Synthetic Geometry in Lean 4

an early experience report with Mathlib4

Xuanrui Qi

Graduate School of Mathematics, Nagoya University

October 30, 2023
Theorem Proving and Provers 2023
@ Tokyo Institute of Technology

The story

Big picture Work with geometric objects *synthetically*, i.e., define them not by construction, but by axioms imposed on them

Why? Easier to work with, and especially within theorem provers

cont'd significantly less: non-constructive constructions, constructions that are only up to equivalence, etc.

Topology Yes, no more point-set topology! They tend to be painful to deal with in PAs.

Why Lean 4?

- Initially, tried to develop with Coq using MathComp library
- Problem 1: MathComp is relatively under-developed in terms of commutative algebra
- Problem 2: MathComp is fundamental based on computability/decidability
- Easy to work with completely computable and completely classical things, but difficult to strike a balance in between
- Lean has a big math library (Mathlib) and makes no computability assumptions!

First goals

The goal To define the "Zariski axioms" [CCH23], i.e. axioms that characterize the "base ring" R

Spec The **Spec** of an R-algebra A is simply $\mathrm{Hom}(R, A)$!

Axiom 1: locality (Loc) R is a local ring

Axiom 2: quasi-coherence (SQC) for any finitely presented R-algebra A, $a \mapsto (\phi \mapsto \phi(a)) : A \to (\mathbf{Spec}\ A \to R)$ is an isomorphism of R-algebras

On axiom 3 Axiom 3, *Zariski choice*, is not required for most results, so we focus on formalizing AX1 and AX2 for the time being

The challenges

- Even to *define* Loc and SQC, a non-trivial amount of commutative algebra is needed
- Definition of local rings, finitely presented algberas, etc.
- Often overlooked non-trivial facts too: e.g., if R is commutative and A, B are R-algebras, then $\mathrm{Hom}(A, B)$ is also an R-algebra
- Currently, the only theorem prover that has a sizable commutative algebra library is Lean 4

Lean experience I: challenges with the theorem prover

Module system: Lean does not have a module system, which is confusing for someone coming from Coq. The best approximation is the Haskell-style type class system.

Search tactic: Lean does not have a Coq-style `search` tactic. There are approximations but they can't quite emulate it. No real good way to search things!

Error messaging: Specific to Lean 4: since most of Lean 4 is defined in Lean using meta-programming, error messages can be quite confusing.

Lean experience II: pragmatic challenges

Documentation Documentation is still lacking. Existing documentation is still for Mathlib3, and it's not clear what has changed. Eventually, I resorted to searching the source code on GitHub.

Work flow Since it can be very hard to search for theorems and results, I developed a work flow where, if I need a result that I believe should have been proven, I write it down and admit it using `sorry` for the time being instead of searching for it.

Community Lean has a vibrant and helpful Zulip community where knowledgeable users answer many questions. However, at the end of the day, community is not a substitute for documentation.