A Lean Verification of the Empty Hexagon Theorem

Cayden Codel, James Gallicchio, Wojciech Nawrocki, and Bernardo Subercaseaux

Carnegie Mellon University, Pittsburgh, PA 15213, USA
{ccodel, jcallicc, wnawrock, bsuberca}@andrew.cmu.edu

Abstract

A recent breakthrough in computer-assisted mathematics proved that every set of 30 points contains an empty hexagon, thus closing a line of research opened by Erdős and Szekeres in 1935. The proof combines geometric insights with automated reasoning techniques, resulting in a sophisticated propositional encoding of the problem with $O(n^4)$ many clauses (where n ends up being 30) that is then solved using a SAT solver and parallel computation. This paper presents a formalization of the proof in the Lean theorem prover, thus certifying its correctness. To achieve this we have formalized several ideas of both discrete computational geometry and SAT-encodings that have been successfully applied in different Erdős-Szekeres type problems. We hope this work sets a new standard for verification when extensive computation is used for discrete geometry problems, and increases the trust of the community in the results obtained by these methods.

1 Introduction

Mathematical results that require extensive computational proofs have often raised suspicion amongst mathematicians. A landmark example is the *four-color theorem*, which states that any planar graph can be colored with at most four colors. In 1879, Alfred Kempe published a proof of the four-color theorem, which was discovered to be incorrect 11 years later by Headwood. The main idea of Kempe's proof was salvaged by Headwood to prove the weaker *five-color theorem*. A full proof of the four-color theorem was not found until 1976, when Kenneth Appel and Wolfgang Haken used a computer to verify the 1834 cases in which they proved that a minimal counterexample must exist. The proof of Appel and Haken was controversial, as some small errors had been found in their initial calculations of 1976. Finally, in 2005, Georges Gonthier formalized a full proof of the four-color theorem in the Coq proof assistant, thus culminating with any lingering doubts.

2 The Sortedness Assumption and Symmetry Breaking

Both for computational efficiency and simplifying the formalization, it is convenient to assume *without loss of generality*, that the lists of points we will be dealing with are in *structured position*, meaning that any such list of points $L = (p_0, \ldots, p_{n-1})$ satisfies the following properties:

- **(x-order)** The points are sorted with respect to their x-coordinates, i.e., $x(p_i) < x(p_j)$ for all $i < j$.
- **(General Position)** No three points are collinear, i.e., for all $i < j < k$, we have $\sigma(p_i, p_j, p_k) \neq 0$.
- **(CCW-order)** All orientations $\sigma(p_0, p_i, p_j)$, with $0 < i < j$, are counterclockwise.

This assumption is by now a standard in computational results regarding Erdős-Szekeres type problems, as for example in the work of Peters and Szekeres, or Scheuher.

Sketch. Let π be an *orientation property* of a set of points. We want to show that if π holds for a set of points S in general position, then it holds for a list of points L in *structured* position. Label the points so that p_0 is the leftmost point (i.e., $x_0 \leq x_i$ for all i), and define $\theta_i \in [0, \pi)$ as *angle* of the line segment $p_0\vec{p}_i$ with respect to the projective segment from $(x(p_0), \infty)$ to p_0. Then, we can sort the points by their angles, so that $0 < i < j$ implies $\theta_i < \theta_j$. As a result, we have that $\sigma(p_0, p_i, p_j) = \text{CCW}$, for all $0 < i < j$, by the slope-orientation equivalence. Then, through a series of transformations that preserve ortientations, we will ensure all coordinates $x(p_i), y(p_i)$ are non-negative and that $x(p_0) = y(p_0) = 0$. We can then apply a projective transformation

$$(x, y) \mapsto \left(x' := \frac{y}{x}, y' := \frac{1}{x}\right),$$

and use the slope-orientation equivalence to show that $\frac{y(p_i)}{x(p_i)}$ is now increasing over i, thus implying sortnedness along the x-axis.

□

```
structure gp_point_list (pts: List Point) : Prop :=
  no_three_collinear: ∀ p1 p2 p3: Point, List.Sublist [p1, p2, p3] pts
                      → σ p1 p2 p3 ≠ Orientation.Collinear

structure structured_point_list (pts: List Point) : Prop :=
  x_order: ∀ i j: (Fin pts.length), i < j → (pts.get i).x < (pts.get j).x
  general_position: gp_point_list pts
  ccw_order: ∀ p1 p2 p3: Point, List.Sublist [p1, p2, p3] pts
                      → σ (pts.get 0) p2 p3 = Orientation.CCW


theorem symmetry_breaking (π : OrientationProperty) :
  ∃ (pts: List Point), GeneralPosition pts ∧ π pts
  → ∃ (pts': List Point), structured_point_list pts' ∧ π pts' := by sorry
```

3 The Empty Triangle Theorem

Let us consider a much simpler theorem, which we call the *empty triangle theorem*. Its proof will be helpful to motivate the different aspects of our formalization of the empty hexagon theorem.

Theorem 1 (Empty Triangle Theorem)**.** *Given a set S of $n \geq 3$ points in general position, there exists 3 points $a, b, c \in S$ such that no point $d \in S$ lies inside the triangle abc.*

(Human Proof). Let a, b, x be any three points of S. Define the relation $p \prec q$ to mean that the triangle abp is contained inside the triangle abq. Note that $\prec$ is a finite partial order, and thus there must exist at least one minimal element for $\prec$. Let c be a minimal element of $\prec$, and now note that abc cannot contain any other point $d \in S$, as otherwise we would have $d \prec c$, contradicting the minimality of c. An illustration of this proof is presented in Figure 1. □

Instead of formalizing the above proof, we will formalize a SAT-based proof, with the goal of approaching the formalization of the empty hexagon theorem. Naturally, to use a SAT-solver

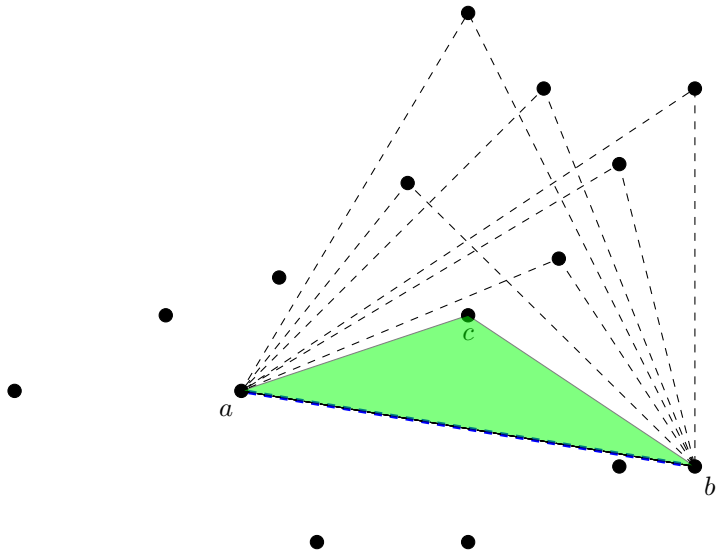

Figure 1: An illustration of the proof for Theorem 1.

we need to parameterize Theorem 1 by $n = |S|$, the number of points, and generate a different proof for each value of $n \geq 3$. For example, consider the following theorem.

```
theorem EmptyTriangle10Theorem (pts : Finset Point)
  (gp : PointFinsetInGeneralPosition pts)
  (h : pts.card = 10) :
  ∃ (p q r : Point), {p, q, r} ⊆ pts ∧ EmptyTriangleIn p q r pts
```

To complete the statement of the theorem, we must define `EmptyTriangleIn`, and before that what it means for a point to be *inside* a triangle.

```
def pt_in_triangle (a : Point) (p q r : Point) : Prop :=
  ∃ p' q' r', ({p', q', r'} : Set Point) = {p, q, r} ∧
    Sorted₃ p' q' r' ∧
    Sorted₃ p' a r' ∧
    a ≠ q' ∧ -- this isn't needed if p,q,r are in GP
    σ p' q' r' = σ p' a r' ∧
    σ p' a q' = σ p' r' q' ∧
    σ q' a r' = σ q' p' r'

/-- S is an empty triangle relative to pts -/
structure EmptyTriangleIn (p q r : Point) (pts : Finset Point) : Prop :=
  gp : InGeneralPosition₃ p q r
  empty: ∀ a ∈ pts, ¬(pt_in_triangle a p q r)
```