Machine-verifiable Proofs

David Paul - David.Paul@une.edu.au

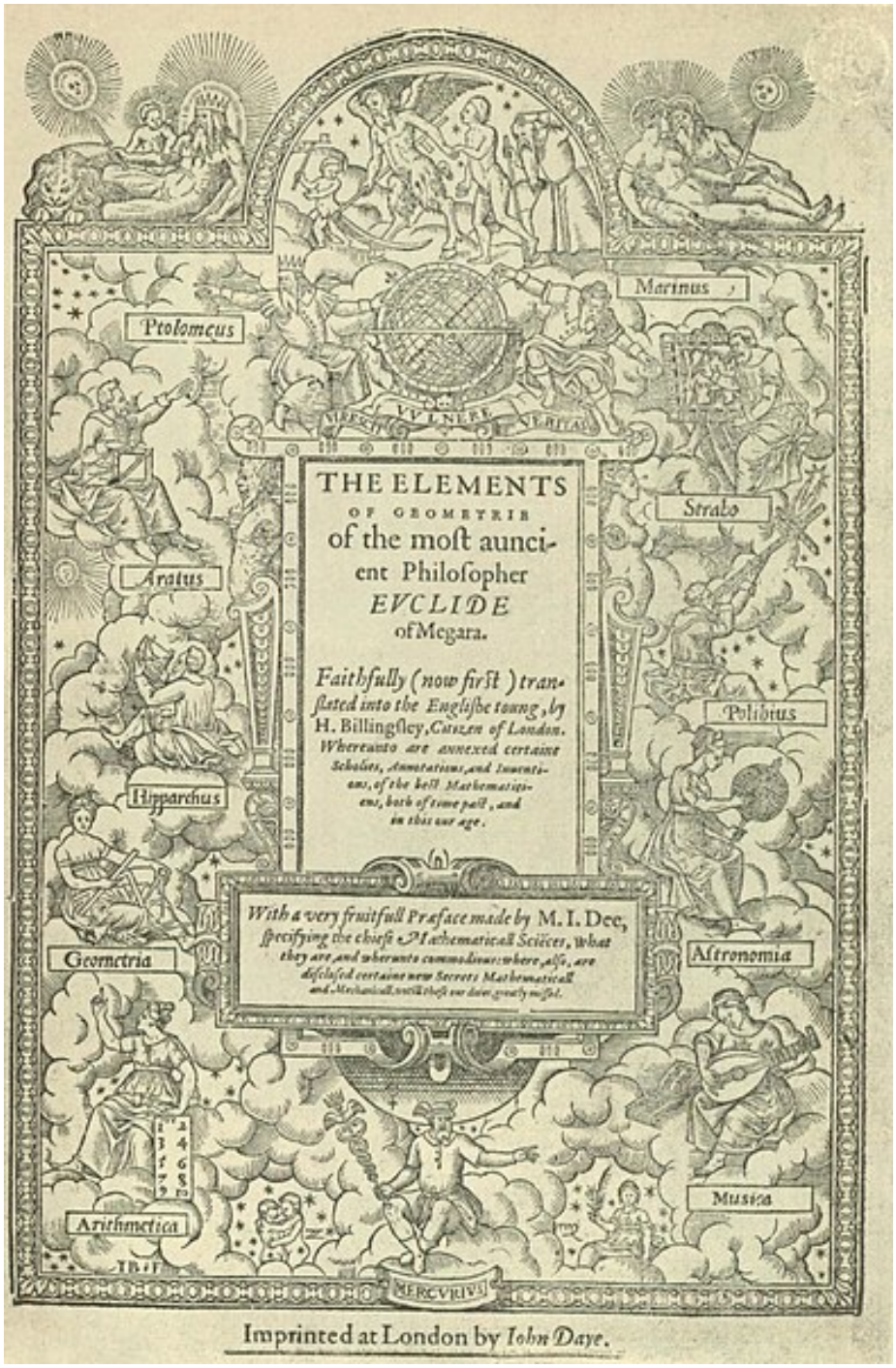
2023-07-28

When I was turing five years old, I asked my grandparents for a calculator for my birthday. Shortly afterwards, my mother showed me the square root button and how you could take any value, press this button, and get a new value that, when you multiplied it by itself, got you back to the original value. I raced to my sister saying "Look at this - it's magic!" and, of course, my sister couldn't care less.

Now, as a computer scientist, I have a better understanding of how the calculator could do this magic trick, but I don't think it takes away any of the magic. Computers can calculate better than any humans, and have been able to for a long time. But what about reasoning? Computers are getting better, and I don't think it's too far in the future that interactive theorem provers will be helping mathematicians advance all areas of maths.

So what is an interactive theorem prover? And how does it allow machine-verifiable proofs? That's what I'm going to introduce today. This is an area that I haven't really done any research in yet, but would quite like to. And I'd certainly be happy to collaborate if it piques anyone's interest.

Euclid's Elements



Euclid's Elements is probably the most successful and influential textbook that's ever existed. It was first published around 300BC, and is still in print today. This is a picture of the titlepage of the first English translation, from 1570.

Mathematics has a wonderful history of starting from some simple assumptions (now called axioms) and then logically going step-by-step to prove more complex results. If you agree with the axioms, then reason means you need to agree with the later results.

This book was really the first to start that tradition. It begins with five postulates which were then used to prove over 450 propositions. But there's a problem - the fifth postulate isn't always true. It just took until the 1800s for mathematicians to seriously consider other options.

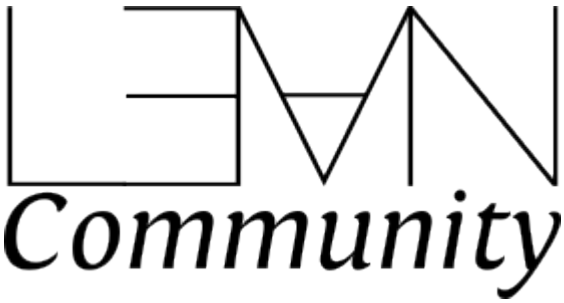
Errors in Mathematics

Even with well-agreed axioms, there's still a lot of errors in mathematics research.

This is a picture of Sir Andrew Wiles, who solved a very famous problem called Fermat's Last Theorem back in the '90s. He presented his proof, and everybody celebrated, but then a gap in the proof was found and it took another year or so to fix.

Proofs are getting so complex that it's easy for even experts to miss something. Hopefully it's eventually caught through peer-review, as was the case with Wiles, though sometimes errors are found years later, after people have built up more work based on the incorrect result. It would be better if these errors could be caught more quickly - or perhaps even automatically.

Lean Theorem Prover



I'm going to be talking about the Lean theorem prover. There are lots of others around, but this is the one I've looked at the most.

Lean uses a set of axioms based on dependent type theory. If you believe the axioms in this base, then you believe proofs in Lean.

Lean is an open source, fully functional programming language originally designed for formal verification that software is correct. However, it can also be used as an interactive theorem prover, and has a large maths library available.

Interactive Theorem Prover

So what is an interactive theorem prover? It's a software tool to assist with formal proofs using human-machine collaboration.

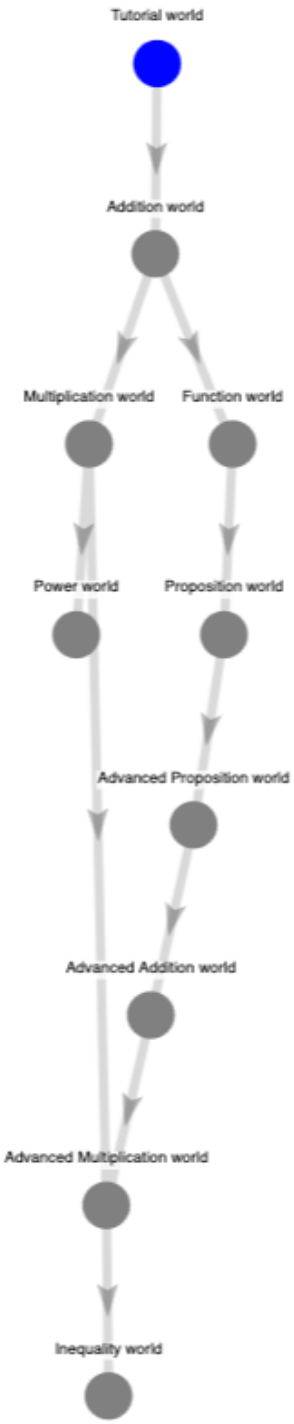
With Lean, you write some Lean statements, which can look reasonably like normal mathematics, or a combination of maths and computer code, and Lean will keep track of the current state: the current hypotheses you have, the goal you're currently trying to prove, and any other goals that have come up along the way.

You progress by telling Lean to use a tactic to move from the current state to a new state, hopefully closer to your solution. Lean's mathlib already includes

most undergraduate mathematics, as well as a bunch of more advanced concepts, which you can also use to help you advance.

Lean automatically determines whether there are any gaps in your proof and presents them as an incomplete goal, so if Lean says there are no goals remaining, you can be sure that your proof is correct.

How to get started



One of the best ways to get started with Lean is with Kevin Buzzard's Natural Number Game. It's a Web-based game, so you don't need to install anything. It starts from Peano's axioms, which are very basic, and guides you through the process of proving quite a lot about positive whole numbers - from things like a + b = b + a, or if neither a and b are 0, then a times b can't be 0 either.

If you like what you see with the natural number game, there are lots of resources online for going further, but installing Lean and playing around with it is a great start. I'm happy to help with that if you're interested.

What have interactive theorem provers done?

- 2004: Verified Four colour theorem
- 2015: The Kepler Conjecture
- 2017: Undergraduate mathematics
- 2018: Some graduate level mathematics
- 2019: Definition of perfectoid space
- 2021: Liquid Tensor Experiment
- 2022: Sphere eversion

So Interative Theorem Provers exist, but are they very useful?

While there were some results before, the first I'll talk about is the Four colour theorem - verifying that, if you have a map, you only need four colours to be able to fill it in so that no neighbouring countries have the same colour. This was proven in the 70s using a large computer search - so you had to trust the programs that implemented the proof. With the interactive theorem prover proof, you can just trust the axioms of the prover.

Next, in 1998, a linear programming method was used to determine the optimal way to pack spheres in three dimensions - think storing oranges into a box. After five years of review, one of the top maths journals came back saying that the reviewers couldn't verify part of the proof because there was too much to calculate. A team of over 20 people then spent 12 years formalising the proof of Kepler's Conjecture - that the best way to pack oranges into boxes is indeed the way grocers have always done it.

In 2017, a mathematician, Kevin Buzzard, started teaching Lean to undergraduate students at Imperial College London (and a lot of this history comes from one of his presentations). They started formalising what they were learning into Lean's mathlib, and soon got into some more ambitious projects, such as formalising some graduate mathematics. Mathlib is a gigantic collaborative project that anybody can add to, and has a fair bit of maths in it.

For example, in 2018, Peter Scholze won a Fields Medal, often called the Nobel prize of mathematics, in part because of his work on perfectiod spaces. The defintion of a perfectoid space was soon formalise in Lean. Scholze took notice and challenged the theorem prover community to prove a key part of the fundamental theorem of liquid vector spaces that he and Clausen had proven in 2019, because

he wasn't certain it was correct. The community took up the challenge and had a formalised proof in 2021 - perhaps the first time a complex proof about complex mathematical objects has been formalised - and it even simplified the proof because the formal system let them realise that some claims could be weakened to still get the required result.

Lean's even been used for more continuous things, like verifying sphere eversion.

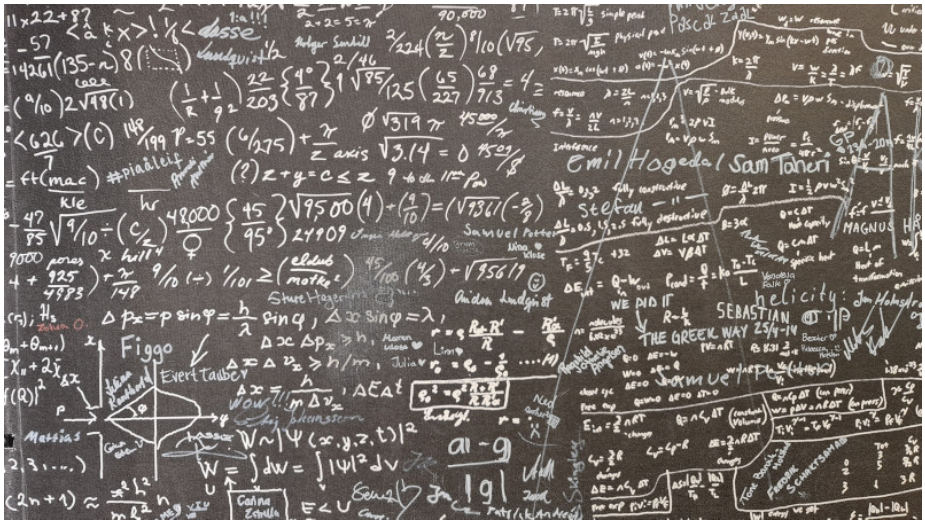
So Lean has been proven to be useful in:

- teaching mathematics - students can step through a proof at their own pace and make sure they understand what is going on
- understanding mathematics - Kevin Buzzard said that formalising perfectoid spaces was a great way for him to understand them because he had to get it right down to the axioms
- verifying research - we know the fundamental theorem of liquid vector spaces is correct because of the formal proof. And some other results have even been verified using interactive theorem provers more quickly than reviewers have gotten back on submitted journal articles.

What does the future hold?

Research Directions

- Formalising existing mathematics
- Writing formal proofs for new results
- Developing new tactics to help solve problems



So, as I mentioned, this is an area in which I'd like to get involved, but haven't really yet. It is a rapidly developing area, and there's a lot of interesting things happening. I'd certainly be interested in things like formalising existing

mathematics or creating formal proofs to be published with new results (who knows? In the future, this may be required by more journals - currently only a few formal proof journals require it, but in the coming decades, it might be more and more prevelant). I'd also be insterested in developing new tactics to help solve problems. For example, some people have recently created a tactic that communicates with GPT-4, which can be useful for suggesting next steps when you get stuck. If you're interested, then please get in touch, and maybe we can collaborate.