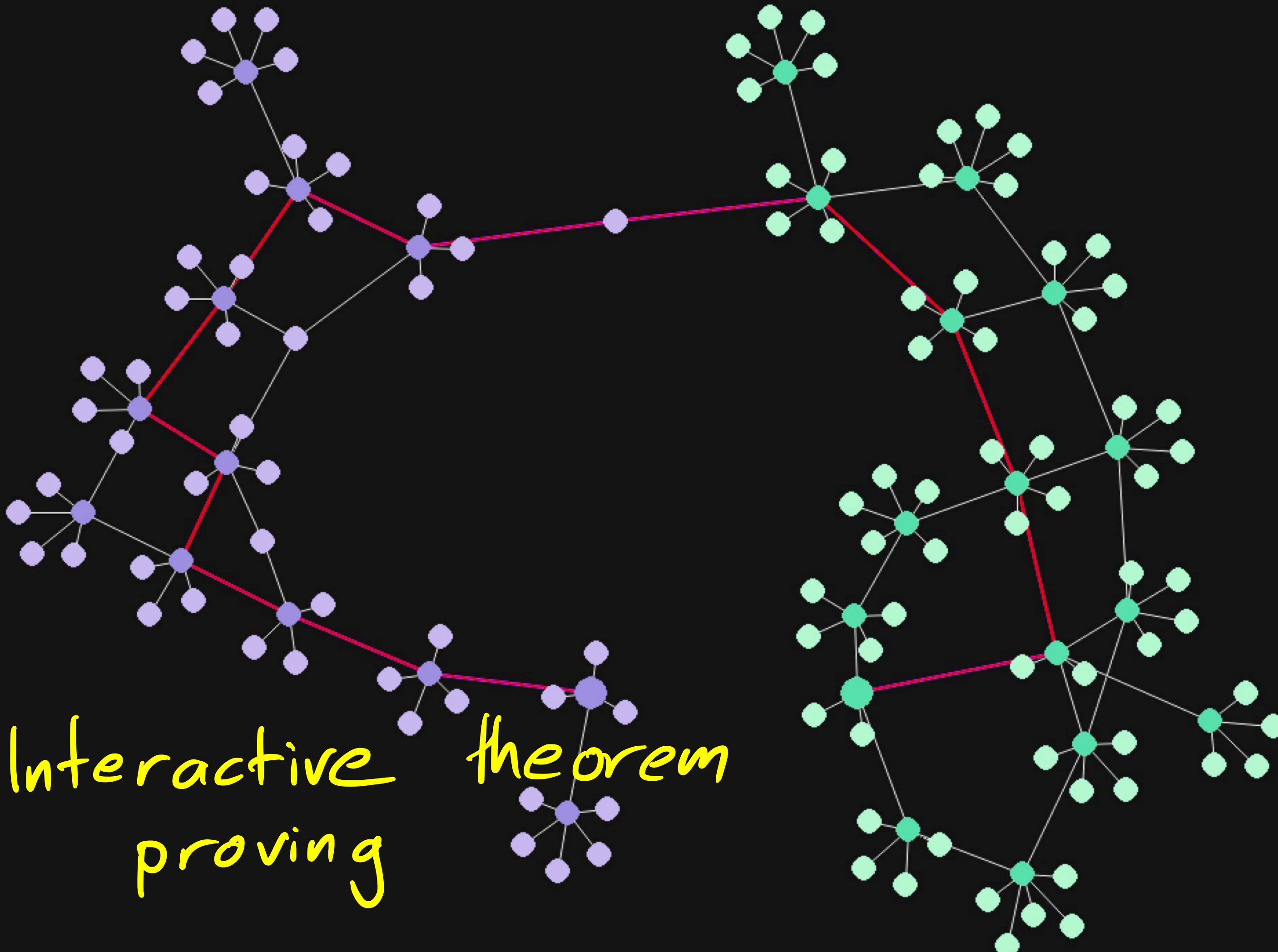


Interactive theorem proving

Scott Morrison – UC Berkeley Colloquium 2020-02-13

Demo

```
theorem infinitude_of_primes (N : ℕ) : ∃ p ≥ N, prime p :=
begin
  use min_fac (fact N + 1),
  split,
  { by_contradiction, back, },
  { back },
end
```

Interactive theorem provers are
not yet useful
to most mathematicians.

But computer scientists are building
amazing tools
that are getting close.

More mathematicians should get involved!

Why!?

- a 'crisis' in mathematics?
  - proofs not checked in refereeing
  - the 'to appear' problem
  - insufficient rigour?
- deeper understanding?
- becoming better mathematicians?
  - a dream for now
  - computer algebra systems have already made us more powerful
  - interactive theorem provers with strong automation and communicability may help us explore and construct proofs.

Why not?

- Sink effort into particular proof systems/foundations

- It's <u>hard</u>. On bad days, the computer is bafflingly obtuse.
  - spell out pedantic details
  - fight with the (dependent) type system
  - slow verification
  - express ideas 'unnaturally' to fit the logic
  - "transport of structure"; a Faustian bargain!

- Maybe it's <u>too hard</u>?

Examples

- The four-colour theorem (Gonthier et al) (1976/1997/2005)
  & the Kepler conjecture (Hales et al) (1998/2015)
  - based on verified decision procedures to handle a big case bash!
- The odd order theorem (Gonthier et al) (1962/2012)
  - massive, but 'easy' (character theory...)
- The definition of a perfectoid space (Buzzard-Commelin-Massot) (2012/2019)
  (from Scholze's Fields Medal work)
  - perhaps the conceptually deepest mathematics yet taught to the computer!

```
/-
Perfectoid Spaces

by Kevin Buzzard, Johan Commelin, and Patrick Massot

Definitions in this file follow Scholze's paper: Étale cohomology of diamonds,
specifically Definition 3.1 and 3.19
-/
```

arXiv:1910.12320

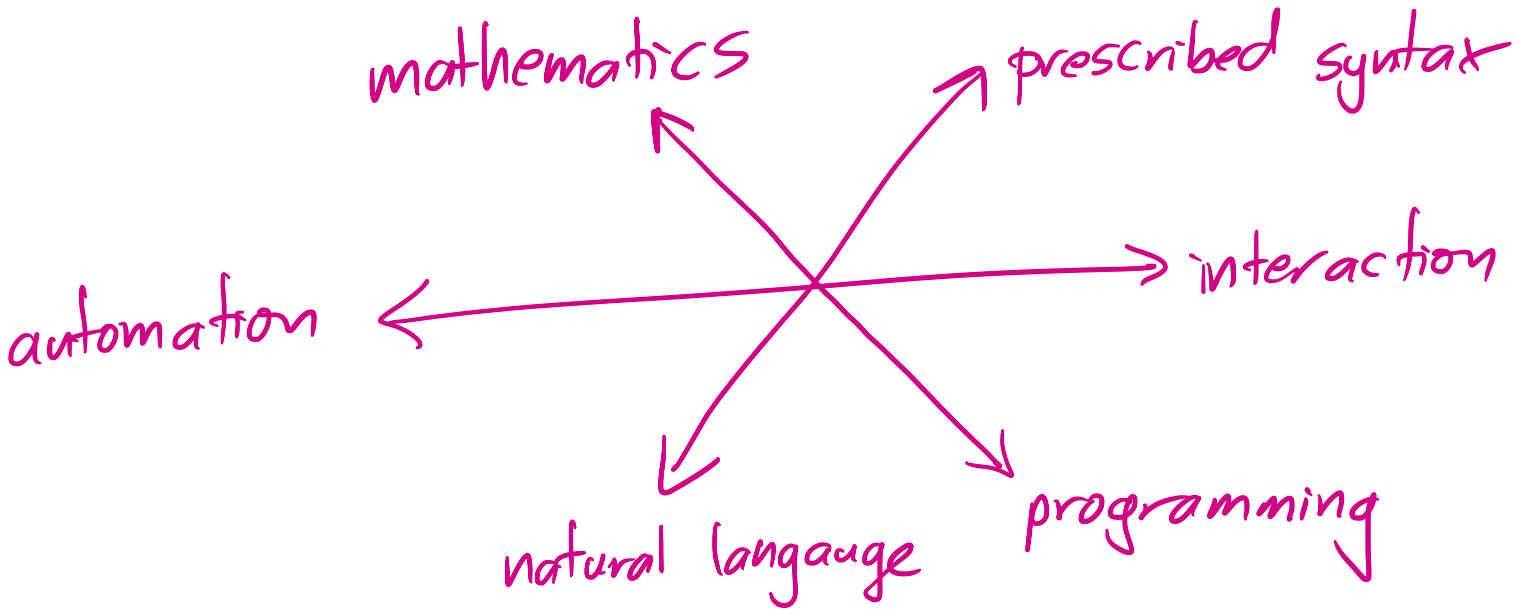
```
-- We fix a prime number p
parameter (p : Prime)

structure perfectoid_ring (R : Type) [Huber_ring R] extends Tate_ring R : Prop :=
(complete  : is_complete_hausdorff R)
(uniform   : is_uniform R)
(ramified  : ∃ ϖ : pseudo_uniformizer R, ϖ^p | p in R°)
(Frobenius : surjective (Frob R°/p))

/-- Condition for an object of CLVRS to be perfectoid: every point should have an open
neighbourhood isomorphic to Spa(A) for some perfectoid ring A.-/
def is_perfectoid (X : CLVRS) : Prop :=
∀ x : X, ∃ (U : opens X) (A : Huber_pair) [perfectoid_ring A],
  (x ∈ U) ∧ (Spa A ≅ U)

/-- The category of perfectoid spaces.-/
def PerfectoidSpace := {X : CLVRS // is_perfectoid X}
```

Theorem provers

- use a variety of different logics

  (ZF, higher order logic, dependent type theory, homotopy type theory)

- make different tradeoffs



There's a whole zoo of systems out there!

Isabelle/HOL, Lean, Coq, Agda, Mizar, Arend, ...

A crash course in dependent types

- every term has a fixed type, e.g. $3 : \mathbb{N}$

  (whereas in ZF, officially "3 is a topology on 2")

- types can depend on terms, e.g.

$$X : \mathrm{Top}$$
$$\mathcal{O} : \text{Sheaf Ring } X$$
$$x : X$$
$$p : \text{stalk\_at } \mathcal{O}\ x$$

- propositions are types too:

  is_prime 57

  is the (usually empty) type of "proofs that 57 is prime"

$\Rightarrow$ proving a theorem is the same as constructing a function

Lean

- try it online at http://leanprover-community.github.io/lean-web-editor/
- core language developed at Microsoft Research
- mathlib is an open source Lean library arXiv:1910.09336
  - mathematics
  - programming
  - proof automation
- an active community of mathematicians and computer scientists
  - organised through a friendly! online forum at http://leanprover.zulipchat.com/
  - a flexible and public review process for new submissions.
- 215k LOC. Noetherian rings & manifolds with corners
  but no Cauchy integral formula.

Interaction and automation

```
def yoneda_long : C ⥤ ((Cᵒᵖ) ⥤ Type v₁) :=
{ obj := λ X,
  { obj := λ Y, (unop Y) → X,
    map := λ Y Y' f g, f.unop ≫ g,
    map_comp' := begin intros, ext1, dsimp, erw [category.assoc] end,
    map_id' := begin intros, ext1, dsimp, erw [category.id_comp] end },
  map := λ X X' f,
    { app := λ Y g, g ≫ f,
      naturality' := begin intros, ext1, dsimp, simp end },
  map_comp' := begin intros, ext1, ext1, dsimp, simp end,
  map_id' := begin intros, ext1, ext1, dsimp, simp end }.

def yoneda_short : C ⥤ ((Cᵒᵖ) ⥤ Type v₁) := ƛ X, ƛ Y, (unop Y) → X.

def yoneda_lemma : (yoneda_pairing C) ≅ (yoneda_evaluation C) :=
{ hom := { app := λ F x, ulift.up ((x.app F.1) (𝟙 (unop F.1))) },
  inv := { app := λ F x, { app := λ X a, (F.2.map a.op) x.down } } }.
```

New automation is 'easy' in Lean

- mathematicians can do it, not just langauge developers
- Lean is it's own 'metalangauge':
  you can write unsafe programs that help construct
  proof scripts, which are then rigourously verified.

Examples

- use, by_contradiction, back are all written in Lean itself.
- tidy very loosely follows a scheme suggested by Ganesalingam-Gowers
  for 'human-style' automation,
  and is good at discharging goals with 'follow your nose' proofs.
- rewrite_search proves $A=B$ by finding chains $A=E_0=E_1=\cdots=E_k=B$
  consisting of partial rewrites using available lemmas,
  guided by edit distance heuristics and simple machine learning
  to identify important features.

Outlook for mathematicians

- Theorem provers to help teach proof?
- Students are already using these tools
  (Riesz representation theorem, combinatorial games, CW complexes, braid groups)
- Initial uses in 'real maths' papers.

A CORRECTED QUANTITATIVE VERSION OF THE MORSE LEMMA

SÉBASTIEN GOUËZEL AND VLADIMIR SHCHUR

Abstract. There is a gap in the proof of the main theorem in the article [Shc13a] on optimal bounds for the Morse lemma in Gromov-hyperbolic spaces. We correct this gap, showing that the main theorem of [Shc13a] is correct. We also describe a computer certification of this result.

The new proof of Theorem 1.1 has been completely formalized in Isabelle/HOL in [Gou18]. Therefore, the above theorem is certified. Here is this statement as proved in Isabelle/HOL.

```
theorem (in Gromov_hyperbolic_space) Morse_Gromov_theorem':
  fixes f::"real ⇒ 'a"
  assumes "lambda C-quasi_isometry_on {a..b} f"
          "geodesic_segment_between G (f a) (f b)"
  shows "hausdorff_distance (f`{a..b}) G ≤ 92 * lambda^2 * (C + deltaG(TYPE('a)))"
```

- Massive libraries to be built
- Lots of automation needed, but lots of low-hanging fruit.
- Collaboration between mathematicians, computers, and computer scientists! (logicians, linguists, ...)

examples — Formal Abstracts, Google AI, IMO.

Getting started

- Google "natural numbers game Lean"

  an online tutorial, for mathematicians, from the basics

- Install a local copy: google

  "mathlib github" and follow the installation instructions.

- Read arXiv:1910.09336, an intro to the mathlib library

  and "Theorem proving in Lean".

- http://leanprover.zulipchat.com/

  Come to the "new members" stream, say hi!

  They'll help install, answer questions, suggest projects, debug.