Formalization of Euclidean Geometry via Hilbert's Axioms

Mark Sandey, Harry Stoltz, Sudhir Murthy

Faculty Mentor: Patricio Gallardo Candela
Department of Mathematics

Formalizing Mathematics

Proof assistants, like Lean, are digital settings to build all of mathematics, from scratch, as code in a computer software. They are tools used to verify the correctness of mathematical proofs. These tools have gained widespread attention due to the efforts of Fields Medalists such as Vladimir Voevodsky and Terence Tao.

Some Advantages of Lean

- Issues certificates of correctness and detects errors.
- Utilizes *tactics*, machine-assisted scripts that mechanically fill in repetitive proof details.
- Enables trusted, wider-scale collaboration.
- Leverages techniques in AI and automation.

In 2020, Peter Scholze challenged the formalization community. He wanted to verify, what he considered, his most important result in a theory that aims to unify topology, complex geometry, and algebraic geometry.

Theorem (Clausen, Scholze) Let $0 < p' < p \leq 1$ be real numbers, S a profinite set, and V a p-Banach space. Let $\mathcal{M}_{p'}(S)$ be the space of p'-measures on S. Then $\mathrm{Ext}^i_{\mathrm{Cond(Ab)}}(\mathcal{M}_{p'}(S), V) = 0$, for all $i \geq 1$.

Within just two years, Lean's community succeeded in formalizing all prerequisites and verified the proof itself.

Logic and Lean

Lean is not based on first-order logic but on a variant of Martin-Löf's dependent type theory. It is both a proof checker and a functional programming language. The Curry-Howard correspondence unifies mathematics and computer science; propositions are *types* and proofs are *terms* of that type. For example, the function that maps each proof of proposition P to itself, `"fun p:P => p"` in Lean, is a *term of type* $P \to P$, that is a proof of proposition $P \Rightarrow P$.

Lean and set theory are two very different foundations of mathematics. Lean has three axioms to facilitate classical mathematical reasoning. It has an exhaustive library of formalized mathematics, *Mathlib*, including algebra, analysis, geometry, topology, number theory, etc.

Euclidean Geometry

Euclid's axioms, outlined in Elements (300 BC), form the basis of Euclidean Geometry. Despite their historical significance, these axioms lack the logical rigor expected by modern standards. For example, Euclid's second postulate informally states a line can be extended infinitely.

Developed in 1899, David Hilbert's 20 Axioms offer a rigorous treatment of Euclidean Geometry. They are divided into five categories:

I Incidence
II Betweenness
III Congruence
IV Parallels
V Continuity

The precision of these axioms, especially compared to Euclid's original axioms, makes them much better suited for formalization. Our work formalizes Hartshorne's description of Hilbert's Axioms, in Lean, from his textbook *Geometry: Euclid and Beyond*.

Axioms of Incidence

I1. For any two distinct points, there exists a unique line containing them.
I2. Every line contains at least two distinct points.
I3. There exist three distinct noncollinear points (that is, three points not all contained in a single line).

Lean4 Code

```
I1 : ∀p₁ p₂:Point, p₁≠p₂ → ∃!L,
Line L ∧ p₁∈L ∧ p₂∈L
I2 : ∀L, Line L → ∃p₁ p₂:Point,
p₁≠p₂ ∧ p₁∈L ∧ p₂∈L
I3 : let Colinear (p₁ p₂ p₃:Point) : Prop
:= ∃L, Line L ∧ p₁∈ L ∧ p₂ ∈ L ∧ p₃ ∈ L
   ∃p₁ p₂ p₃:Point, p₁ ≠ p₂ ∧ p₁ ≠ p₃ ∧ p₂ ≠ p₃
   ∧ ¬ Colinear p₁ p₂ p₃
```

Hartshorne introduces the axioms in *higher-order logic*, where one can quantify over sets of *points*.

- `Point` is a primitive type.
- `Line` is a primitive term of type `Set Point→Prop`, equivalently written as `(Point→Prop)→Prop`.

Lean handles sets which are built-in as typed families.

Formalizing Euclidean Geometry

Proposition 7.1 in Hartshorne

The set of points in a plane not on a line l can be partitioned into two disjoint subsets: two points are in the same subset if and only if the segment between them does not intersect l.

We first translate this proposition to Lean.

Lean4 Code

```
lemma Prop7_1 (l:Set Point) (_ :Line l) :
∃S₁ S₂:Set Point, (∃x,x∈ S₁)∧ (∃x,x ∈ S₂)
∧ S₁∩S₂=∅ ∧ S₁ ∪ S₂ = univ \ l ∧
(∀A B:Point, A ∉ l → B ∉ l →
 ((A ∈ S₁ ∧ B ∈ S₁) ∨ (A ∈ S₂ ∧ B ∈ S₂))
  ↔ Seg A B ∩ l = ∅))
```

The "segment between two points" is defined as the set of all points between them (including the endpoints).

Lean4 Code

```
def Seg (A B:Point):Set Point:={P | (fun
X ↦ X = A ∨ X = B∨ Between B X A) P}
```

This uses the first two categories of Hilbert's Axioms. The second category axiomatizes the primitive ternary *"Betweenness"* relation, which is a primitive term of type `Point→Point→Point→Prop`.

Proof Sketch

The strategy is to define an equivalence relation which partitions the set of points not in l into exactly two equivalence classes — namely, the two "sides" of l.

Lean4 Code

```
let R (A B:Point) : Prop :=
 A=B ∨ ∀ᵉx ∈Seg A B, x ∉ l
have R_refl : ∀A:Point, A ∉ l → R A A :=
 by intro A _; dsimp; left; rfl
have R_symm : ∀A B:Point, A ∉ l → B ∉ l
 → R A B → R B A := by
 intro A B _ _ hR; obtain (hR | hR) := hR
  · left; symm; exact hR
  · right; rw [← Note7_1 A B]; exact hR
```

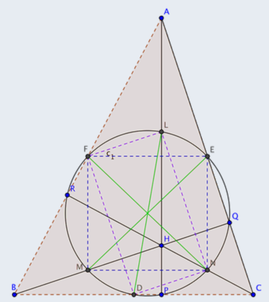
The rest of the proof has over 300 lines of code!

Beyond Hilbert's Axioms

In 1822, K. Feuerbach and O. Terquem discovered a new result in Euclidean geometry:

Nine points circle theorem

Given any triangle, one can construct a circle containing the midpoint of each side, the foot of each altitude, and the midpoint of each line segment connecting the orthocenter to one of the vertices of the triangle.



Hilbert's axiomatization of geometry does not include a definition for circles, so stating the nine-point circle theorem using only Hilbert's axioms is impossible. Instead, we will show how it might look like if it is stated in Lean.

Lean4 Code

```
variable (Circle : Set Point→Prop)
 (Mid : Point→Point→Point→Prop)
 (Ortho : Point→Point→Point→Point→Prop)
 (Perp : Point→Point→Point→Point→Prop)

theorem Nine_Points (A B C : Point)
  (_: A ≠ B) (_: A ≠ C) (_: B ≠ C):
  ∃ (Z : Set Point),
  ∃ D E F P Q R M N L H: Point,
  Circle Z ∧ Ortho A B C H ∧ Mid B D C
  ∧ Mid C E A ∧ Mid A F B ∧ Mid A L H
  ∧ Mid B M H ∧ Mid C N H ∧ Perp B C A P
  ∧ Perp A B C R ∧ Perp A C B Q ∧ D ∈ Z
  ∧ E ∈ Z ∧ F ∈ Z ∧ Q ∈ Z ∧ R ∈ Z
  ∧ M ∈ Z ∧ N ∈ Z ∧ L ∈ Z:= by sorry
```