wordpress.com/2009/01/27/is-massively-collaborative-mathematics-possible/>.
- [10] D. H. J. Polymath. A new proof of the density Hales–Jewett theorem. *Annals of Mathematics*, 175(3):1283–1327, 2012. <https://doi.org/10.4007/annals.2012.175.3.6>.
- [11] T. Hales et al. Formal Abstracts. Project website. Accessed September 28, 2026. <https://formalabstracts.github.io/>.