Computer theorem provers in the classroom?

K. Buzzard, Imperial College London

19th April 2022, OLSUME

Before we start

Two things before we start:

1) Thank you Tara and Haynes for the invitation, and thanks to you all for coming!

2) I am not a computer scientist – I am an algebraic number theorist who 5 years ago knew *absolutely nothing* about this stuff.

Overview

I started using the Lean theorem prover in 2017 for reasons connected to teaching.

Lean has completely changed my *research* life. But I'm not going to talk much about that.

I will concentrate on how I've been using Lean in the classroom with undergraduates (and occasionally Masters/PhD students).

The beginning

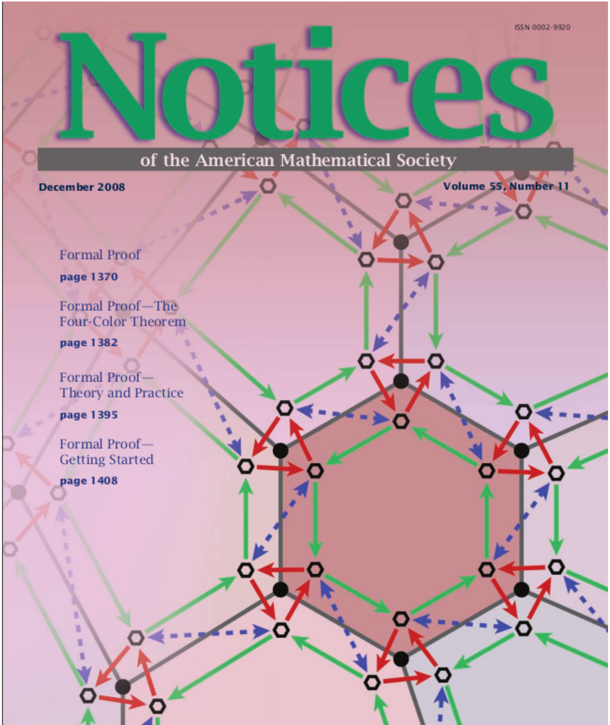
In 2017 I found myself with less childcare.

I was also given Imperial College London's "introduction to proof" course to teach, so thought I would shake it up.

I had read about computer theorem provers in the December 2008 issue of the Notices of the AMS.

Question: Can I use a theorem prover to show weak students what we want from them?

Spoiler alert: I failed to do this.


ISSN 0002-9920
Notices
of the American Mathematical Society
December 2008
Volume 55, Number 11
Formal Proof
page 1370
Formal Proof—The Four-Color Theorem
page 1382
Formal Proof—Theory and Practice
page 1395
Formal Proof—Getting Started
page 1408

The beginning

I spent a few weeks in June/July 2017 surveying theorem provers.

I found very little in the theorem prover literature for people like me (beginner, no CS background, professional number theorist). I also found essentially nobody working in the area who had my profile.

I learnt that modern theorem provers use “tactics” and until you know about 20 of them you can’t really express your ideas.

I learnt that you also can’t get anywhere until you know your way around a robust maths library which other people wrote.

I also learnt that this stuff was a lot of fun and in fact highly addictive.

A revelation

And then I (virtually) went to a talk by Tom Hales at the Newton Institute.

Hales works in a related area to me. He talked about the idea of explaining to theorem provers the statements of modern research number theory.

Slide from Hales' talk

Formalizing statements in Automorphic Representation Theory (a branch of number theory).

This will require a great deal of work just to state the theorems that are proved: algebraic geometry (schemes, motives, stacks, moduli spaces, and sheaves), measure theory and functional analysis, algebra (rings, modules, Galois theory, homological algebra, derived categories), category theory, complex analysis (L-functions and modular forms), class field theory (local and global), Lie theory and linear algebraic groups (Cartan classification and structure theory), representation theory (infinite dimensional, spectral theory), Shimura varieties, locally symmetric spaces, Hecke operators, cohomology (singular, deRham, intersection homology, l-adic), rigid geometry, perfectoid spaces,. . .

Switching to Lean

Hales suggested that Lean would be a good theorem prover for this task.

So I started learning about Lean. It was a brand new theorem prover with very little mathematics in it.

Principal designer: Leo de Moura at Microsoft Research. Free and open source.

There was a book – “theorem proving in Lean” by Jeremy Avigad et al. It had very little mathematics in it.

I read the book, and then got started on Lean solutions for the ten problem sheets for my intro to proofs course.

Sheet 1 question 1 part 1

"True or false? If x is a real number, then $x^2 - 3x + 2 = 0 \implies x = 1$".

Model answer – "false; x can be 2".

First problem: translating the question into Lean.

What is x? What does the question *mean*?

I realised that if $x = 1$ the assertion is true.

I then realised that if $x = 37$ the assertion is also true. In fact the assertion is true with probability 1.

Sheet 1 question 1 part 1

Conclusion: "True or false? If x is a real number, then $x^2 - 3x + 2 = 0 \implies x = 1$" **means**

"True or false? For all real numbers x, $x^2 - 3x + 2 = 0 \implies x = 1$."

No wonder some of the students struggle.

I typed this question into Lean, and then typed in the Lean-speak for "false: set $x = 2$".

I expected the goal to be solved. Instead, it became two goals:

- $2^2 - 3 \times 2 + 2 = 0$;
- $2 \neq 1$.

real arithmetic is hard

I logged onto the Lean chat and asked how to prove $2^2 - 3 \times 2 + 2 = 0$ in Lean.

"Which 2 is that?"

"The real number 2."

"Oh that's hard. But it is true!" (The claim is false for the natural number 2).

Similarly $2 \neq 1$ was also hard (for real numbers).

The course is supposed to be starting in two months and we can't prove $2 \neq 1$.

A week later, a computer science MSc student called Mario Carneiro writes a new tactic `norm_num` which solves both of these questions automatically, and adds the tactic to the maths library.

Sheet 2 : Square roots

Second problem sheet in the course is about square roots of real numbers. They don't exist in Lean, so I define $\sqrt{x} := \mathrm{Sup}\{y : \mathbb{R} \mid y^2 \leq x\}$ for non-negative reals.

The computer scientists laugh at me for writing a function with an unused input.

They write a square root function using the same formula but without assuming $x \geq 0$. The function outputs junk if the input is negative.

Their version works much better than mine in practice.

I write a 200 line proof that $\sqrt{3}$ is irrational; it takes me hours.

LONDON MATHEMATICAL SOCIETY
EST. 1865


NEWSLETTER

Issue: 484 - September 2019

ETHICS IN MATHEMATICS | DRINFELD MODULES AND FACTORISATION | COMPUTERS AND PROOF

Sheets 3/4: complex numbers

Sheet 3 from the course is on complex numbers. The maths library doesn't have them. I make my first pull request; I define a complex number to be a pair of real numbers, and prove they're a ring.

Sheet 4 uses $e^{i\theta} = \cos(\theta) + i\sin(\theta)$ which needs a robust theory of rearrangements of infinite sums in complete metric spaces, and definitions of exponential, sine and cosine plus proofs that they converge absolutely.

I now realise that proving this is a little research project.

I also realise that the course I am about to teach is not really designed for theorem provers.

Furthermore, I feel that it would be hypocritical (and unacceptable) to change the course.

Number theory

The next two sheets (5,6) are on number theory (congruences etc) and they are much nicer to do.

My proofs are still extremely long and inelegant.

The fact that $\mathbb{N} \subseteq \mathbb{Z} \subseteq \mathbb{Q}$ is *false* in Lean makes some stuff very awkward.

The next three sheets (7,8,9) are on functions, relations, and equivalence relations, and these are *really really really easy* to do in Lean.

Counting equivalence relations.

Sheet 9 question: “How many equivalence relations are there on a set with two elements?”

I’ve seen smart students say “infinitely many”.

This used to confuse me. The argument goes like this.

We’ve proved that if S is a set of integers and if n is a positive integer then being congruent mod n is an equivalence relation on S.

Now set $n = 1, 2, 3, 4, \ldots$. *Obviously all these equivalence relations are different*.

Now let S be a set of two integers and we’re done.

Equality of equivalence relations

What does it mean for two equivalence relations to be equal?

Is Fermat's Last Theorem equal to $2+2=4$?

If $x \sim_1 y$ is defined to be Fermat's Last Theorem, and $x \sim_2 y$ is defined to be $2+2=4$, is a student supposed to realise that these equivalence relations are supposed to be equal?

I realised that *I* was learning my course better.

“Intuitive arguments”

The final sheet (10) is on $F - E + V = 2$ and the proof in my notes contains lines like “Now imagine the polyhedron is made out of glass, and put your eye very close to one of the faces”. Was this even mathematics?

I did not formalise this proof.

Autumn 2017

When Autumn came, I just leapt in.

I did the first couple of lectures on logic, switching between Lean, the visualiser, and pdf slides.

I made Lean versions of most of the problem sheet questions available to the students. Lean was really difficult to install, and the problem sheets were *really* difficult to do in Lean.

I was still working on them, and sometimes I felt like a struggling 1st year all over again.

The Xena Project

I also started a club called the Xena Project – “mathematicians learning Lean”.

The top students came. Not part of the plan – but not unexpected.

Within a few months we were doing original research.

All four undergraduates involved are now doing PhDs.

Autumn 2017

I used Lean when doing functions and equivalence relations as well.

A smart student said to me “I didn’t understand your lecture on equivalence relations, so I formalised them myself in Lean and now I understand them”.

I would blog about interesting problems I found. Such as a fun way to build the natural numbers from scratch and prove basic stuff about them.

Proving stuff like “if $a \leq b$ and $b \leq a$ then $a = b$” is harder than you think.
I made a game about this and it’s a nice introduction to Lean, of a *completely* different nature to the other stuff out there.

Summer 2018

I got some funding from Imperial's teaching innovation grant scheme to run some Lean projects and do hire an education specialist for a few months.

Projects: Summer 2018 I ran summer projects with 20 students, who made linear maps, quadratic reciprocity, sums of squares, fundamental groups. . . .

What I had not appreciated was that part of the fun was that *it had never been done before in Lean*.

Autumn 2018

I had of course realised that I didn't know what I was doing (other than having a lot of fun). We hired Athina Thoma to take a look (a student of Paola Iannone in Loughborough).

In Autumn 2018 I did basically the same course with the same Lean, but in a far more polished way. Athina interviewed and surveyed students.

You can read her paper with Iannone to see their conclusions.

One conclusion: you cannot expect a weak student who is struggling with the material to also take on the task of learning how to use a completely optional and extremely complicated computer theorem prover.

Autumn 2019

In 2018–19 there was a huge curriculum review at Imperial College London.

I was on the subcommittee whose job it was to design the new “intro to proofs” module.

I completely rewrote it so it was far more Lean-appropriate early on. First delivery Autumn 2019.

I start with logic, sets, functions and relations (8 hours). Then Marie-Amelie Lawn does natural numbers via Peano, integers, rationals and reals, and then some congruences. I knew that all this stuff would be *really nice* to do in Lean.

Autumn 2019

New Lean-friendly syllabus. I do a bunch of Lean in the lectures, all proofs really slick.

Lean versions of all problem sheets.

Big problem: problem sheets *still much too hard*.

You need to know 20 tactics/theorems to prove that equivalence relations are the same thing as partitions.

Conversely, I *had* to ask difficult questions on these sheets (I can't just make the questions easier because of my own agenda).

Autumn 2019

Another problem: all the easy stuff is now in Lean’s maths library.

“Sure you can prove the complexes are algebraically closed, but we don’t need your proof in our maths library because we already have one.”

Autumn 2019

Summary:

- I am not attracting weak undergraduates.
- I have now become an expert in Lean, and (with Commelin and Massot) have formalised the definition of a perfectoid space.
- Some of my undergraduates are also now Lean experts.
- I have understood that I cannot teach Lean and mathematics at the same time to beginners.

I decide that next year I'll try teaching Lean to PhD students in pure mathematics.

Then Covid.

Autumn 2020 (Covid)

In 2020 we moved everything online.

I taught the “intro to proof” module via video lectures; most were Lean-free, and then some optional videos were 100 percent Lean.

Spring 2021

In Spring 2021 I took a term's sabbatical and wrote a graduate Lean course which I delivered to PhD students in Oxford, Warwick, Bristol, Bath, Imperial.

We ran it all on Discord.

15 or so students, many of whom learnt something.

For the first time, I was giving a course whose primary objective was to teach students Lean.

This meant I could now make problem sheets containing *basic logic questions* such as "If $P \implies Q$ and $Q \implies R$, show that $P \implies R$.

Students could learn one tactic at a time. And I didn't have to teach them the mathematics.

Spring 2021

Graduate course attempted to do all of logic, sets, functions, and relations in first two hour slot (on the basis that “they know the material”). The PhD students were just finishing “sheet 1, logic” at the end of the 2 hours.

We went on to do basic theory of limits in real analysis, topology, filters, uniform spaces, quotients and some group cohomology.

A PhD student did a project with me afterwards where they stated the Birch and Swinnerton-Dyer conjecture in Lean.

Autumn 2021

And so to this year; last term I taught the UG course again (again using videos). I am running out of ideas in some sense.

And then this term I've been teaching an UG course: Formalising Mathematics 2022.

Formalising Mathematics 2022

Ran Jan–March 2022.

Optional UG course with compulsary Lean component.

Assessed by three projects and short orals.

1) Formalise something you learnt in 1st year (20%).
2) Formalise something you learnt in 2nd year (30%).
3) Formalise something you learnt this year (50%).

Orals were zero percent and pass/fail, and everyone passed.

Each project: I was looking for 200–400 lines of Lean code and a 5-7 page pdf.

Formalising Mathematics 2022

22 students. Very mixed crowd (for some maths people it was their only pure option, a computing student, some very weak students, some Lean experts).

Plans to “just copy the PhD course” were thrown into chaos when I realised students didn’t *want* to hear about filters and uniform spaces.

Students wanted number theory and graph theory examples.

FM22

22 undergraduates (19 maths, two maths+computing, one computing).

Notes online. All course videos up on YouTube.

Everyone coming out of the course is now Lean-competent.

Of course some came into it better Lean coders than me.

Final projects

Examples of projects which students handed in:

If $2^n - 1$ is a prime then $(2^n - 1)(2^{n-1})$ is perfect.

Existence of a non-trivial solution to Pell's equation.

Hensel's Lemma for complete nonarchimedean fields.

Almost complete classification of integer solutions to $x^2 + 1 = y^3$.

Final projects

The adjunction between topological spaces and “frames”.

Definition of the boundary map in the snake lemma for arbitrary abelian categories.

Distance from a point to a nonempty convex subset in a Hilbert space.

The “fundamental theorem of asset pricing”.

The Chinese Remainder Theorem for commutative rings.

Conclusions

I started wanting to teach the weak 1st years. I have still not figured out how to do this.

I ended up teaching the strong 1st years to become experts.

Now I'm teaching 3rd/4th years to use Lean and formalise mathematics that they *think* they already understand.

More than once people said they felt like they were 1st years again – and I remember feeling like this too.

That's it – thank you.