Formalizing 21st century mathematics in Lean

K. Buzzard

8th April 2021, 68NQRT, Inria

Before we start

1) Thank you to Khalil for the invitation, and thank you to all of you for coming!

2) Sorry but I am going to speak in English :-(

3) I am not a computer scientist – I am a pure mathematician (an algebraic number theorist).

4) I am quite autistic, and I am going to speak honestly and plainly. This sometimes gets me into trouble.

The history

In June 2017 I was 48 years old and I was having a "mid-life crisis" about pure mathematics.

I became worried about the following:

- Some published theorems have incomplete proofs.
- No clear way to fix broken proofs in the literature.
- The refereeing process does not always work.
- The "elders" decide what is OK.

I knew that "computer proof systems" existed.

However I had not seen any mathematics done in these systems which was interesting to *me personally* as a researcher.

But I thought they might be fun to use for undergraduate teaching in my mathematics department.

Big Proof

July 2017: "Computer-aided mathematical proof" workshop at Newton Institute. I was too late to register :-(Talks were live streamed :-)

Talk 1: Tom Hales.

Hales spoke of a vision where more "mainstream" mathematics would be formalised.

He spoke of a future system where it would be possible to formalise statements of theorems of modern mathematics in real time.

I knew well what this entailed – and so did Hales.

A slide from Hales' presentation

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,...

Note the first thing on the list is "schemes" and the last thing is "perfectoid spaces".

Why Lean?

During the questions someone asked Hales which proof system he wanted to use, and Hales suggested Lean.

I respected Hales' work, and trusted him, so I decided to invest time in Lean.

Why not Coq?

Talk 2: Vladimir Veovodsky.

I had attended lectures by Voevodsky in the past, on his proof of the Milnor conjecture. Voevodsky is a Fields Medallist and I hugely respected his work in algebraic geometry.

Voevodsky talked about UniMath.

He spoke a lot about the foundations of mathematics (a topic that I and most working mathematicians are absolutely uninterested in).

He said he had lost interest in formalising his proof of the Milnor conjecture.

He said that his system did not assume the Law of the Excluded Middle (a law which I and most working mathematicians use every day).

I lost interest in UniMath.

The problem.

You computer scientists have developed systems which can eat mathematics.

Most working mathematicians have never heard of them.

Voevodsky was at the IAS but seemed to be no longer interested in standard mathematics.

In fact many of the mathematicians using Coq and Agda are doing foundational stuff, or constructive stuff, or reasoning about "simple objects".

But Hales seemed to be offering a bridge to the kind of things which 95 percent of working mathematicians (including *the important people*) *actually thought about*.

What happened next.

I learnt Lean by formalising the problem sheets I was giving to the undergraduates.

The people on the Lean chat were an essential help for me at this time. Without this chat (and without Mario Carneiro) I could never have got started.

I then started a club – we would meet on Thursday evenings.

Undergraduates got interested, and some got really good (better than me).

Schemes

With two undergraduates (Kenny Lau, Chris Hughes) I formalised the definition of a scheme (basic algebraic geometry, MSc level), and we proved some basic theorems about schemes.

We made many design decision mistakes and I learnt a *lot*, and had a lot of fun too. And we got a publication out of it!

By 2018 I began to feel that these computer proof systems were far more reliable than humans, and could do everything we were doing, but better. I decided to declare war.

2018–2019: the crazy years

I had formalised schemes and I was convinced that Lean could do anything that a mathematician could do (and I still am).

My *personal impression* of the other theorem provers at that time:

The HOL systems: their logic was too weak to do 21st century mathematics (although they are spectacularly good at 20th century analysis).

Coq and Agda: the right logic, but people were working on the wrong questions, as far as 95% of working mathematicians were concerned. There was a cultural issue.

Lean: no mathematicians at all!

A lot of noise.

I spent a lot of 2018 and 2019 going around the world and telling mathematicians that their work was not rigorous, and that Lean would somehow save their souls.

I thought it was important to make as much noise as possible.

I challenged mathematicians to justify their work and started calling people out.

I made some mathematicians annoyed, but this was OK because they were my friends, and people could see that I had a point. There were certainly counterpoints though. e.g. "it has always been like this!"

However...

My mistake, and an apology

I also argued (in public) that Lean was *intrinsically better* than the other systems for doing 21st century mathematics.

I made some computer scientists quite annoyed. This was *not* my intention.

I also learnt that once something is out there on the internet, you can't get rid of it.

I am *genuinely* sorry that I upset some serious Coq users. I wanted to get noticed by the mathematics community, but I said some things which I now regret.

I would love it if "mainstream" geometers/topologists/analysts etc started to use Coq and Isabelle/HOL, but I do not have the time to make this happen.

2019–2020

By 2019 we (Johan Commelin, Patrick Massot and myself) had formalised the definition of a perfectoid space in Lean.

For me, this was a proof that Lean's dependent type theory could handle any modern mathematics – but that it was going to take a *lot* of person-hours.

The work was published in CPP, but its real purpose was to attract the attention of working arithmetic geometers.

This worked well – more arithmetic geometers and number theorists began to appear on the Lean chat.

And then Peter Scholze (Fields Medallist) issued a challenge in December 2020.

The Liquid Tensor Experiment

In 2019–2020 Peter Scholze gave two courses in Bonn, based on his work with Clausen: "Lectures on condensed mathematics" and "Lectures on analytic geometry". He wrote up the main results in two pdf files.

Scholze had become concerned that the community was not checking the work properly.

His comments reminded me of Voevodsky's quote about a technical results by a trusted author not being checked properly.

Ultimately Scholze asked whether a specific theorem (Theorem 9.1 of the analytic geometry course) could be checked in a theorem prover. The Lean community was (as far as I know) the only community who responded.

"I think this may be my most important theorem to date. (It does not really have any applications so far, but I'm sure this will change.) Better be sure it's correct. . . "

Current state of things

A group of mathematicians (mostly professional arithmetic geometers) on the Lean Zulip chat are hard at work on this.

Within two or three months I think we will have done enough to stop Scholze worrying.

And within a few months after that I think we will have a formal proof.

What else will we have gained?

General wins

I personally have found it a very interesting way to learn about some of the details of Scholze's new results.

We are building up a very powerful library of standard MSc level mathematical objects.

Undergraduates have been able to help (e.g., with profinite sets).

It is an open collaboration, and Scholze watches over us.

When we succeed, I will make more noise, which will hopefully attract even more mathematicians into the area.

An idea

Many mathematicians are attracted to Lean because of "the natural number game" – a game where you develop the API for `nat` using tactics. It is a dumb idea but it *works*.

I never mention (a) type theory (b) definitional equality (c) λ terms, proof terms, or functional programming. I just talk about axioms and theorems – I speak a language mathematicians understand. I introduce tactic proofs *extremely slowly*.

There are many other APIs one can develop as games, for example the real numbers, or group theory, or filters. Someone should make a game in Coq, suitable for undergraduate *mathematicians*.

Related: a lot of the manuals for the systems we use are *very hard for mathematicians to read*.

But young mathematicians are interested and this is *very healthy*.

A monopoly is probably bad

I am now very careful to mention Coq and Isabelle/HOL whenever I talk about Lean.

I think that a monopoly is bad, and my actions are partly to blame.

There is still a very long way to go to get mathematicians interested.

But hopefully my actions nowadays are more beneficial to the formal verification community as a whole.

Thank you for coming to my talk!