The future of mathematics?

Kevin Buzzard

Imperial College London

MSR, 5th September 2019.

- I am a pure mathematician, a professor at Imperial College London.

- I am a pure mathematician, a professor at Imperial College London.
- Two years ago I started experimenting with the Lean Theorem Prover (written by Leo de Moura at MSR).

- I am a pure mathematician, a professor at Imperial College London.
- Two years ago I started experimenting with the Lean Theorem Prover (written by Leo de Moura at MSR).
- I now *clearly understand* that software such as Lean is part of the *inevitable future of mathematics*.

- I am a pure mathematician, a professor at Imperial College London.
- Two years ago I started experimenting with the Lean Theorem Prover (written by Leo de Moura at MSR).
- I now *clearly understand* that software such as Lean is part of the *inevitable future of mathematics*. And right now, I can say with high confidence that Lean is easily the most promising of the theorem provers currently available.

- I am a pure mathematician, a professor at Imperial College London.
- Two years ago I started experimenting with the Lean Theorem Prover (written by Leo de Moura at MSR).
- I now *clearly understand* that software such as Lean is part of the *inevitable future of mathematics*. And right now, I can say with high confidence that Lean is easily the most promising of the theorem provers currently available.
- Clear: tools such as Lean will one day help us mathematician

- I am a pure mathematician, a professor at Imperial College London.
- Two years ago I started experimenting with the Lean Theorem Prover (written by Leo de Moura at MSR).
- I now *clearly understand* that software such as Lean is part of the *inevitable future of mathematics*. And right now, I can say with high confidence that Lean is easily the most promising of the theorem provers currently available.
- Clear: tools such as Lean will one day help us mathematicians search for theorems in the literature, and help us to prove theorems. These tools may also change the way we teach.

- I am a pure mathematician, a professor at Imperial College London.
- Two years ago I started experimenting with the Lean Theorem Prover (written by Leo de Moura at MSR).
- I now *clearly understand* that software such as Lean is part of the *inevitable future of mathematics*. And right now, I can say with high confidence that Lean is easily the most promising of the theorem provers currently available.
- Clear: tools such as Lean will one day help us mathematicians search for theorems in the literature, and help us to prove theorems. These tools may also change the way we teach.
- Possible: tools such as Lean will begin to do research semi-autonomously, perhaps uncover problems in the literature.

- I am a pure mathematician, a professor at Imperial College London.
- Two years ago I started experimenting with the Lean Theorem Prover (written by Leo de Moura at MSR).
- I now *clearly understand* that software such as Lean is part of the *inevitable future of mathematics*. And right now, I can say with high confidence that Lean is easily the most promising of the theorem provers currently available.
- Clear: tools such as Lean will one day help us mathematicians search for theorems in the literature, and help us to prove theorems. These tools may also change the way we teach.
- Possible: tools such as Lean will begin to do research semi-autonomously, perhaps uncover problems in the literature. Maybe these tools will replace research mathematicians.

- I am a pure mathematician, a professor at Imperial College London.
- Two years ago I started experimenting with the Lean Theorem Prover (written by Leo de Moura at MSR).
- I now *clearly understand* that software such as Lean is part of the *inevitable future of mathematics*. And right now, I can say with high confidence that Lean is easily the most promising of the theorem provers currently available.
- Clear: tools such as Lean will one day help us mathematicians search for theorems in the literature, and help us to prove theorems. These tools may also change the way we teach.
- Possible: tools such as Lean will begin to do research semi-autonomously, perhaps uncover problems in the literature. Maybe these tools will replace research mathematicians.
- In April, Christian Szegedy from Google told me that he believes that computers will be beating humans at math within ten years.

Who am I?

Who am I?

- I am a number theorist, so interested in questions about $\mathbb{N} := \{0, 1, 2, 3, \ldots\}$.

Who am I?

- I am a number theorist, so interested in questions about $\mathbb{N} := \{0, 1, 2, 3, \ldots\}$.
- For example, I am interested in Fermat's Last Theorem (If $x, y, z, n \in \mathbb{N}$ and $n \geq 3$ then $x^n + y^n = z^n$ only has the obvious solutions with $x = 0$ or $y = 0$).

Who am I?

- I am a number theorist, so interested in questions about $\mathbb{N} := \{0, 1, 2, 3, \ldots\}$.
- For example, I am interested in Fermat's Last Theorem (If $x, y, z, n \in \mathbb{N}$ and $n \geq 3$ then $x^n + y^n = z^n$ only has the obvious solutions with $x = 0$ or $y = 0$).
- The proof of Fermat's Last Theorem is long, and structurally extremely complex.

Who am I?

- I am a number theorist, so interested in questions about $\mathbb{N} := \{0, 1, 2, 3, \ldots\}$.
- For example, I am interested in Fermat's Last Theorem (If $x, y, z, n \in \mathbb{N}$ and $n \geq 3$ then $x^n + y^n = z^n$ only has the obvious solutions with $x = 0$ or $y = 0$).
- The proof of Fermat's Last Theorem is long, and structurally extremely complex. The advent of the internet means that proofs are getting longer.

Who am I?

- I am a number theorist, so interested in questions about $\mathbb{N} := \{0, 1, 2, 3, \ldots\}$.
- For example, I am interested in Fermat's Last Theorem (If $x, y, z, n \in \mathbb{N}$ and $n \geq 3$ then $x^n + y^n = z^n$ only has the obvious solutions with $x = 0$ or $y = 0$).
- The proof of Fermat's Last Theorem is long, and structurally extremely complex. The advent of the internet means that proofs are getting longer.
- Nervousness about the state of the mathematical literature was one reason I started to experiment with computer theorem provers.

- Four years ago: I tried and failed to learn Haskell.

- Four years ago: I tried and failed to learn Haskell.
- Three yea

- Four years ago: I tried and failed to learn Haskell.
- Three years ago: I was given Imperial College's "Introduction to proof" course to teach.
- Two and a half years ago : I tried to learn Haskell again.

- Four years ago: I tried and failed to learn Haskell.
- Three years ago: I was given Imperial College's "Introduction to proof" course to teach.
- Two and a half years ago : I tried to learn Haskell again.
- June 2017 : I stumbled upon a mathoverflow question about the Coq proof of the odd order theorem.

- Four years ago: I tried and failed to learn Haskell.
- Three years ago: I was given Imperial College's "Introduction to proof" course to teach.
- Two and a half years ago : I tried to learn Haskell again.
- June 2017 : I stumbled upon a mathoverflow question about the Coq proof of the odd order theorem.
- I spent two weeks playing with Coq.

- Four years ago: I tried and failed to learn Haskell.
- Three years ago: I was given Imperial College's "Introduction to proof" course to teach.
- Two and a half years ago : I tried to learn Haskell again.
- June 2017 : I stumbled upon a mathoverflow question about the Coq proof of the odd order theorem.
- I spent two weeks playing with Coq.
- And then I saw Tom Hales' talk at Big Proof 2017.

- Four years ago: I tried and failed to learn Haskell.
- Three years ago: I was given Imperial College's "Introduction to proof" course to teach.
- Two and a half years ago : I tried to learn Haskell again.
- June 2017 : I stumbled upon a mathoverflow question about the Coq proof of the odd order theorem.
- I spent two weeks playing with Coq.
- And then I saw Tom Hales' talk at Big Proof 2017.
- I realised there was perhaps nothing stopping us from formalising all of mathematics, *in theory*.

- Four years ago: I tried and failed to learn Haskell.
- Three years ago: I was given Imperial College's "Introduction to proof" course to teach.
- Two and a half years ago : I tried to learn Haskell again.
- June 2017 : I stumbled upon a mathoverflow question about the Coq proof of the odd order theorem.
- I spent two weeks playing with Coq.
- And then I saw Tom Hales' talk at Big Proof 2017.
- I realised there was perhaps nothing stopping us from formalising all of mathematics, *in theory*.
- I chose Lean, because of Hales. I stayed with Lean because of a firm belief that it is the only system ready for this lofty goal.

- Four years ago: I tried and failed to learn Haskell.
- Three years ago: I was given Imperial College's "Introduction to proof" course to teach.
- Two and a half years ago : I tried to learn Haskell again.
- June 2017 : I stumbled upon a mathoverflow question about the Coq proof of the odd order theorem.
- I spent two weeks playing with Coq.
- And then I saw Tom Hales' talk at Big Proof 2017.
- I realised there was perhaps nothing stopping us from formalising all of mathematics, *in theory*.
- I chose Lean, because of Hales. I stayed with Lean because of a firm belief that it is the only system ready for this lofty goal.
- And now I never want to go back to pen and paper mathematics – I am beginning to mistrust it.

- Four years ago: I tried and failed to learn Haskell.
- Three years ago: I was given Imperial College's "Introduction to proof" course to teach.
- Two and a half years ago : I tried to learn Haskell again.
- June 2017 : I stumbled upon a mathoverflow question about the Coq proof of the odd order theorem.
- I spent two weeks playing with Coq.
- And then I saw Tom Hales' talk at Big Proof 2017.
- I realised there was perhaps nothing stopping us from formalising all of mathematics, *in theory*.
- I chose Lean, because of Hales. I stayed with Lean because of a firm belief that it is the only system ready for this lofty goal.
- And now I never want to go back to pen and paper mathematics – I am beginning to mistrust it.
- So my personal main goal at this point is to bring other mathematicans into the area, so things begin to happen more quickly.

My work with mathematics undergraduates:

- In 2017 I started a blog, and an undergraduate club (the Xena project – follow me on Twitter!). We meet on Thursday nights. The objective: I teach them Lean, they try to use Lean to do the maths they're learning, or want to learn.

My work with mathematics undergraduates:

- In 2017 I started a blog, and an undergraduate club (the Xena project – follow me on Twitter!). We meet on Thursday nights. The objective: I teach them Lean, they try to use Lean to do the maths they're learning, or want to learn.
- October 2017: Spike in week 1 quickly receded when we realised nobody knew what they were doing.

My work with mathematics undergraduates:

- In 2017 I started a blog, and an undergraduate club (the Xena project – follow me on Twitter!). We meet on Thursday nights. The objective: I teach them Lean, they try to use Lean to do the maths they're learning, or want to learn.
- October 2017: Spike in week 1 quickly receded when we realised nobody knew what they were doing.
- Only two survivors – Chris Hughes and Kenny Lau. Both now experts – better than me.

My work with mathematics undergraduates:

- In 2017 I started a blog, and an undergraduate club (the Xena project – follow me on Twitter!). We meet on Thursday nights. The objective: I teach them Lean, they try to use Lean to do the maths they're learning, or want to learn.
- October 2017: Spike in week 1 quickly receded when we realised nobody knew what they were doing.
- Only two survivors – Chris Hughes and Kenny Lau. Both now experts – better than me.
- Summer 2018 – summer project with me and 20 undergradutes, formalising our curriculum, funded by Imperial.

My work with mathematics undergraduates:

- In 2017 I started a blog, and an undergraduate club (the Xena project – follow me on Twitter!). We meet on Thursday nights. The objective: I teach them Lean, they try to use Lean to do the maths they're learning, or want to learn.
- October 2017: Spike in week 1 quickly receded when we realised nobody knew what they were doing.
- Only two survivors – Chris Hughes and Kenny Lau. Both now experts – better than me.
- Summer 2018 – summer project with me and 20 undergradutes, formalising our curriculum, funded by Imperial.
- October 2018 – surge in week one took a lot longer to die down.

My work with mathematics undergraduates:

- In 2017 I started a blog, and an undergraduate club (the Xena project – follow me on Twitter!). We meet on Thursday nights. The objective: I teach them Lean, they try to use Lean to do the maths they're learning, or want to learn.
- October 2017: Spike in week 1 quickly receded when we realised nobody knew what they were doing.
- Only two survivors – Chris Hughes and Kenny Lau. Both now experts – better than me.
- Summer 2018 – summer project with me and 20 undergradutes, formalising our curriculum, funded by Imperial.
- October 2018 – surge in week one took a lot longer to die down. We sort-of knew what we were doing.

My work with mathematics undergraduates:

- In 2017 I started a blog, and an undergraduate club (the Xena project – follow me on Twitter!). We meet on Thursday nights. The objective: I teach them Lean, they try to use Lean to do the maths they're learning, or want to learn.
- October 2017: Spike in week 1 quickly receded when we realised nobody knew what they were doing.
- Only two survivors – Chris Hughes and Kenny Lau. Both now experts – better than me.
- Summer 2018 – summer project with me and 20 undergradutes, formalising our curriculum, funded by Imperial.
- October 2018 – surge in week one took a lot longer to die down. We sort-of knew what we were doing.
- October 2019 – it's going to be interesting.

Now it's 2019, and what have Imperial maths undergraduates formalised in Lean?

Now it's 2019, and what have Imperial maths undergraduates formalised in Lean?

- The theorem of quadratic reciprocity,
- Sylow's theorems,
- the fundamental theorem of algebra,
- matrices and bilinear maps,
- the theory of localisation of rings,
- the sine, cosine and exponential functions,
- tensor products of modules,
- Lots and lots of other *undergraduate and MSc level things*.

Now it’s 2019, and what have Imperial maths undergraduates formalised in Lean?

- The theorem of quadratic reciprocity,
- Sylow’s theorems,
- the fundamental theorem of algebra,
- matrices and bilinear maps,
- the theory of localisation of rings,
- the sine, cosine and exponential functions,
- tensor products of modules,
- Lots and lots of other *undergraduate and MSc level things*.

Ellen Arlt, Chris Hughes, Sangwoo Jo, Kenny Lau, Guy Leroy, Amelia Livingston, Jean Lo, Rohan Mitta, Blair Shi, Abhimanyu Pallavi Sudhir, Calle Sönne, Andreas Swerdlow – all of these people have formalised something in Lean which has ended up in Lean’s maths library.

Now it's 2019, and what have Imperial maths undergraduates formalised in Lean?

- The theorem of quadratic reciprocity,
- Sylow's theorems,
- the fundamental theorem of algebra,
- matrices and bilinear maps,
- the theory of localisation of rings,
- the sine, cosine and exponential functions,
- tensor products of modules,
- Lots and lots of other *undergraduate and MSc level things*.

Ellen Arlt, Chris Hughes, Sangwoo Jo, Kenny Lau, Guy Leroy, Amelia Livingston, Jean Lo, Rohan Mitta, Blair Shi, Abhimanyu Pallavi Sudhir, Calle Sönne, Andreas Swerdlow – all of these people have formalised something in Lean which has ended up in Lean's maths library.

Sian Carey, Anca Ciobanu, Clara List and Ramon Fernandez Mir have all formalised mathematics in Lean as part of projects.

See Computers and Mathematics, London Mathematical Society newsletter, September 2019 (pages 32 to 36) for more details of my work with undergraduates.

See Computers and Mathematics, London Mathematical Society newsletter, September 2019 (pages 32 to 36) for more details of my work with undergraduates.

The main frustrations for undergraduates were *pragmatic* rather than foundational.

See Computers and Mathematics, London Mathematical Society newsletter, September 2019 (pages 32 to 36) for more details of my work with undergraduates.

The main frustrations for undergraduates were *pragmatic* rather than foundational. For example, lack of appropriate documentation (I will fix this).

See Computers and Mathematics, London Mathematical Society newsletter, September 2019 (pages 32 to 36) for more details of my work with undergraduates.

The main frustrations for undergraduates were *pragmatic* rather than foundational. For example, lack of appropriate documentation (I will fix this).
Genuine dumb problem: it's hard for a maths UG to install Lean and its maths library on Windows, because they need to use the command line. We can use CoCalc instead – it costs money, but I have run some successful webinars on it.

See Computers and Mathematics, London Mathematical Society newsletter, September 2019 (pages 32 to 36) for more details of my work with undergraduates.

The main frustrations for undergraduates were *pragmatic* rather than foundational. For example, lack of appropriate documentation (I will fix this).
Genuine dumb problem: it's hard for a maths UG to install Lean and its maths library on Windows, because they need to use the command line. We can use CoCalc instead – it costs money, but I have run some successful webinars on it.

Conclusions: it is possible to teach undergraduate mathematicians how to do some of their homework in Lean.

See Computers and Mathematics, London Mathematical Society newsletter, September 2019 (pages 32 to 36) for more details of my work with undergraduates.

The main frustrations for undergraduates were *pragmatic* rather than foundational. For example, lack of appropriate documentation (I will fix this).
Genuine dumb problem: it's hard for a maths UG to install Lean and its maths library on Windows, because they need to use the command line. We can use CoCalc instead – it costs money, but I have run some successful webinars on it.

Conclusions: it is possible to teach undergraduate mathematicians how to do some of their homework in Lean. Forthcoming paper by Iannone and Thoma will say something more formal about what I have achieved so far.

See Computers and Mathematics, London Mathematical Society newsletter, September 2019 (pages 32 to 36) for more details of my work with undergraduates.

The main frustrations for undergraduates were *pragmatic* rather than foundational. For example, lack of appropriate documentation (I will fix this).
Genuine dumb problem: it's hard for a maths UG to install Lean and its maths library on Windows, because they need to use the command line. We can use CoCalc instead – it costs money, but I have run some successful webinars on it.

Conclusions: it is possible to teach undergraduate mathematicians how to do some of their homework in Lean. Forthcoming paper by Iannone and Thoma will say something more formal about what I have achieved so far.

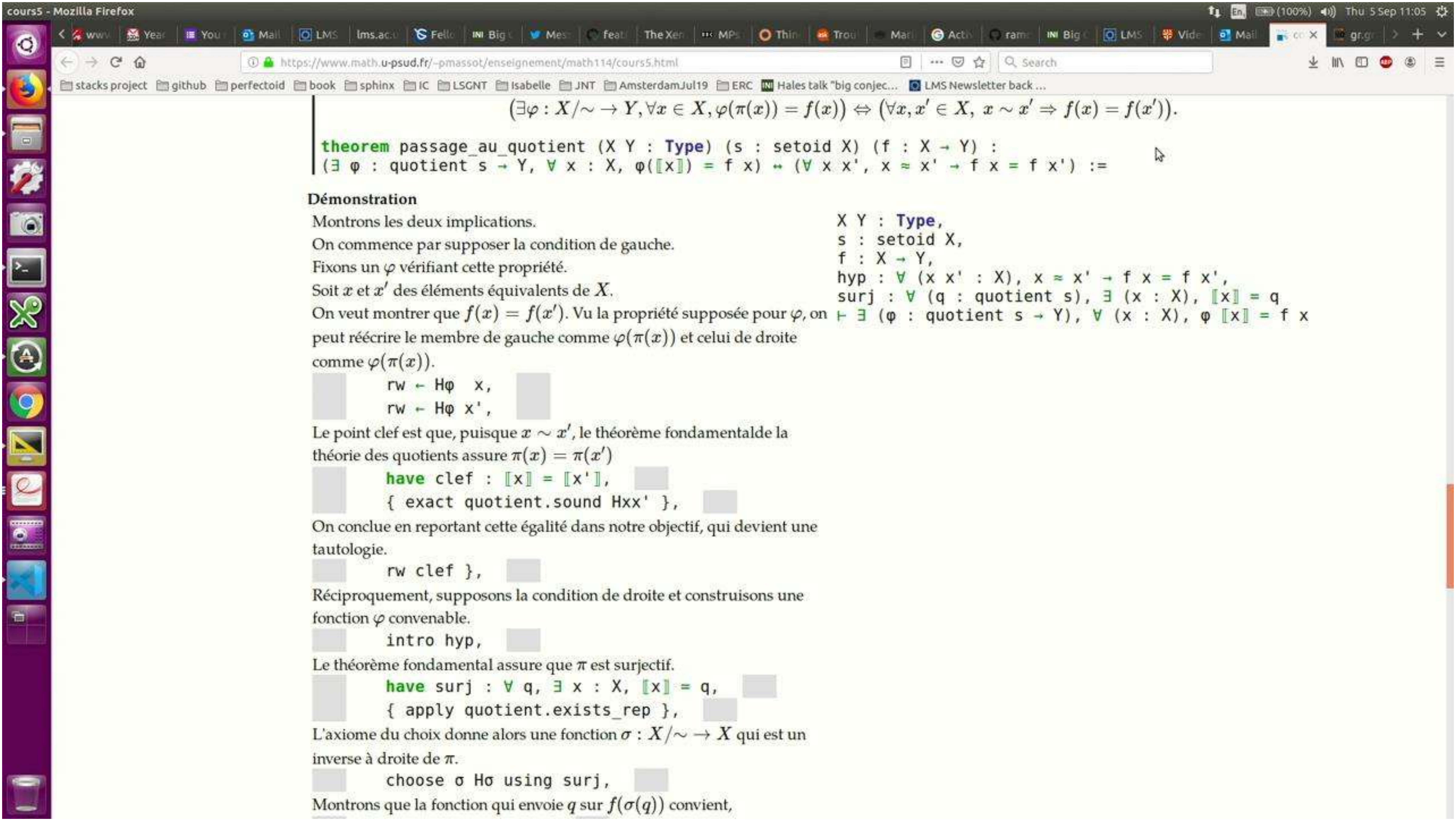
In October 2019 all of the homework in my course will be in Lean format, and all of the course notes too.

See Computers and Mathematics, London Mathematical Society newsletter, September 2019 (pages 32 to 36) for more details of my work with undergraduates.

The main frustrations for undergraduates were *pragmatic* rather than foundational. For example, lack of appropriate documentation (I will fix this).
Genuine dumb problem: it's hard for a maths UG to install Lean and its maths library on Windows, because they need to use the command line. We can use CoCalc instead – it costs money, but I have run some successful webinars on it.

Conclusions: it is possible to teach undergraduate mathematicians how to do some of their homework in Lean. Forthcoming paper by Iannone and Thoma will say something more formal about what I have achieved so far.

In October 2019 all of the homework in my course will be in Lean format, and all of the course notes too. Like this.



$$\left(\exists\varphi : X/\sim \to Y, \forall x \in X, \varphi(\pi(x)) = f(x)\right) \Leftrightarrow \left(\forall x, x' \in X,\ x \sim x' \Rightarrow f(x) = f(x')\right).$$

```
theorem passage_au_quotient (X Y : Type) (s : setoid X) (f : X → Y) :
(∃ φ : quotient s → Y, ∀ x : X, φ(⟦x⟧) = f x) ↔ (∀ x x', x ≈ x' → f x = f x') :=
```

Démonstration

Montrons les deux implications.

On commence par supposer la condition de gauche.

Fixons un φ vérifiant cette propriété.

Soit x et x' des éléments équivalents de X.

On veut montrer que $f(x) = f(x')$. Vu la propriété supposée pour φ, on peut réécrire le membre de gauche comme $\varphi(\pi(x))$ et celui de droite comme $\varphi(\pi(x))$.

```
X Y : Type,
s : setoid X,
f : X → Y,
hyp : ∀ (x x' : X), x ≈ x' → f x = f x',
surj : ∀ (q : quotient s), ∃ (x : X), ⟦x⟧ = q
⊢ ∃ (φ : quotient s → Y), ∀ (x : X), φ ⟦x⟧ = f x
```

```
    rw ← Hφ  x,
    rw ← Hφ x',
```

Le point clef est que, puisque $x \sim x'$, le théorème fondamentalde la théorie des quotients assure $\pi(x) = \pi(x')$

```
    have clef : ⟦x⟧ = ⟦x'⟧,
    { exact quotient.sound Hxx' },
```

On conclue en reportant cette égalité dans notre objectif, qui devient une tautologie.

```
    rw clef },
```

Réciproquement, supposons la condition de droite et construisons une fonction φ convenable.

```
    intro hyp,
```

Le théorème fondamental assure que π est surjectif.

```
    have surj : ∀ q, ∃ x : X, ⟦x⟧ = q,
    { apply quotient.exists_rep },
```

L'axiome du choix donne alors une fonction $\sigma : X/\sim \to X$ qui est un inverse à droite de π.

```
    choose σ Hσ using surj,
```

Montrons que la fonction qui envoie q sur $f(\sigma(q))$ convient,

See Computers and Mathematics, London Mathematical Society newsletter, September 2019 (pages 32 to 36) for more details of my work with undergraduates.

The main frustrations for undergraduates were *pragmatic* rather than foundational. For example, lack of appropriate documentation (I will fix this).
Genuine dumb problem: it's hard for a maths UG to install Lean and its maths library on Windows, because they need to use the command line. We can use CoCalc instead – it costs money, but I have run some successful webinars on it.

Conclusions: it is possible to teach undergraduate mathematicians how to do some of their homework in Lean. Forthcoming paper by Iannone and Thoma will say something more formal about what I have achieved so far.

In October 2019 all of the homework in my course will be in Lean format, and all of the course notes too. Like this. My job is to teach the undergraduates what a proof is. Lean is a wonderful tool for this.

See Computers and Mathematics, London Mathematical Society newsletter, September 2019 (pages 32 to 36) for more details of my work with undergraduates.

The main frustrations for undergraduates were *pragmatic* rather than foundational. For example, lack of appropriate documentation (I will fix this).
Genuine dumb problem: it's hard for a maths UG to install Lean and its maths library on Windows, because they need to use the command line. We can use CoCalc instead – it costs money, but I have run some successful webinars on it.

Conclusions: it is possible to teach undergraduate mathematicians how to do some of their homework in Lean. Forthcoming paper by Iannone and Thoma will say something more formal about what I have achieved so far.

In October 2019 all of the homework in my course will be in Lean format, and all of the course notes too. Like this. My job is to teach the undergraduates what a proof is. Lean is a wonderful tool for this. My goal here is to digitise our curriculum in Lean.

Example of what I have learnt myself from using Lean:
First part of first question on first problem sheet of my course:
"True or false – if x is a real number, and $x^2 - 3x + 2 = 0$, then $x = 1$."

Example of what I have learnt myself from using Lean:
First part of first question on first problem sheet of my course:
“True or false – if x is a real number, and $x^2 - 3x + 2 = 0$, then $x = 1$.”

My answer “False – set $x = 2$.

Example of what I have learnt myself from using Lean:
First part of first question on first problem sheet of my course:
“True or false – if x is a real number, and $x^2 - 3x + 2 = 0$, then $x = 1$.”

My answer “False – set $x = 2$.

Lean: “OK, so it now suffices to prove that (a) $2^2 - 3 \times 2 + 2 = 0$ and that (b) $2 \neq 1$.

Example of what I have learnt myself from using Lean:
First part of first question on first problem sheet of my course:
“True or false – if x is a real number, and $x^2 - 3x + 2 = 0$, then $x = 1$.”

My answer “False – set $x = 2$.

Lean: “OK, so it now suffices to prove that (a) $2^2 - 3 \times 2 + 2 = 0$ and that (b) $2 \neq 1$.

Me in 2017: “. . . ”

A few weeks later, this was fixed by computer scientists, who wrote a tactic which solved these goals.

Despite successes and popularity with students, my “proper mathematician” colleagues at Imperial are less interested. “Can the software tell us anything *new?*”

Despite successes and popularity with students, my “proper mathematician” colleagues at Imperial are less interested. “Can the software tell us anything *new?*” Not yet.

Despite successes and popularity with students, my “proper mathematician” colleagues at Imperial are less interested. “Can the software tell us anything *new?*” Not yet.

- There’s a long and really low-level formal proof of the four colour theorem in Coq.

Despite successes and popularity with students, my "proper mathematician" colleagues at Imperial are less interested. "Can the software tell us anything *new?*" Not yet.

- There's a long and really low-level formal proof of the four colour theorem in Coq.
- There's a very long and quite low-level formal proof of the odd order theorem in Coq.

Despite successes and popularity with students, my “proper mathematician” colleagues at Imperial are less interested. “Can the software tell us anything *new?*” Not yet.

- There’s a long and really low-level formal proof of the four colour theorem in Coq.
- There’s a very long and quite low-level formal proof of the odd order theorem in Coq.
- Lean is like Coq but better. So it will be able to handle these things.

Despite successes and popularity with students, my "proper mathematician" colleagues at Imperial are less interested. "Can the software tell us anything *new?*" Not yet.

- There's a long and really low-level formal proof of the four colour theorem in Coq.
- There's a very long and quite low-level formal proof of the odd order theorem in Coq.
- Lean is like Coq but better. So it will be able to handle these things.
- So why is no "proper mathematician" interested?

Despite successes and popularity with students, my “proper mathematician” colleagues at Imperial are less interested. “Can the software tell us anything *new?*” Not yet.

- There’s a long and really low-level formal proof of the four colour theorem in Coq.
- There’s a very long and quite low-level formal proof of the odd order theorem in Coq.
- Lean is like Coq but better. So it will be able to handle these things.
- So why is no “proper mathematician” interested?
- Two reasons!
    1. [Aesthetics / fashion] Because “proper mathematicians” like me don’t care about these results – we like high-level proofs about modern objects.

Despite successes and popularity with students, my “proper mathematician” colleagues at Imperial are less interested. “Can the software tell us anything *new?*” Not yet.

- There’s a long and really low-level formal proof of the four colour theorem in Coq.
- There’s a very long and quite low-level formal proof of the odd order theorem in Coq.
- Lean is like Coq but better. So it will be able to handle these things.
- So why is no “proper mathematician” interested?
- Two reasons!
  1. [Aesthetics / fashion] Because “proper mathematicians” like me don’t care about these results – we like high-level proofs about modern objects.
  2. [Belief system] We don’t need formal proof anyway – we have a system of elders, which has worked for centuries.

Despite successes and popularity with students, my “proper mathematician” colleagues at Imperial are less interested. “Can the software tell us anything *new?*” Not yet.

- There’s a long and really low-level formal proof of the four colour theorem in Coq.
- There’s a very long and quite low-level formal proof of the odd order theorem in Coq.
- Lean is like Coq but better. So it will be able to handle these things.
- So why is no “proper mathematician” interested?
- Two reasons!
  ① [Aesthetics / fashion] Because “proper mathematicians” like me don’t care about these results – we like high-level proofs about modern objects.
  ② [Belief system] We don’t need formal proof anyway – we have a system of elders, which has worked for centuries.
- I think there is a non-zero chance that some of our great castles are built on sand.

Despite successes and popularity with students, my “proper mathematician” colleagues at Imperial are less interested. “Can the software tell us anything *new?*” Not yet.

- There’s a long and really low-level formal proof of the four colour theorem in Coq.
- There’s a very long and quite low-level formal proof of the odd order theorem in Coq.
- Lean is like Coq but better. So it will be able to handle these things.
- So why is no “proper mathematician” interested?
- Two reasons!
  - ① [Aesthetics / fashion] Because “proper mathematicians” like me don’t care about these results – we like high-level proofs about modern objects.
  - ② [Belief system] We don’t need formal proof anyway – we have a system of elders, which has worked for centuries.
- I think there is a non-zero chance that some of our great castles are built on sand. But I think it’s small.

The statement of Fermat's Last Theorem can be explained to a high school kid.

The statement of Fermat's Last Theorem can be explained to a high school kid. What does the proof of Fermat's Last Theorem look like?

- First you invent elliptic curves.

The statement of Fermat's Last Theorem can be explained to a high school kid. What does the proof of Fermat's Last Theorem look like?

- First you invent elliptic curves.
- Then you invent modular forms.

The statement of Fermat's Last Theorem can be explained to a high school kid. What does the proof of Fermat's Last Theorem look like?

- First you invent elliptic curves.
- Then you invent modular forms.
- Then you invent finite flat group schemes, automorphic representations, *p*-adic Galois representations, Hecke algebras, universal deformation rings, Galois cohomology, local and global class field theory, harmonic analysis, algebraic geometry, arithmetic geometry, nonabelian Fourier theory.

The statement of Fermat’s Last Theorem can be explained to a high school kid. What does the proof of Fermat’s Last Theorem look like?

- First you invent elliptic curves.
- Then you invent modular forms.
- Then you invent finite flat group schemes, automorphic representations, p-adic Galois representations, Hecke algebras, universal deformation rings, Galois cohomology, local and global class field theory, harmonic analysis, algebraic geometry, arithmetic geometry, nonabelian Fourier theory. This took us about 350 years.

The statement of Fermat's Last Theorem can be explained to a high school kid. What does the proof of Fermat's Last Theorem look like?

- First you invent elliptic curves.
- Then you invent modular forms.
- Then you invent finite flat group schemes, automorphic representations, p-adic Galois representations, Hecke

The statement of Fermat's Last Theorem can be explained to a high school kid. What does the proof of Fermat's Last Theorem look like?

- First you invent elliptic curves.
- Then you invent modular forms.
- Then you invent finite flat group schemes, automorphic representations, p-adic Galois representations, Hecke algebras, universal deformation rings, Galois cohomology, local and global class field theory, harmonic analysis, algebraic geometry, arithmetic geometry, nonabelian Fourier theory. This took us about 350 years. Note that these are *not undergraduate or MSc level things*. So undergraduates are of limited use here.
- Then you prove some really profound theorems about some of these objects, using the rest of these objects.

The statement of Fermat's Last Theorem can be explained to a high school kid. What does the proof of Fermat's Last Theorem look like?

- First you invent elliptic curves.
- Then you invent modular forms.
- Then you invent finite flat group schemes, automorphic representations, p-adic Galois representations, Hecke algebras, universal deformation rings, Galois cohomology, local and global class field theory, harmonic analysis, algebraic geometry, arithmetic geometry, nonabelian Fourier theory. This took us about 350 years. Note that these are *not undergraduate or MSc level things.* So undergraduates are of limited use here.
- Then you prove some really profound theorems about some of these objects, using the rest of these objects.
- And then Fermat's Last Theorem comes out in the wash.

The statement of Fermat's Last Theorem can be explained to a high school kid. What does the proof of Fermat's Last Theorem look like?

- First you invent elliptic curves.
- Then you invent modular forms.
- Then you invent finite flat group schemes, automorphic representations, p-adic Galois representations, Hecke algebras, universal deformation rings, Galois cohomology, local and global class field theory, harmonic analysis, algebraic geometry, arithmetic geometry, nonabelian Fourier theory. This took us about 350 years. Note that these are *not undergraduate or MSc level things.* So undergraduates are of limited use here.
- Then you prove some really profound theorems about some of these objects, using the rest of these objects.
- And then Fermat's Last Theorem comes out in the wash.
- The full proof takes thousands of pages.

Give me 100 million dollars and 10 years and I believe I could get a team together to formalise a proof of Fermat's Last Theorem.

Give me 100 million dollars and 10 years and I believe I could get a team together to formalise a proof of Fermat's Last Theorem. No mathematician I have met disputes this.

Give me 100 million dollars and 10 years and I believe I could get a team together to formalise a proof of Fermat's Last Theorem. No mathematician I have met disputes this. Currently prohibitively expensive.

Give me 100 million dollars and 10 years and I believe I could get a team together to formalise a proof of Fermat's Last Theorem. No mathematician I have met disputes this. Currently prohibitively expensive.

But what is worse, *no proper mathematician would care.*

Give me 100 million dollars and 10 years and I believe I could get a team together to formalise a proof of Fermat's Last Theorem. No mathematician I have met disputes this. Currently prohibitively expensive.

But what is worse, *no proper mathematician would care.*

The elders have decreed that the proof is OK.

Give me 100 million dollars and 10 years and I believe I could get a team together to formalise a proof of Fermat's Last Theorem. No mathematician I have met disputes this. Currently prohibitively expensive.

But what is worse, *no proper mathematician would care.*

The elders have decreed that the proof is OK.

I believe that no human, alive or dead, knows all the details of the proof of Fermat's Last Theorem.

Give me 100 million dollars and 10 years and I believe I could get a team together to formalise a proof of Fermat's Last Theorem. No mathematician I have met disputes this. Currently prohibitively expensive.

But what is worse, *no proper mathematician would care.*

The elders have decreed that the proof is OK.

I believe that no human, alive or dead, knows all the details of the proof of Fermat's Last Theorem. But the community accept the proof nonetheless, because the proof is modular.

Give me 100 million dollars and 10 years and I believe I could get a team together to formalise a proof of Fermat's Last Theorem. No mathematician I have met disputes this. Currently prohibitively expensive.

But what is worse, *no proper mathematician would care.*

The elders have decreed that the proof is OK.

I believe that no human, alive or dead, knows all the details of the proof of Fermat's Last Theorem. But the community accept the proof nonetheless, because the proof is modular.

Our community even accepts proofs if the author says "There are now 100 missing pages, which we will get to later on."

Give me 100 million dollars and 10 years and I believe I could get a team together to formalise a proof of Fermat's Last Theorem. No mathematician I have met disputes this. Currently prohibitively expensive.

But what is worse, *no proper mathematician would care.*

The elders have decreed that the proof is OK.

I believe that no human, alive or dead, knows all the details of the proof of Fermat's Last Theorem. But the community accept the proof nonetheless, because the proof is modular.

Our community even accepts proofs if the author says "There are now 100 missing pages, which we will get to later on."

We accepted the proof of the odd order theorem in 1970 – that's why we gave John Thompson a Fields Medal.

Give me 100 million dollars and 10 years and I believe I could get a team together to formalise a proof of Fermat's Last Theorem. No mathematician I have met disputes this. Currently prohibitively expensive.

But what is worse, *no proper mathematician would care.*

The elders have decreed that the proof is OK.

I believe that no human, alive or dead, knows all the details of the proof of Fermat's Last Theorem. But the community accept the proof nonetheless, because the proof is modular.

Our community even accepts proofs if the author says "There are now 100 missing pages, which we will get to later on."

We accepted the proof of the odd order theorem in 1970 – that's why we gave John Thompson a Fields Medal. We don't care that it got formalised – it was already "checked".

So if proper mathematicians aren't interested in a proof of the odd order theorem, what are they interested in?

So if proper mathematicians aren't interested in a proof of the odd order theorem, what are they interested in?

Example: Perfectoid spaces.

So if proper mathematicians aren't interested in a proof of the odd order theorem, what are they interested in?

Example: Perfectoid spaces.

	Proof of odd order theorem	Perfectoid spaces
Got author a Fields Medal?	Yes (1970)	Yes (2018)
High level mathematics?	No	Yes
Lots of PhD students and post-docs working in the area?	No	Yes
Talks happening about these things all over the world?	No	Yes
Mathematicans interested in 2019?	No	Yes

Earlier this year, Patrick Massot, Johan Commelin and myself formalised the definition of a perfectoid space in Lean.

Earlier this year, Patrick Massot, Johan Commelin and myself formalised the definition of a perfectoid space in Lean.

I am getting invitations from across the EU to speak in mathematics departments about the work.

Earlier this year, Patrick Massot, Johan Commelin and myself formalised the definition of a perfectoid space in Lean.

I am getting invitations from across the EU to speak in mathematics departments about the work. Serious piece of research, or elaborate PR stunt? Maybe both.

Earlier this year, Patrick Massot, Johan Commelin and myself formalised the definition of a perfectoid space in Lean.

I am getting invitations from across the EU to speak in mathematics departments about the work. Serious piece of research, or elaborate PR stunt? Maybe both. It doesn't tell us any new theorems, but it does prove that *Lean can understand the question*.

Earlier this year, Patrick Massot, Johan Commelin and myself formalised the definition of a perfectoid space in Lean.

I am getting invitations from across the EU to speak in mathematics departments about the work. Serious piece of research, or elaborate PR stunt? Maybe both. It doesn't tell us any new theorems, but it does prove that *Lean can understand the question*.

Mathematical aside: why is formalising a definition hard work?

Earlier this year, Patrick Massot, Johan Commelin and myself formalised the definition of a perfectoid space in Lean.

I am getting invitations from across the EU to speak in mathematics departments about the work. Serious piece of research, or elaborate PR stunt? Maybe both. It doesn't tell us any new theorems, but it does prove that *Lean can understand the question*.

Mathematical aside: why is formalising a definition hard work?

A real manifold is a topological space which locally looks like a ball. For this to typecheck we need to know that a ball is a topological space. This is not difficult.

A perfectoid space is a locally ringed space which locally looks like an affinoid perfectoid space. For this to typecheck we need to show that affinoid perfectoid spaces are locally ringed spaces (or actually something slightly weaker).

As I've said, my next step is to get more research mathematicians using the software.

As I've said, my next step is to get more research mathematicians using the software.

Why? To make some powerful high-level tools which future mathematicians will use, we need to teach Lean hundreds, or maybe thousands, of high-level mathematical definitions.

As I've said, my next step is to get more research mathematicians using the software.

Why? To make some powerful high-level tools which future mathematicians will use, we need to teach Lean hundreds, or maybe thousands, of high-level mathematical definitions. Advances in comprehension of natural language will not do this for us.

As I've said, my next step is to get more research mathematicians using the software.

Why? To make some powerful high-level tools which future mathematicians will use, we need to teach Lean hundreds, or maybe thousands, of high-level mathematical definitions. Advances in comprehension of natural language will not do this for us. This has "synergy" written all over it.

As I've said, my next step is to get more research mathematicians using the software.

Why? To make some powerful high-level tools which future mathematicians will use, we need to teach Lean hundreds, or maybe thousands, of high-level mathematical definitions. Advances in comprehension of natural language will not do this for us. This has "synergy" written all over it.

The Coq theorem prover was written in 1989. Thirty years later, a modern mathematician will find that there is still a very high chance that they cannot formalise the *statements* of what they are working on in any of the available theorem provers.

As I've said, my next step is to get more research mathematicians using the software.

Why? To make some powerful high-level tools which future mathematicians will use, we need to teach Lean hundreds, or maybe thousands, of high-level mathematical definitions. Advances in comprehension of natural language will not do this for us. This has “synergy” written all over it.

The Coq theorem prover was written in 1989. Thirty years later, a modern mathematician will find that there is still a very high chance that they cannot formalise the *statements* of what they are working on in any of the available theorem provers.

We mathematicians don't see the modern complex mathematical objects which we use every day, in theorem provers.

As I've said, my next step is to get more research mathematicians using the software.

Why? To make some powerful high-level tools which future mathematicians will use, we need to teach Lean hundreds, or maybe thousands, of high-level mathematical definitions. Advances in comprehension of natural language will not do this for us. This has "synergy" written all over it.

The Coq theorem prover was written in 1989. Thirty years later, a modern mathematician will find that there is still a very high chance that they cannot formalise the *statements* of what they are working on in any of the available theorem provers.

We mathematicians don't see the modern complex mathematical objects which we use every day, in theorem provers. Yet. I just wrote some EU grant proposal to fund post-docs who will write a bunch of Lean code defining the objects which "make a mathematican tick". And then (following Tom Hales) we can start to make a database, or a network, mapping out the state of the beliefs of the elders.

Conclusions:

- Lean's type theory seems to be perfect for modern pure mathematics.

Conclusions:

- Lean's type theory seems to be perfect for modern pure mathematics.
- Crucial next step: put some modern pure mathematics into it.
- Need professional mathematicians, trained to use the software, to do this.

Conclusions:

- Lean's type theory seems to be perfect for modern pure mathematics.
- Crucial next step: put some modern pure mathematics into it.
- Need professional mathematicians, trained to use the software, to do this.
- Only then we can take the first steps towards Tom Hales' idea of a formalised database of definitions and theorem statements.

Conclusions:

- Lean's type theory seems to be perfect for modern pure mathematics.
- Crucial next step: put some modern pure mathematics into it.
- Need professional mathematicians, trained to use the software, to do this.
- Only then we can take the first steps towards Tom Hales' idea of a formalised database of definitions and theorem statements. A new kind of database.

Conclusions:

- Lean's type theory seems to be perfect for modern pure mathematics.
- Crucial next step: put some modern pure mathematics into it.
- Need professional mathematicians, trained to use the software, to do this.

Conclusions:

- Lean's type theory seems to be perfect for modern pure mathematics.
- Crucial next step: put some modern pure mathematics into it.
- Need professional mathematicians, trained to use the software, to do this.
- Only then we can take the first steps towards Tom Hales' idea of a formalised database of definitions and theorem statements. A new kind of database.
- And then we're looking at: (1) search for mathematicians, (2) tools to help mathematicians do computations,

Conclusions:

- Lean's type theory seems to be perfect for modern pure mathematics.
- Crucial next step: put some modern pure mathematics into it.
- Need professional mathematicians, trained to use the software, to do this.
- Only then we can take the first steps towards Tom Hales' idea of a formalised database of definitions and theorem statements. A new kind of database.
- And then we're looking at: (1) search for mathematicians, (2) tools to help mathematicians do computations, (3) automatic marking and instant feedback,

Conclusions:

- Lean's type theory seems to be perfect for modern pure mathematics.
- Crucial next step: put some modern pure mathematics into it.
- Need professional mathematicians, trained to use the software, to do this.
- Only then we can take the first steps towards Tom Hales' idea of a formalised database of definitions and theorem statements. A new kind of database.
- And then we're looking at: (1) search for mathematicians, (2) tools to help mathematicians do computations, (3) automatic marking and instant feedback, (4) world domination by computer AI and I can retire early.

Conclusions:

- Lean's type theory seems to be perfect for modern pure mathematics.
- Crucial next step: put some modern pure mathematics into it.
- Need professional mathematicians, trained to use the software, to do this.
- Only then we can take the first steps towards Tom Hales' idea of a formalised database of definitions and theorem statements. A new kind of database.
- And then we're looking at: (1) search for mathematicians, (2) tools to help mathematicians do computations, (3) automatic marking and instant feedback, (4) world domination by computer AI and I can retire early.

Thanks for coming!