Quanta magazine

How Close Are Computers to Automating Mathematical Reasoning?

By Stephen Ornes

August 27, 2020

AI tools are shaping next-generation theorem provers, and with them the relationship between math and machine.

Maria Nguyen for Quanta Magazine

In the 1970s, the late mathematician Paul Cohen, the only person to ever win a Fields Medal for work in mathematical logic, reportedly made a sweeping prediction that continues to excite and irritate mathematicians — that “at some unspecified future time, mathematicians would be replaced by computers.” Cohen, legendary for his daring methods in set theory, predicted that all of mathematics could be automated, including the writing of proofs.

A proof is a step-by-step logical argument that verifies the truth of a conjecture, or a mathematical proposition. (Once it’s proved, a conjecture becomes a theorem.) It both establishes the validity of a statement and explains why it’s true. A proof is strange, though. It’s abstract and untethered to material experience. “They’re this crazy contact between an imaginary, nonphysical world and biologically evolved creatures,” said the cognitive scientist Simon DeDeo of Carnegie Mellon University, who studies mathematical certainty by analyzing the structure of proofs. “We did not evolve to do this.”

Computers are useful for big calculations, but proofs require something different. Conjectures arise from inductive reasoning — a kind of intuition about an interesting problem — and proofs generally follow deductive, step-by-step logic. They often require complicated creative thinking as well as the more laborious work of filling in the gaps, and machines can’t achieve this combination.

Computerized theorem provers can be broken down into two categories. Automated theorem provers, or ATPs, typically use brute-force methods to crunch through big calculations. Interactive theorem provers, or ITPs, act as proof assistants that can verify the accuracy of an argument and check existing proofs for errors. But these two strategies, even when combined (as is the case with newer theorem provers), don’t add up to automated reasoning.

Simon DeDeo of Carnegie Mellon helped show that people and machines seem to construct mathematical proofs in similar ways.

Courtesy of Simon DeDeo

Plus, the tools haven't been met with open arms, and the majority of mathematicians don't use or welcome them. "They're very controversial for mathematicians," DeDeo said. "Most of them don't like the idea."

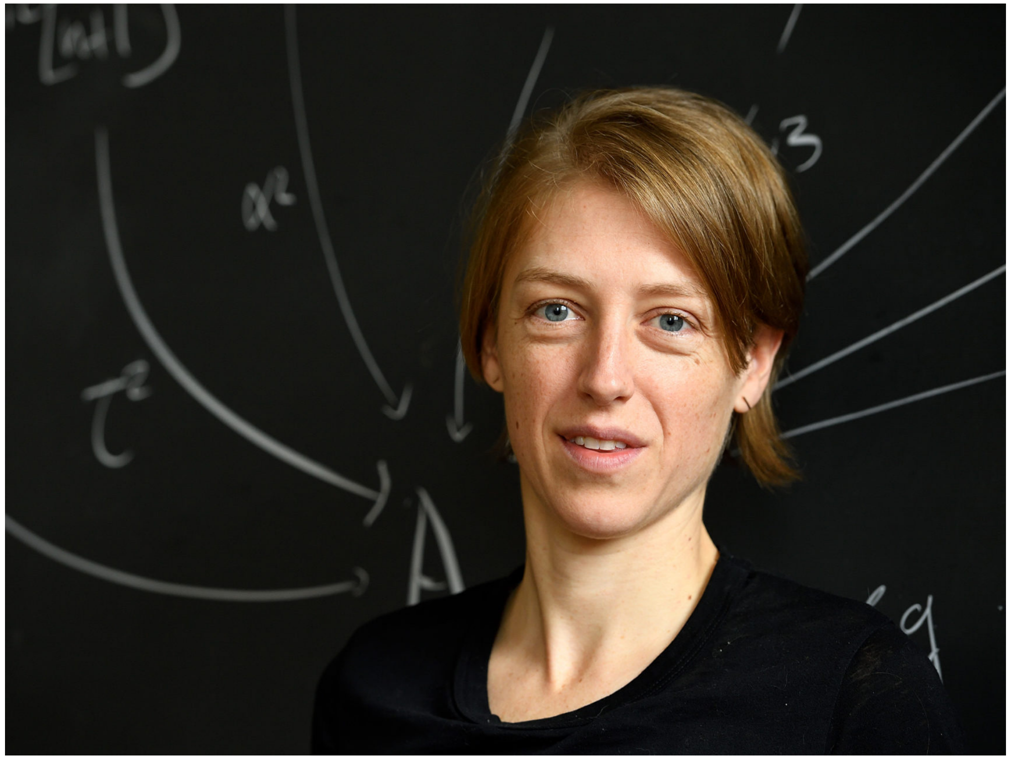
A formidable open challenge in the field asks how much proof-making can actually be automated: Can a system generate an interesting conjecture and prove it in a way that people understand? A slew of recent advances from labs around the world suggests ways that artificial intelligence tools may answer that question. Josef Urban at the Czech Institute of Informatics, Robotics and Cybernetics in Prague is exploring a variety of approaches that use machine learning to boost the efficiency and performance of existing provers. In July, his group reported a set of original conjectures and proofs generated and verified by machines. And in June, a group at Google Research led by Christian Szegedy posted recent results from efforts to harness the strengths of natural language processing to make computer proofs more human-seeming in structure and explanation.

Some mathematicians see theorem provers as a potentially game-changing tool for training undergraduates in proof writing. Others say that getting computers to write proofs is unnecessary for advancing mathematics and probably impossible. But a system that can predict a useful conjecture and prove a new theorem will achieve something new — some machine version of understanding, Szegedy said. And that suggests the possibility of automating reason itself.

Useful Machines

Mathematicians, logicians and philosophers have long argued over what part of creating proofs is fundamentally human, and debates about mechanized mathematics continue today, especially in the deep valleys connecting computer science and pure mathematics.

For computer scientists, theorem provers are not controversial. They offer a rigorous way to verify that a program works, and arguments about intuition and creativity are less important than finding an efficient way to solve a problem. At the Massachusetts Institute of Technology, for example, the computer scientist Adam Chlipala has designed theorem-proving tools that generate cryptographic algorithms — traditionally written by humans — to safeguard internet transactions. Already, his group's code is used for the majority of the communication on Google's Chrome browser.



Emily Riehl of Johns Hopkins University uses theorem provers in teaching students and proof assistants in her own work. "Using a proof assistant has changed the way I think about writing proofs," she said.

Will Kirk/Johns Hopkins University

"You can take any kind of mathematical argument and code it with one tool, and connect your arguments together to create proofs of security," Chlipala said.

In math, theorem provers have helped produce complicated, calculation-heavy proofs that otherwise would have occupied hundreds of years of mathematicians' lives. The Kepler conjecture, which describes the best way to stack spheres (or, historically, oranges or cannonballs), offers a telling example. In 1998, Thomas Hales, together with his student Sam Ferguson, completed a proof using a variety of computerized math techniques. The result was so cumbersome — the results took up 3 gigabytes — that 12 mathematicians analyzed it for years before announcing they were 99% certain it was correct.

The Kepler conjecture isn't the only famous question to be solved by machines. The four-color theorem, which says you only need four hues to color any two-dimensional map so that no two adjoining regions share a color, was settled in 1977 by mathematicians using a computer program that churned through five-colored maps to show they could all be reduced to four. And in 2016, a trio of mathematicians used a computer program to prove a longstanding open challenge called the Boolean Pythagorean triples problem, but the initial version of the proof was 200 terabytes in size. With a high-speed internet connection, a person could download it in a little over three weeks.

Complicated Feelings

These examples are often trumpeted as successes, but they've also added to the debate. The computer code proving the four-color theorem, which was settled more than 40 years ago, was impossible for humans to check on their own. "Mathematicians have been arguing ever since whether or not it's a proof," said the mathematician Michael Harris of Columbia University.

Many mathematicians, like Columbia University's Michael Harris, disagree with the idea that computerized theorem provers are necessary — or that they'll make human mathematicians obsolete.

Béatrice Antolin

Another gripe is that if they want to use theorem provers, mathematicians must first learn to code and then figure out how to express their problem in computer-friendly language — activities that detract from the act of doing math. "By the time I've reframed my question into a form that could fit into this technology, I would have solved the problem myself," Harris said.

Many just don't see a need for theorem solvers in their work. "They have a system, and it's pencil and paper, and it works," said Kevin Buzzard, a mathematician at Imperial College London who three years ago pivoted his work from pure math to focus on theorem provers and formal proofs. "Computers have done amazing calculations for us, but they have never solved a hard problem on their own," he said. "Until they do, mathematicians aren't going to be buying into this stuff."

But Buzzard and others think maybe they should. For one thing, "computer proofs may not be as alien as we think," DeDeo said. Recently, together with Scott Viteri, a computer scientist now at Stanford University, he reverse-engineered a handful of famous canonical proofs (including one from Euclid's *Elements*) and dozens of machine-generated proofs, written using a theorem prover called Coq, to look for commonalities. They found that the networked structure of machine proofs was remarkably similar to the structure of proofs made by people. That shared trait, he said, may help researchers find a way to get proof assistants to, in some sense, explain themselves.

“Machine proofs may not be as mysterious as they appear,” DeDeo said.

Others say theorem provers can be useful teaching tools, in both computer science and mathematics. At Johns Hopkins University, the mathematician Emily Riehl has developed courses in which students write proofs using a theorem prover. “It forces you to be very organized and think clearly,” she said. “Students who write proofs for the first time can have trouble knowing what they need and understanding the logical structure.”

Riehl also says that she’s been increasingly using theorem provers in her own work. “It’s not necessarily something you have to use all the time, and will never substitute for scribbling on a piece of paper,” she said, “but using a proof assistant has changed the way I think about writing proofs.”

Theorem provers also offer a way to keep the field honest. In 1999, the Russian American mathematician Vladimir Voevodsky discovered an error in one of his proofs. From then until his death in 2017, he was a vocal proponent of using computers to check proofs. Hales said that he and Ferguson found hundreds of errors in their original proof when they checked it with computers. Even the very first proposition in Euclid’s *Elements* isn’t perfect. If a machine can help mathematicians avoid such mistakes, why not take advantage of it? (The practical objection, justified or not, is the one suggested by Harris: If mathematicians have to spend their time formalizing math to be understood by a computer, that’s time they’re not spending doing new math.)

But Timothy Gowers, a mathematician and Fields medalist at the University of Cambridge, wants to go even further: He envisions a future in which theorem provers replace human referees at major journals. “I can see it becoming standard practice that if you want your paper to be accepted, you have to get it past an automatic checker,” he said.

Talking to Computers

But before computers can universally check or even devise proofs, researchers first have to clear a significant hurdle: the communication barrier between the language of humans and the language of computers.

Today’s theorem provers weren’t designed to be mathematician-friendly. ATPs, the first type, are generally used to check if a statement is correct, often by testing possible cases. Ask an ATP to verify that a person can drive from Miami to Seattle, for example, and it might search all cities connected by roads leading away from Miami and eventually finding a city with a road leading into Seattle.

Not every mathematician hates theorem provers. Timothy Gowers, of the University of Cambridge, thinks they may one day replace human reviewers at mathematical journals.

The Abel Prize

With an ATP, a programmer can code in all the rules, or axioms, and then ask if a particular conjecture follows those rules. The computer then does all the work. “You just type in the conjecture you want to prove, and you hope you get an answer,” said Daniel Huang, a computer scientist who recently left the University of California, Berkeley, to work at a startup.

But here’s the rub: What an ATP doesn’t do is explain its work. All that calculating happens within the machine, and to human eyes it would look like a long string of 0s and 1s. Huang said it’s impossible to scan the proof and follow the reasoning, because it looks like a pile of random data. “No human will ever look at that proof and be able to say, ‘I get it,’” he said.

ITPs, the second category, have vast data sets containing up to tens of thousands of theorems and proofs, which they can scan to verify that a proof is accurate. Unlike ATPs, which operate in a kind of black box and just spit out an answer, ITPs require human interaction and even guidance along the way, so they're not as inaccessible. "A human could sit down and understand what the proof-level techniques are," said Huang. (These are the kinds of machine proofs DeDeo and Viteri studied.)

ITPs have become increasingly popular in recent years. In 2017, the trio behind the Boolean Pythagorean triples problem used Coq, an ITP, to create and verify a formal version of their proof; in 2005 Georges Gonthier at Microsoft Research Cambridge used Coq to formalize the four-color theorem. Hales also used ITPs called HOL Light and Isabelle on the formal proof of the Kepler conjecture. ("HOL" stands for "higher-order logic.")

Efforts at the forefront of the field today aim to blend learning with reasoning. They often combine ATPs with ITPs and also integrate machine learning tools to improve the efficiency of both. They envision ATP/ITP programs that can use deductive reasoning — and even communicate mathematical ideas — the same way people do, or at least in similar ways.

The Limits of Reason

Josef Urban thinks that the marriage of deductive and inductive reasoning required for proofs can be achieved through this kind of combined approach. His group has built theorem provers guided by machine learning tools, which allow computers to learn on their own through experience. Over the last few years, they've explored the use of neural networks — layers of computations that help machines process information through a rough approximation of our brain's neuronal activity. In July, his group reported on new conjectures generated by a neural network trained on theorem-proving data.

Urban was partially inspired by Andrej Karpathy, who a few years ago trained a neural network to generate mathematical-looking nonsense that looked legitimate to nonexperts. Urban didn't want nonsense, though — he and his group instead designed their own tool to find new proofs after training on millions of theorems. Then they used the network to generate new conjectures and checked the validity of those conjectures using an ATP called E.

The network proposed more than 50,000 new formulas, though tens of thousands were duplicates. "It seems that we are not yet capable of proving the more interesting conjectures," Urban said.

Szegedy at Google Research sees the challenge of automating reasoning in computer proofs as a subset of a much bigger field: natural language processing, which involves pattern recognition in the usage of words and sentences. (Pattern recognition is also the driving idea behind computer vision, the object of Szegedy's previous project at Google.) Like other groups, his team wants theorem provers that can find and explain useful proofs.

Inspired by the rapid development of AI tools like AlphaZero — the DeepMind program that can defeat humans at chess, Go and shogi — Szegedy's group wants to capitalize on recent advances in language recognition to write proofs. Language models, he said, can demonstrate surprisingly solid mathematical reasoning.

His group at Google Research recently described a way to use language models — which often use neural networks — to generate new proofs. After training the model to recognize a kind of treelike structure in theorems that are known to be true, they ran a kind of free-form experiment, simply asking the network to generate and prove a theorem without any further guidance. Of the thousands of generated conjectures, about 13% were both provable and new (meaning they didn't duplicate other theorems in the database). The experiment, he said, suggests that the neural net could teach itself a kind of understanding of what a proof looks like.

"Neural networks are able to develop an artificial style of intuition," Szegedy said.

Of course, it's still unclear whether these efforts will fulfill Cohen's prophecy from over 40 years ago. Gowers has said that he thinks computers will be able to out-reason mathematicians by 2099. At first, he predicts, mathematicians will enjoy a kind of golden age, "when mathematicians do all the fun parts and computers do all the boring parts. But I think it will last a very short time."

After all, if the machines continue to improve, and they have access to vast amounts of data, they should become very good at doing the fun parts, too. "They will learn how to do their own prompts," Gowers said.

Harris disagrees. He doesn't think computer provers are necessary, or that they will inevitably "make human mathematicians obsolete." If computer scientists are ever able to program a kind of synthetic intuition, he says, it still won't rival that of humans. "Even if computers understand, they don't understand in a human way."