Formalization and Arithmetic Geometry

past, present, and future

Alex J. Best

9/4/2024

Formalization

Expressing mathematics (objects, arguments) in a format that a computer can handle and interact with rigorously.

Formalization

Expressing mathematics (objects, arguments) in a format that a computer can handle and interact with rigorously.

Some examples of the state of the art:

```
variable {G : Type*} [AddCommGroup G] [MeasurableSpace G]
  [MeasurableSingletonClass G] {A : Set G} [Finite A] {K : ℝ} [Countable G]
  [ElementaryAddCommGroup G 2] [Fintype G]

/-- The polynomial Freiman-Ruzsa (PFR) conjecture: if $A$ is a subset of an elementary abelian
2-group of doubling constant at most $K$, then $A$ can be covered by at most $2K^{12}$ cosets of
a subgroup of cardinality at most $|A|$. -/
theorem PFR_conjecture (h₀A : A.Nonempty) (hA : Nat.card (A + A) ≤ K * Nat.card A) :
    ∃ (H : AddSubgroup G) (c : Set G),
     Nat.card c < 2 * K ^ 12 ∧ Nat.card H ≤ Nat.card A ∧ A ⊆ c + H := by
  obtain ⟨A_pos, -, K_pos⟩ : (0 : ℝ) < Nat.card A ∧ (0 : ℝ) < Nat.card (A + A) ∧ 0 < K :=
    PFR_conjecture_pos_aux' h₀A hA
  -- consider the subgroup `H` given by Lemma `PFR_conjecture_aux`.
  obtain ⟨H, c, hc, IHA, IAH, A_subs_cH⟩ : ∃ (H : AddSubgroup G) (c : Set G),
    Nat.card c ≤ K ^ (13/2) * (Nat.card A) ^ (1/2) * (Nat.card (H : Set G)) ^ (-1/2)
      ∧ Nat.card (H : Set G) ≤ K ^ 11 * Nat.card A ∧ Nat.card A ≤ K ^ 11 * Nat.card (H : Set G)
      ∧ A ⊆ c + H :=
    PFR_conjecture_aux h₀A hA
  have H_pos : (0 : ℝ) < Nat.card (H : Set G) := by
```

Formalization

Expressing mathematics (objects, arguments) in a format that a computer can handle and interact with rigorously.

Some examples of the state of the art:

```
variable {G : Type*} [AddCommGroup G] {A : Set G} [Finite A] {K : ℝ}
  [ElementaryAddCommGroup G 2] [Fintype G]

/-- The polynomial Freiman-Ruzsa (PFR) conjecture: if $A$ is a subset of an elementary abelian
2-group of doubling constant at most $K$, then $A$ can be covered by at most $2K^{12}$ cosets of
a subgroup of cardinality at most $|A|$. -/
theorem PFR_conjecture (h₀A : A.Nonempty) (hA : Nat.card (A + A) ≤ K * Nat.card A) :
    ∃ (H : AddSubgroup G) (c : Set G),
     Nat.card c < 2 * K ^ 12 ∧ Nat.card H ≤ Nat.card A ∧ A ⊆ c + H := by
  obtain ⟨A_pos, -, K_pos⟩ : (0 : ℝ) < Nat.card A ∧ (0 : ℝ) < Nat.card (A + A) ∧ 0 < K :=
    PFR_conjecture_pos_aux' h₀A hA
  -- consider the subgroup `H` given by Lemma `PFR_conjecture_aux`.
  obtain ⟨H, c, hc, IHA, IAH, A_subs_cH⟩ : ∃ (H : AddSubgroup G) (c : Set G),
    Nat.card c ≤ K ^ (13/2) * (Nat.card A) ^ (1/2) * (Nat.card (H : Set G)) ^ (-1/2)
      ∧ Nat.card (H : Set G) ≤ K ^ 11 * Nat.card A ∧ Nat.card A ≤ K ^ 11 * Nat.card (H : Set G)
      ∧ A ⊆ c + H :=
    PFR_conjecture_aux h₀A hA
  have H_pos : (0 : ℝ) < Nat.card (H : Set G) := by
    have : 0 < Nat.card (H : Set G) := Nat.card_pos; positivity
```

Formalization

Expressing mathematics (objects, arguments) in a format that a computer can handle and interact with rigorously.

Some examples of the state of the art:

```
Exercise "Continuity implies sequential continuity"
  Given: (f : ℝ → ℝ) (u : ℕ → ℝ) (x₀ : ℝ)
  Assume: (hu : u converges to x₀) (hf : f is continuous at x₀)
  Conclusion: (f ∘ u) converges to f x₀
Proof:
  Let's prove that ∀ ε > 0, ∃ N, ∀ n ≥ N, |f (u n) - f x₀| ≤ ε
  Fix ε > 0
  By hf applied to ε using that ε > 0 we get δ such that
    (δ_pos : δ > 0) (Hf : ∀ x, |x - x₀| ≤ δ → |f x - f x₀| ≤ ε)
  By hu applied to δ using that δ > 0 we get N such that Hu : ∀ n ≥ N, |u n - x₀| ≤ δ
  Let's prove that N works : ∀ n ≥ N, |f (u n) - f x₀| ≤ ε
  Fix n ≥ N
  By Hf applied to u n it suffices to prove |u n - x₀| ≤ δ
  We conclude by Hu applied to n using n_ge
QED
```

Potential use cases of formalization

- Quickly searching for previously formalized results that may be useful in a given situation
- Automation of routine arguments, letting the software worry about the details
- Producing documents for which we can easily look up precise definitions, or tell halfway through a paper what the current objects being talked about are
- Producing interactive documents where the user can choose the level of detail they want to see, or the shortest path to understand a given result.

Potential use cases of formalization, cont

- Error free or higher confidence in the details of published mathematics
- Can lead to bug free mathematical software, verified plotting
- Allow easier modification of previously formalized material
- Machine learning and AI; some types of machine learning based tools can already be helpful when formalizing. Can they they eventually produce a page of mathematics given a brief prompt? Can they big ideas autonomously? Even being able to check that routine arguments similar to those in the literature hold would be very useful.

Would like to have as many of these as possible without sacrificing benefits of existing presentation methods; readability and easy comprehension for trained people (any mathematician) and ease of writing. Not only should this technology make it easier for computers to do maths, but ideally also humans.

Demo time!

Lets see how proving a result using a proof assistant (the most common way of formalizing advanced mathematics) looks.

We will prove Euclid's theorem using Mathlib, a large library of formalized mathematics in Lean, that has attracted a community of mathematicians as developers and users.

Some recent high profile examples of mathematical formalization projects

- Tao led a project to formalize a proof of the Polynomial Freiman-Rusza conjecture (Gowers-Green-Manners-Tao). Finished 3 weeks! 25 contributors, analogous to a long reading group / summer school.
- Scholze challenged the formalization community to formalize key result in liquid condensed mathematics (Clausen-Scholze). This took a group of up to 25 people a year and a half.
- Dillies Mehta and Bloom formalizing results in additive combinatorics in "real time"
- Massot, van Doorn and Nash - Gromov's h-principle and sphere eversion
- Buzzard, Commelin and Massot, formalizing the definition of a perfectoid space.
- Formal proof of the Kepler conjecture

Future goals

- Strong evidence that with effort most modern mathematical results can be formalized. Challenge is to make this not just a one-off, but a sustainable process that doesn't require as many person-hours as the above projects took. Can come from finding efficient mathematical arguments but also improving the formalization language and surrounding tools. To be closer to the level of abstraction we are used to.

Future goals

- Strong evidence that with effort most modern mathematical results can be formalized. Challenge is to make this not just a one-off, but a sustainable process that doesn't require as many person-hours as the above projects took. Can come from finding efficient mathematical arguments but also improving the formalization language and surrounding tools. To be closer to the level of abstraction we are used to.
- Standard undergraduate curriculum is close to all already formalized.

Future goals

- Strong evidence that with effort most modern mathematical results can be formalized. Challenge is to make this not just a one-off, but a sustainable process that doesn't require as many person-hours as the above projects took. Can come from finding efficient mathematical arguments but also improving the formalization language and surrounding tools. To be closer to the level of abstraction we are used to.
- Standard undergraduate curriculum is close to all already formalized.
- Goals for future formalization projects are to consider, higher level arguments, areas less obviously formal, those with more appeals to intuition or unverified computation, or sheer volume of very technical material or techniques. Can most areas of mathematics be conveniently expressed in a proof assistant?

Arithmetic Geometry

Formalization and arithmetic geometry sometimes feels less well developed than many other fields, why?:

- Requires a lot of theory to get to some basic tools, scheme theory, cohomology theories

Arithmetic Geometry

Formalization and arithmetic geometry sometimes feels less well developed than many other fields, why?:

- Requires a lot of theory to get to some basic tools, scheme theory, cohomology theories
- social reasons, researchers in arithmetic geometry interested in formalization ended up working on projects like Liquid Tensor Experiment

Arithmetic Geometry

Formalization and arithmetic geometry sometimes feels less well developed than many other fields, why?:

- Requires a lot of theory to get to some basic tools, scheme theory, cohomology theories
- social reasons, researchers in arithmetic geometry interested in formalization ended up working on projects like Liquid Tensor Experiment
- ... maybe its not true at all?

Some general results of interest that have been formalized

- Local and global fields
- Algebraic number theory tools, finiteness of class group, Dirichlet's unit theorem, Kummer-Dedekind
- (Absolute) Galois groups (and some cohomology thereof)
- Ideles and Adeles
- Witt Vectors (and p-adic fields)
- L-series, and modular forms
- p-adic L-functions
- Ostrowski's theorem (again at LFTCM last week!)
- Divided power structures (towards B_{dR})
- Schemes
- Elliptic Curves, group law in all characteristics

Roblot, de Frutos Fernandez, Baanen, Dahmen, Narayanan, Nuccio, Chambert-Loir, Loeffler, Stoll, Commelin, Lewis, Livingston, Birkbeck, etc

In the (near) future

Alex Kontorovich and Terrence Tao have started a project to formalize the PNT+ project, to formalize the Prime Number theorem in Lean, and other related results such as Chebotarev and Dirichlet's theorem on primes in arithmetic progressions.

We will hear more on this from either Michael Stoll or Alex Kontorovich in this seminar!

Chebotarev is a vital technical tool for Arithmetic Geometry, underpinning many important results and techniques so this is likely to be very useful.

In the (near) future

Alex Kontorovich and Terrence Tao have started a project to formalize the PNT+ project, to formalize the Prime Number theorem in Lean, and other related results such as Chebotarev and Dirichlet's theorem on primes in arithmetic progressions.

We will hear more on this from either Michael Stoll or Alex Kontorovich in this seminar!

Chebotarev is a vital technical tool for Arithmetic Geometry, underpinning many important results and techniques so this is likely to be very useful.

Kevin Buzzard is starting a longer term project to formalize much of the mathematics around Fermat's Last Theorem

Algebraic number theory: FLT-regular

With Birbeck, Brasca, Rodriguez, Yang we formalized the proof due to Kummer of Fermat's last theorem for regular primes.

- It pays to take the time to find the right proof, a proof reducing the main technical tool, Kummer's Lemma, to Hilbert's theorem 92 avoiding CFT was in the exercises of Swinnerton-Dyer's textbook!

Algebraic number theory: FLT-regular

With Birbeck, Brasca, Rodriguez, Yang we formalized the proof due to Kummer of Fermat's last theorem for regular primes.

- It pays to take the time to find the right proof, a proof reducing the main technical tool, Kummer's Lemma, to Hilbert's theorem 92 avoiding CFT was in the exercises of Swinnerton-Dyer's textbook!
  Kummer: Let p be a regular prime and let $u \in \mathbb{Z}[\zeta_p]^\times$. If $u \equiv a \bmod p$ for some $a \in \mathbb{Z}$, then there exists $v \in \mathbb{Z}[\zeta_p]^\times$such that $u = v^p$.
  Hilbert: Let K/F be a Galois extension of $F = \mathbb{Q}(\zeta_p)$ with Galois group $\mathrm{Gal}(K/F)$ cyclic with generator σ. Then there exists a unit $\eta \in \mathcal{O}_K$ such that $N_{K/F}(\eta) = 1$ but does not have the form $\epsilon/\sigma(\epsilon)$ for any unit $\epsilon \in \mathcal{O}_K$.
  See: `https://leanprover-community.github.io/flt-regular/blueprint/`

Algebraic number theory: FLT-regular

With Birbeck, Brasca, Rodriguez, Yang we formalized the proof due to Kummer of Fermat's last theorem for regular primes.

- It pays to take the time to find the right proof, a proof reducing the main technical tool, Kummer's Lemma, to Hilbert's theorem 92 avoiding CFT was in the exercises of Swinnerton-Dyer's textbook!
- Required developing a theory of cyclotomic fields and rings, its much easier to work with abstract generality than with the model you actually need.

Basic Diophantine equations

With Baanen, Coppola and Dahmen (2023) we wanted to try some classic examples of arithmetic geometry: determining explicitly the integral points on some elliptic curves. E.g.

$$y^2 = x^3 - 5, y^2 = x^3 - 17$$

this is via Mordell-style descent

Basic Diophantine equations

With Baanen, Coppola and Dahmen (2023) we wanted to try some classic examples of arithmetic geometry: determining explicitly the integral points on some elliptic curves. E.g.

$$y^2 = x^3 - 5, y^2 = x^3 - 17$$

this is via Mordell-style descent

Lessons learned:

- theory can be easier than calculation, when calculating we tend to make a lot more steps implicitly e.g. when saying the product of two explicit ideals in some number field is nontrival in class group

Basic Diophantine equations

With Baanen, Coppola and Dahmen (2023) we wanted to try some classic examples of arithmetic geometry: determining explicitly the integral points on some elliptic curves. E.g.

$$y^2 = x^3 - 5, y^2 = x^3 - 17$$

this is via Mordell-style descent

Lessons learned:

- theory can be easier than calculation, when calculating we tend to make a lot more steps implicitly e.g. when saying the product of two explicit ideals in some number field is nontrival in class group
- Having the system automatically do calculations in explicit rings (given by a times table, or in positive characteristic), is a massive help, as is being able to add this sort of of functionality as a user is essential.

Certification

Lessons learned (cont):

- More generally importing results from computer algebra systems will be useful when formalizing explicit type results.

Certification

Lessons learned (cont):

- More generally importing results from computer algebra systems will be useful when formalizing explicit type results.
- How do you "certify" a class or unit group computation? What is the shortest/simplest proof that a given class group is what is claimed assuming that checking takes a lot more work than finding. With the analytic class number formula this becomes easier, but then leads to how to do numerics for L-functions with verified bounds.

Certification

Lessons learned (cont):

- More generally importing results from computer algebra systems will be useful when formalizing explicit type results.
- How do you "certify" a class or unit group computation? What is the shortest/simplest proof that a given class group is what is claimed assuming that checking takes a lot more work than finding. With the analytic class number formula this becomes easier, but then leads to how to do numerics for L-functions with verified bounds.
- Doing things in a low-tech way can lead you to look at some very pretty mathematics. Previously I would never have thought about how to bound class groups of quadratic fields with anything other than Minkowski, but it turns out there is a fascinating connection to the Farey sequence there.

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 Z.

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

Thanks for listening

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 nice arguments and general techniques can be a lot of fun, and the tool may occasionally surprise you.

If you are interested in getting more actively involved check out `https://leanprover-community.github.io/` and `https://leanprover.zulipchat.com/` also see if there is a "Lean for the curious mathematician" or even a "Lean for the curious arithmetic geometer" near you.

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

```
26
27  class perfectoid_space (X : Type u) [topological_space X] extends adic_space X :=
28  (perfectoid_cover : ∀ x : X, ∃ (U : opens X) (A : Huber_pair) [perfectoid_ring A],
29    (x ∈ U) ∧ (𝒞.Spa A) ≅_𝒞 (locally_ringed_valued_space.to_𝒞.restrict U))
30
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

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

```
26
27  class perfectoid_space (X : Type u) [topological_space X] extends adic_space X :=
28  (perfectoid_cover : ∀ x : X, ∃ (U : opens X) (A : Huber_pair) [perfectoid_ring A],
29    (x ∈ U) ∧ (𝒞.Spa A) ≅_𝒞 (locally_ringed_valued_space.to_𝒞.restrict U))
30
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

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

```
26
27  class perfectoid_space (X : Type u) [topological_space X] extends adic_space X :=
28  (perfectoid_cover : ∀ x : X, ∃ (U : opens X) (A : Huber_pair) [perfectoid_ring A],
29    (x ∈ U) ∧ (𝒞.Spa A) ≅_𝒞 (locally_ringed_valued_space.to_𝒞.restrict U))
30
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

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

```
26
27  class perfectoid_space (X : Type u) [topological_space X] extends adic_space X :=
28  (perfectoid_cover : ∀ x : X, ∃ (U : opens X) (A : Huber_pair) [perfectoid_ring A],
29    (x ∈ U) ∧ (𝒞.Spa A) ≅_𝒞 (locally_ringed_valued_space.to_𝒞.restrict U))
30
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

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.

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.

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 $\mathbb{Q}(\sqrt{-5})$!

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.

The other strategy is to leverage existing computer algebra systems

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 $K/\mathbb{Q}$ is a cyclic Galois extension and $x \in \mathbb{Q}$ 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 $K/\mathbb{Q}$ is a cyclic Galois extension and $x \in \mathbb{Q}$ 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!

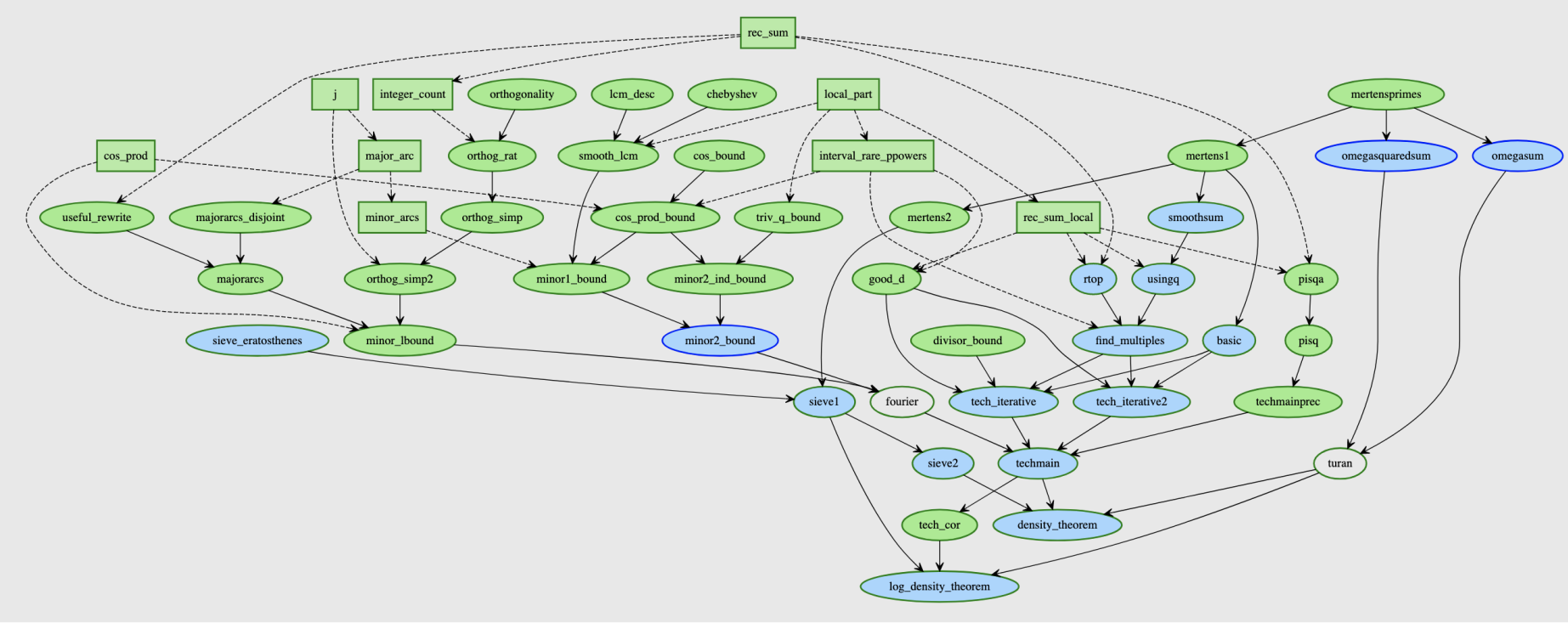
Unit fractions

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 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