Formalized number theory

Past, present, and future

Alex J. Best (VU)

3/6/22 – Intercity Seminar

In this talk

- Past – some results and areas that people have thought about in the past
- Present – some current works in progress
- Future – where we are going, what needs to be different to make progress

In number theory particularly, but also other areas of interest.

In the 2000s

Large projects such as the Kepler conjecture and the Odd order theorem.

These were big collaborations with one main goal, and did involve some number theory adjacent topics.

Now there is more of a trend to build on existing libraries to make more progress on deeper topics.

Perfectoids

Peter Scholze won a Fields medal in 2018 for “transforming arithmetic algebraic geometry over p-adic fields through his introduction of perfectoid spaces, with application to Galois representations, and for the development of new cohomology theories.”

Perfectoids

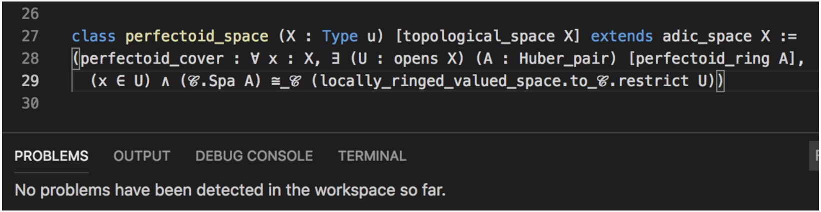
Peter Scholze won a Fields medal in 2018 for "transforming arithmetic algebraic geometry over p-adic fields through his introduction of perfectoid spaces, with application to Galois representations, and for the development of new cohomology theories." The definition is highly nontrivial, an unusual geometric object created from an extremely non-Noetherian ring.

Perfectoids

Peter Scholze won a Fields medal in 2018 for “transforming arithmetic algebraic geometry over p-adic fields through his introduction of perfectoid spaces, with application to Galois representations, and for the development of new cohomology theories.” The definition is highly nontrivial, an unusual geometric object created from an extremely non-Noetherian ring.

In 2020ish Kevin Buzzard, Johan Commelin, Patrick Massot (building on others) completed a long term project to define a perfectoid space formally in Lean.

Perfectoids

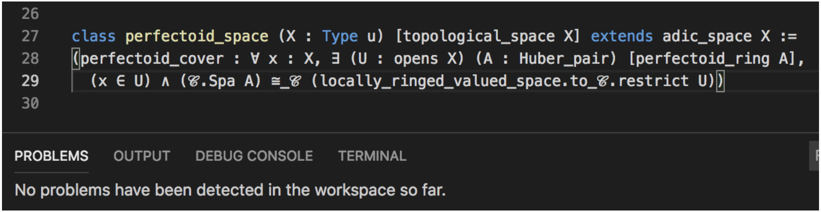


```
class perfectoid_space (X : Type u) [topological_space X] extends adic_space X :=
(perfectoid_cover : ∀ x : X, ∃ (U : opens X) (A : Huber_pair) [perfectoid_ring A],
  (x ∈ U) ∧ (𝒞.Spa A) ≅_𝒞 (locally_ringed_valued_space.to_𝒞.restrict U))
```

No problems have been detected in the workspace so far.

Lean has accepted the chain of definitions that lead to this are all valid, topological spaces, sheaves, valuations, adic spaces, perfectoid rings,...

Perfectoids



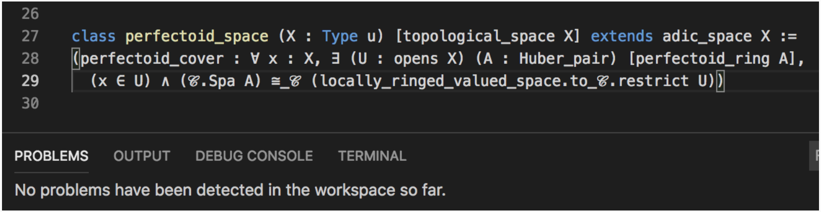
```
class perfectoid_space (X : Type u) [topological_space X] extends adic_space X :=
(perfectoid_cover : ∀ x : X, ∃ (U : opens X) (A : Huber_pair) [perfectoid_ring A],
  (x ∈ U) ∧ (𝒞.Spa A) ≅_𝒞 (locally_ringed_valued_space.to_𝒞.restrict U))
```

No problems have been detected in the workspace so far.

Lean has accepted the chain of definitions that lead to this are all valid, topological spaces, sheaves, valuations, adic spaces, perfectoid rings,...

It is difficult to estimate the amount of human effort expended to achieve this.

Perfectoids



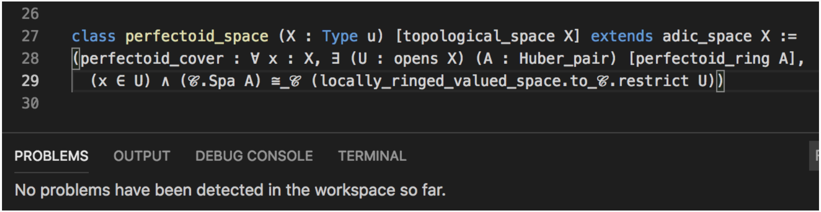
```
class perfectoid_space (X : Type u) [topological_space X] extends adic_space X :=
(perfectoid_cover : ∀ x : X, ∃ (U : opens X) (A : Huber_pair) [perfectoid_ring A],
  (x ∈ U) ∧ (𝒞.Spa A) ≅_𝒞 (locally_ringed_valued_space.to_𝒞.restrict U))
```

No problems have been detected in the workspace so far.

Lean has accepted the chain of definitions that lead to this are all valid, topological spaces, sheaves, valuations, adic spaces, perfectoid rings,...

It is difficult to estimate the amount of human effort expended to achieve this. Their work relied on that of many others who are building `mathlib`, a general purpose library of mathematics from the ground up.

Perfectoids



```
class perfectoid_space (X : Type u) [topological_space X] extends adic_space X :=
(perfectoid_cover : ∀ x : X, ∃ (U : opens X) (A : Huber_pair) [perfectoid_ring A],
  (x ∈ U) ∧ (𝒞.Spa A) ≅_𝒞 (locally_ringed_valued_space.to_𝒞.restrict U))
```

No problems have been detected in the workspace so far.

Lean has accepted the chain of definitions that lead to this are all valid, topological spaces, sheaves, valuations, adic spaces, perfectoid rings,...

It is difficult to estimate the amount of human effort expended to achieve this. Their work relied on that of many others who are building `mathlib`, a general purpose library of mathematics from the ground up.

However, it also takes a long time for a human with no mathematical background to learn such a definition.

One side effect: ”new” algebraic structures

One ingredient of the theory surrounding perfectoid spaces (adic, spectral, Huber rings, etc.) is the notion of a valuation

$$K \to \Gamma \cup \{0\}$$

sending 0 to 0.

In the course of the project the authors noticed they were having to repeat a lot of work on basic lemmas that were true both for fields and the value group above, inspired the creation of a new definition, a *group with zero* (and monoid with zero, etc.).

> *“Every sufficiently good analogy is yearning to become a functor.” – John Baez*
>
> *Every sufficiently similar proof is yearning to become a new algebraic structure.*

Niche algebraic structures

There is even a lot of duplication between lemmas about groups, and those about groups with a zero.

Earlier this year Yaël Dillies introduced a new algebraic structure, a division monoid, to be the correct setting for theorems, this is a monoid with an involutive inverse operation that doesn't always have $a \cdot a^{-1} = 1$, but does have $a \cdot b = 1$ implies $a^{-1} = b$.

Upshot: In order to formalize effectively and reduce duplication of effort generalizing proofs to unfamiliar algebraic structures is helpful.

Generalizing theorems automatically

When we have so many algebraic structures, we don’t want to spend our time trying to find the right structure to prove theorems in.

Would prefer to write the proof under some assumptions we know work, and then let the proof *assistant* tell us the most general and widely useful assumptions.

Generalizing theorems automatically

When we have so many algebraic structures, we don't want to spend our time trying to find the right structure to prove theorems in.

Would prefer to write the proof under some assumptions we know work, and then let the proof *assistant* tell us the most general and widely useful assumptions.

Lemma

Let $f : K \to L$ be a ring homomorphism between two fields, and p be a natural number, then K is characteristic p if and only if L is (including $p = 0$).

Generalizing theorems automatically

When we have so many algebraic structures, we don't want to spend our time trying to find the right structure to prove theorems in.

Would prefer to write the proof under some assumptions we know work, and then let the proof *assistant* tell us the most general and widely useful assumptions.

Lemma

Let $f : K \to L$ be a ring homomorphism between two fields, and p be a natural number, then K is characteristic p if and only if L is (including $p = 0$).

A Lean based tool I wrote last year will happily inform us that K can be any division ring and L can in fact be any nontrivial semiring. This works just by inspecting the original formalized proof

Backing up

Despite there being impressive progress on very advanced number theory, at least in the mathlib library there was not even the definition of a number field in Lean at the time

Baanen, Dahmen, Ashvni Narayanan, Filippo Nuccio added Dedekind domains, and proved finiteness of the class group last year.

Interestingly this formalization is uniform in the number field and function field cases, and avoids Minkowski's theorem in favour of simpler pigeonhole-type principles.

But the basics of algebraic number theory are not really complete (Kummer-Dedekind, Kummer theory, Kronecker-Weber) in any formal system that I know.

FLT-regular

In attempt to fill the gaps and add more down-to-earth algebraic number theory we have started a project to formalize Kummer's proof of Fermat's last theorem for regular primes

$$p \nmid h_{\mathbf{Q}(\zeta_p)}$$

this splits into two cases for

$$x^p + y^p = z^p$$

Case I: $p \nmid xyz$ (comparatively elementary, lots of progress, computing basics about cyclotomic fields, ...)

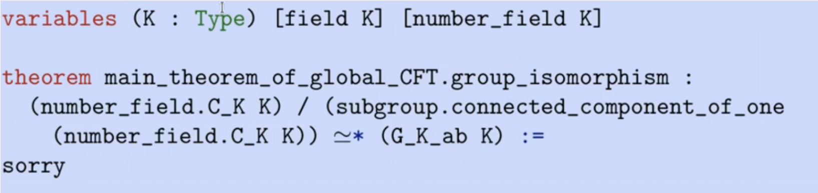
Case II: $p \mid xyz$ (requires some class field theory in an essential way, even on paper the proofs are long)

Some progress

María Inés de Frutos Fernández has formalized the ring of Adèles (and Idèles) and given the *statement* of the main theorem of global CFT in Lean:

Theorem

Let K be a number field. Denote by C_K^1 the quotient of C_K by the connected component of the identity. There is an isomorphism of topological groups $C_K^1 \simeq G_K^{ab}$.



```
variables (K : Type) [field K] [number_field K]

theorem main_theorem_of_global_CFT.group_isomorphism :
  (number_field.C_K K) / (subgroup.connected_component_of_one
    (number_field.C_K K)) ≃* (G_K_ab K) :=
sorry
```

Descent

With Anne Baanen, Nirvana Coppola, Sander Dahmen, we have been formalizing some Mordell-style descent to find integral points on elliptic curves: for example the non-existence of integral points on

$$y^2 = x^3 - 5$$

Descent

With Anne Baanen, Nirvana Coppola, Sander Dahmen, we have been formalizing some Mordell-style descent to find integral points on elliptic curves: for example the non-existence of integral points on

$$y^2 = x^3 - 5$$

Basically works, except, we still need to compute the class group of $\mathbf{Q}(\sqrt{-5})$!

This sort of proof necessarily involves some amount of hands on calculation, this is often harder to formalize than clean theory.

In order to work conveniently with such calculations we have added tactics to handle calculations in rings with a finite "multiplication table" automatically, and write formal proofs that aren't significantly longer than paper ones.

Certifying number theoretic computations

Eventually would be helpful to have code that computes class groups implemented in a formal system.

Right now this is a lot of work repeating the excellent pre-existing algorithms in a new language.

Certifying number theoretic computations

Eventually would be helpful to have code that computes class groups implemented in a formal system.

Right now this is a lot of work repeating the excellent pre-existing algorithms in a new language.

Question: Is it possible to compute the class group with a computer algebra system (e.g. Sage), and write down a certificate of the result that is easily checkable (fast to check, not too long, and mathematically simple!)

Ideally the certificate would be a text file, other users shouldn't need to install the CAS to repeat the calculation, but it should be provable in the system.

But the certification itself should not rely on GRH etc.

The Hasse Norm theorem

Suppose we want to check that an explicitly given ideal in a number field is non-principal, can we give a certificate for this.

One idea: If an ideal is principal, it's norm must be equal to the norm of an element (and this holds everywhere locally too).

Theorem (Hasse Norm theorem)

If $K/\mathbf{Q}$ is a cyclic Galois extension and $x \in \mathbf{Q}$ is everywhere locally a norm, then x is globally a norm.

The Hasse Norm theorem

Suppose we want to check that an explicitly given ideal in a number field is non-principal, can we give a certificate for this.

One idea: If an ideal is principal, it's norm must be equal to the norm of an element (and this holds everywhere locally too).

Theorem (Hasse Norm theorem)

If $K/\mathbf{Q}$ is a cyclic Galois extension and $x \in \mathbf{Q}$ is everywhere locally a norm, then x is globally a norm.

There are counterexamples to this in the biquadratic case due to Hasse (and Serre-Tate) (and for any non-cyclic case Frei, Loughran, Newton).

Number fields for which this property holds are said to satisfy the *Hasse norm principle*.

The Hasse Norm theorem

Theorem (Frei, Loughran, Newton)

Let k be a number field and G a finite abelian group. Then 100% of G-extensions of k, ordered by conductor, satisfy the Hasse norm principle.

The Hasse Norm theorem

Theorem (Frei, Loughran, Newton)

Let k be a number field and G a finite abelian group. Then 100% of G-extensions of k, ordered by conductor, satisfy the Hasse norm principle.

But if we order by discriminant:

Theorem (Frei, Loughran, Newton)

Let G be a non-trivial finite abelian group and let Q be the smallest prime dividing $|G|$. Assume that G is not isomorphic to a group of the form $\mathbb{Z}/n\mathbb{Z} \oplus (\mathbb{Z}/Q\mathbb{Z})^r$ for any n divisible by Q and $r \geq 0$. Then a positive proportion of G-extensions of k fail the Hasse norm principle, ordered by discriminant.

So locally verifying non-principality might be viable for abelian number fields.

Other ideas

There are many useful algorithms with ”obvious” certificates:

- Ideal membership
- Matrix normal forms (SNF, HNF, LU, RREF)
- Factoring
- Checking solubility modulo primes

Other ideas

There are many useful algorithms with ”obvious” certificates:

- Ideal membership
- Matrix normal forms (SNF, HNF, LU, RREF)
- Factoring
- Checking solubility modulo primes

Have a tool that talks to Sage to certify some of these in Lean already, working on others.

I’d be happy to learn of other instances of this pattern!

This might be independently a nice check for CASes, when further advanced.

Implementing number theoretic algorithms

Alternatively we can implement algorithms within a proof assistant, as efficient functions that give the same output as what we want to compute

- Gives us a guaranteed correct implementation.
- We can experiment with modifying / improving the algorithm, and prove correctness or equality with the original one.
- We can prove properties, or ”run” the algorithm in families, in ways normal code can’t.

After writing the algorithm down, it is only accepted as a genuine mathematical function when it is shown to halt. With some functions this is obvious, but for algorithms that use recursion or unbounded loops, less so!

Tate's algorithm

Sacha Huriot-Tattegrain (+B.+Dahmen) has implemented Tate's algorithm in Lean(4).

- Complete algorithm to compute local invariants of an elliptic curve, including the $c_p(E), \mathrm{ord}_p(\Delta_E), \mathrm{ord}_p(N_E)$
- Works in characteristic 2 and 3.
- Based on Cohen's description of the algorithm, but at times consulting other sources and even the GP source code was necessary to get it right.
- It runs fast!
- Partly generalized to base rings beyond $\mathbf{Z}$.

Without an independent definition of the Kodaira types and conductor exponent we cannot actually check the algorithm does what it says. Nevertheless we could prove certain properties of the algorithm in future, such as invariance under

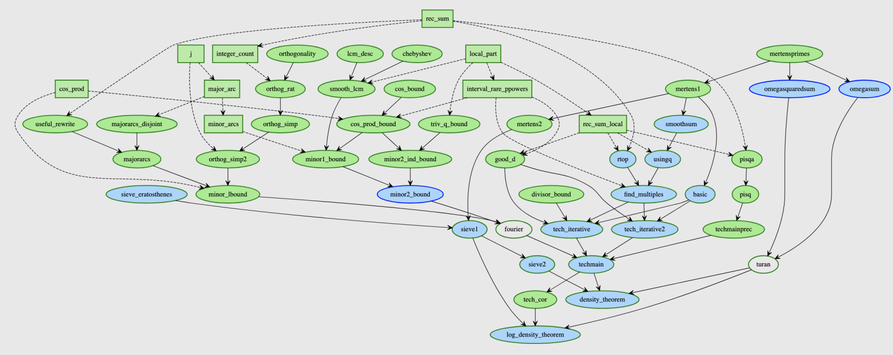
Unit fractions

In December 2021 Thomas Bloom posted a paper: On a Density Conjecture about Unit Fractions to arXiv (2112.03726)

> *Abstract: We prove that any set $A \subset \mathbb{N}$ of positive upper density contains a finite $S \subset A$ such that $\sum_{n \in S} \frac{1}{n} = 1$, answering a question of Erdős and Graham.*

18 pages, quickly recognized as correct and widely applauded in popular press (Quanta, etc), generalizes an older result of Croot.

Thomas Bloom and Bhavik Mehta are working hard to formalize the paper.



Many nice outputs from this project for analytic number theory and density results too.

Collaboration Galore

One nice aspect of formalization is community, we are building on each others work, but the gaps have to line up precisely.

This both eases collaboration (I can not worry about the details of your proof if it compiles and I understand the statement), but it also makes it harder, I have to contend and work with the community agreed upon definition of an object, rather than make my own variant.

Nevertheless working on such a library has the feeling of collaborating on a large textbook / reference work.

Collaboration Galore

- Chris Birkbeck: Defining modular forms + Eisenstein series (like Manuel!)
- David Loeffler: Defining the Gamma function, analytic continuation
- Antoine Chambert-Loir: Finite groups, simplicity of A_n's
- Amelia Livingston: Group cohomology
- Brandon H. Gomes and Alex Kontorovich: statement of the Riemann Hypothesis
- Michael Stoll: re-doing Legendre symbols, proved Hilbert reciprocity for quadratic Hilbert symbols over $\mathbf{Q}$
- Sophie Bernard & Cyril Cohen & Assia Mahboubi & Pierre-Yves Strub, and Thomas Browning: Insolvability of General Higher Degree Equations
- Kevin Wilson: calculation of the density of squarefree numbers as $\zeta(2)^{-1} = 6/\pi^2$.

Closing thoughts

Formalization of mathematics (including number theory) is still slow and painful at times.

But we have several thousand years of mathematics, and learning how to think about, and explain mathematics, to catch up on.

Thinking about these issues and finding clean arguments can be a lot of fun, and the tool may occasionally surprise you.