Formal Verification of the Empty Hexagon Number

Bernardo Subercaseaux
Carnegie Mellon University

Wojciech Nawrocki
Carnegie Mellon University

James Gallicchio
Carnegie Mellon University

Cayden Codel
Carnegie Mellon University

Mario Carneiro
Carnegie Mellon University

Marijn J. H. Heule
Carnegie Mellon University

Abstract

A recent breakthrough in computer-assisted mathematics showed that every set of 30 points in the plane in general position (i.e., no three points on a common line) contains an empty convex hexagon. Heule and Scheucher solved this problem with a combination of geometric insights and automated reasoning techniques by constructing CNF formulas ϕ_n, with $O(n^4)$ clauses, such that if ϕ_n is unsatisfiable then every set of n points in general position must contain an empty convex hexagon. An unsatisfiability proof for $n = 30$ was then found with a SAT solver using 17 300 CPU hours of parallel computation. In this paper, we formalize and verify this result in the Lean theorem prover. Our formalization covers ideas in discrete computational geometry and SAT encoding techniques by introducing a framework that connects geometric objects to propositional assignments. We see this as a key step towards the formal verification of other SAT-based results in geometry, since the abstractions we use have been successfully applied to similar problems. Overall, we hope that our work sets a new standard for the verification of geometry problems relying on extensive computation, and that it increases the trust the mathematical community places in computer-assisted proofs.

2012 ACM Subject Classification Theory of computation → Logic and verification

Keywords and phrases Empty Hexagon Number, Discrete Computational Geometry, Erdős-Szekeres

Digital Object Identifier 10.4230/LIPIcs.ITP.2024.24

Supplementary Material *Software (Source Code)*: `https://github.com/bsubercaseaux/EmptyHexagonLean/tree/itp2024`

Funding Supported by the National Science Foundation (NSF) grant CCF-2229099.

1 Introduction

Mathematicians are often rightfully skeptical of proofs that rely on extensive computation (e.g., the controversy around the four color theorem [41]). Nonetheless, many mathematically-interesting theorems have been resolved with the help of computers. SAT solving in particular has been a powerful tool for mathematics, successfully resolving Keller's conjecture [2], the packing chromatic number of the infinite grid [35], the Pythagorean triples problem [20], Lam's problem [3], and one case of the Erdős discrepancy conjecture [25]. Notably, all of these proofs follow the same two-step structure:

- **(Reduction)** Show that the mathematical theorem of interest is true if a concrete propositional formula ϕ is unsatisfiable.

15th International Conference on Interactive Theorem Proving (ITP 2024).
Editors: Yves Bertot, Temur Kutsia, and Michael Norrish; Article No. 24; pp. 24:1–24:19
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

- **(Solving)** Show that ϕ is indeed unsatisfiable.

Formal methods researchers have developed techniques that make the *solving* step reliable, reproducible, and trustworthy. For example, modern SAT solvers produce proofs of unsatisfiability in formal proof systems such as DRAT [42] that can in turn be checked with verified proof checkers such as `cake_lpr` [40]. These tools ensure that when a SAT solver declares a formula ϕ to be unsatisfiable, the formula is indeed unsatisfiable. In contrast, the *reduction* step is not as trustworthy, as it can use problem-specific mathematical insights that, when left unverified, threaten the correctness of the proof. A perfect example of a complex reduction can be found in a recent breakthrough in discrete computational geometry due to Heule and Scheucher [21]. They constructed (and solved) a formula ϕ whose unsatisfiability implies that every set of 30 points in the plane, without three in a common line, must contain an empty convex hexagon. However, as is common with such results, their reduction argument was only sketched, relied heavily on intuition, and had several gaps.

In this paper we complete and formalize their reduction in the Lean theorem prover [10]. We do so by connecting existing geometric definitions in the mathematical proof library mathlib [29] to the unsatisfiability of a particular SAT instance, thus setting a new standard for verifying results which rely on extensive computation. Our formalization is publicly available at `https://github.com/bsubercaseaux/EmptyHexagonLean/tree/itp2024`.

Verification of SAT proofs. Formal verification makes the SAT *solving* step trustworthy. For example, theorem provers and formal methods tools have been used to verify solvers [27,31,33] and proof checkers [26, 40]. However, the *reduction* step has not received similar scrutiny, with only a few reductions having been verified. For instance, Cruz-Filipe and coauthors [8,9] used the Coq proof assistant to verify the reduction of the Pythagorean triples problem [20] to SAT, and Delemazure and colleagues [11] used Isabelle/HOL to verify SAT-based results in social choice theory for which minimal unsatisfiable sets of clauses were too large to extract human-readable proofs. More generally, Giljegård and Wennerbreck [16] built a CakeML library of verified SAT encodings, which they used to write verified SAT reductions for different puzzles, such as the *N-queens* problem. In this paper, we use reduction verification techniques based on those of Codel, Avigad, and Heule [6], which they developed in Lean.

Formal verification for SAT-based combinatorial geometry was pioneered by Marić [28]. He formally verified a reduction of a case of the Happy Ending Problem (see below) to SAT in Isabelle/HOL. We compare our work to his in Section 7.

Lean. Initially developed by Leonardo de Moura in 2013 [10], the Lean theorem prover has become a popular choice for formalizing modern mathematical research. Recent successes include the *Liquid Tensor Experiment* [5] and the proof of the polynomial Freiman-Ruzsa conjecture [17,34], both of which brought significant attention to Lean. A major selling point for Lean is the mathlib project [29], a monolithic formalization of foundational mathematics. By relying on mathlib for definitions, lemmas, and proof tactics, mathematicians can focus on the interesting components of a formalization while avoiding duplication of proof efforts. In turn, by making a formalization compatible with mathlib, future proof efforts can rely on work done today. In this spirit, we connect our results to mathlib as much as possible.

The Empty Hexagon Number. In the 1930s, Erdős and Szekeres, inspired by Esther Klein, showed that for any $k \geq 3$, one can find a sufficiently large number n such that every n points in the plane in *general position* (i.e., with no three points collinear) contain a convex *k-gon*, i.e., a convex polygon with k vertices [13]. The minimal such n is denoted $g(k)$. The same authors later showed that $g(k) > 2^{k-2}$ and conjectured that this bound is tight [12]. Indeed, it is known that $g(5) = 9$ and $g(6) = 17$, with the latter result obtained by Szekeres and Peters 71 years after the initial conjecture via exhaustive computer search [39]. Larger cases remain open, with $g(k) \leq 2^{k+o(k)}$ being the best known upper bound [22, 37]. This problem is now known as the *Happy Ending Problem*, as it led to the marriage of Klein and Szekeres.

In a similar spirit, Erdős defined $h(k)$ to be the minimal number of points in general position that is guaranteed to contain a *k-hole*, or *empty k-gon*, meaning a convex k-gon with no other point inside. It is easy to check that $h(3) = 3$ and $h(4) = 5$. In 1978, Harborth established that $h(5) = 10$ [19]. Surprisingly, in 1983, Horton discovered constructions of arbitrarily large point sets that avoid k-holes for $k \geq 7$ [23]. Only $h(6)$ remained. The *Empty Hexagon Theorem*, establishing $h(6)$ to be finite, was proven independently by Gerken [15] and Nicolás [30] in 2006. In 2008, Valtr narrowed the range of possible values down to $30 \leq h(6) \leq 1717$, where the problem remained until the breakthrough by Heule and Scheucher [21], who used a SAT solver to prove that $h(6) \leq 30$, a result we refer to as the *Empty Hexagon Number.*

2 Outline of the proof

We will incrementally build sufficient machinery to prove the following theorem.

▶ **Theorem.** *Any finite set of* 30 *or more points in the plane in general position has a* 6*-hole.*

Outline of the proof. We begin Section 3 with a precise statement in Lean of the above theorem and involved geometric terms. In a nutshell, the proof consists of building a CNF formula ϕ_n such that from any set S of n points in general position without a 6-hole we can construct a satisfying assignment τ_S for ϕ_n. Then, checking that ϕ_{30} is unsatisfiable implies that no such set S of size 30 exists, thus implying the theorem. In order to construct ϕ_n, one must first discretize the continuous space $\mathbb{R}^2$. *Triple orientations*, presented in Section 4, are a way to achieve this. Concretely, any three points p, q, r in general position correspond to either a clockwise turn, denoted by $\sigma(p, q, r) = -1$, or a counterclockwise turn, denoted by $\sigma(p, q, r) = +1$, depending on whether r is above the directed line $\overrightarrow{pq}$ or not. In this way, every set S of points in general position induces an assignment $\sigma_S : S^3 \to \{-1, +1\}$ of triple orientations. We show in Section 4 that whether S contains a k-hole (i.e., `HasEmptyKGon k S`) depends entirely on σ_S. As each orientation $\sigma(p, q, r)$ can only take two values, we can represent each orientation $\sigma(p, q, r)$ with a boolean variable. Any set of points S in general position thus induces an assignment τ_S over its *orientation variables.* Because `HasEmptyKGon k S` depends only on σ_S, it can be written as a boolean formula over the orientation variables. Unfortunately, it is practically infeasible to determine if such a formula is satisfiable with a naïve encoding. In order to create a better encoding, Section 5 shows that one can assume, without loss of generality, that the set of points S is in *canonical position.* Canonicity eliminates a number of symmetries from the problem – ordering, rotation, and mirroring – significantly reducing the search space. In Section 6, we show the correctness of the efficient encoding of Heule and Scheucher [21] for constructing a smaller CNF formula ϕ_n. Concretely, we show that any finite set of n points in canonical position containing no 6-hole would give rise to a propositional assignment τ_S satisfying ϕ_n. However, ϕ_{30} (depicted in Section 6) is unsatisfiable; therefore no such set of size 30 exists and the theorem follows by contradiction. As detailed in Section 6, to establish unsatisfiability of ϕ_{30} we passed the formula produced by our verified encoder to a SAT solver, and used a verified proof checker to certify the correctness of the resulting unsatisfiability proof. The construction of ϕ_n and τ_S involves sophisticated optimizations which we justify using geometric arguments. ◀

3 Geometric Preliminaries

We identify points with elements of $\mathbb{R}^2$. Concretely, `abbrev Point := EuclideanSpace ℝ (Fin 2)`. The next step is to define what it means for a k-gon to be *empty* (with respect to a set of points) and *convex*, which we do in terms of mathlib primitives.

```
/-- ‘EmptyShapeIn S P’ means that ‘S’ carves out an empty shape in ‘P’:
the convex hull of ‘S’ contains no point of ‘P’ other than those already in ‘S’. -/
def EmptyShapeIn (S P : Set Point) : Prop :=
    ∀ p ∈ P \ S, p ∉ convexHull ℝ S

/-- ‘ConvexIndep S’ means that ‘S’ consists of extremal points of its convex hull,
i.e., the point set encloses a convex polygon. -/
def ConvexIndep (S : Set Point) : Prop :=
    ∀ a ∈ S, a ∉ convexHull ℝ (S \ {a})

/-- ‘ConvexEmptyIn S P’ means that ‘S’ forms a convex empty polygon in ‘P’ -/
def ConvexEmptyIn (S P : Set Point) : Prop :=
    ConvexIndep S ∧ EmptyShapeIn S P

/-- ‘HasEmptyKGon k P’ means that ‘P’ has a convex, empty ‘k’-gon -/
def HasEmptyKGon (k : Nat) (P : Set Point) : Prop :=
    ∃ S : Finset Point, S.card = k ∧ ↑S ⊆ P ∧ ConvexEmptyIn S P
```

Let `SetInGenPos` be a predicate that states that a set of points is in *general position*, i.e., no three points lie on a common line (made precise in Section 4). With this we can already state the core theorem.

```
theorem hole_6_theorem : ∀ (pts : Finset Point),
    SetInGenPos pts → pts.card = 30 → HasEmptyKGon 6 pts
```

At the root of the encoding of Heule and Scheucher is the idea that the `ConvexEmptyIn` predicate can be determined by analyzing only triangles. In particular, that a set s of k points in a pointset S form an empty convex k-gon if and only if all the $\binom{k}{3}$ triangles induced by vertices in s are empty with respect to S. This is discussed informally in [21, Section 3, Eq. 4]. Concretely, we prove the following theorem:

```
theorem ConvexEmptyIn.iff_triangles {s : Finset Point} {S : Set Point}
    (sS : ↑s ⊆ S) (sz : 3 ≤ s.card) :
    ConvexEmptyIn s S ↔
    ∀ (t : Finset Point), t.card = 3 → t ⊆ s → ConvexEmptyIn t S
```

Proof sketch. We first prove a simple monotonicity lemma: if $\mathsf{ConvexIndep}(s)$, then $\mathsf{ConvexIndep}(s')$ for every $s' \subseteq s$, and similarly $\mathsf{EmptyShapeIn}(s, S) \Rightarrow \mathsf{EmptyShapeIn}(s', S)$ for every set of points S. By instantiating this monotonicity lemma over all subsets $t \subseteq s$ with $|t| = 3$ we get the forward direction of the theorem. For the backward direction it is easier to reason contrapositively: if the ConvexIndep predicate does not hold of s, or if s is not empty w.r.t. S, then we want to show that there is a triangle $t \subseteq s$ that is also not empty w.r.t. S. To see this, let H be the convex hull of s, and then by Carathéodory's theorem (cf. `theorem convexHull_eq_union` from mathlib), every point in H is a convex combination of at most 3 points in s, and consequently, of exactly 3 points in s. If s is non-empty w.r.t. S, then there is a point $p \in S \setminus s$ that belongs to H, and by Carathéodory, p is a convex combination of 3 points in $s \setminus \{a\}$, and thus lies inside a triangle $t \subseteq s$ (Figure 1a). If s does

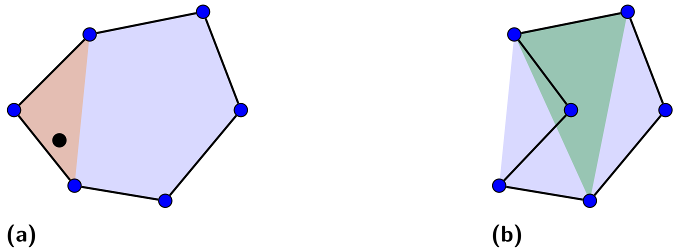


Figure 1 Illustration of the proof for `ConvexEmptyIn.iff_triangles`. The left subfigure shows how a point in $S \setminus s$ that lies inside s will be inside one of the triangles induced by the convex hull of s (orange triangle). The right subfigure shows how if the ConvexIndep predicate does not hold of s, then some point $a \in s$ will be inside one of the triangles induced by the convex hull of $s \setminus \{a\}$.

not hold ConvexIndep, then there is a point $a \in s$ such that $a \in \mathsf{convexHull}(s \setminus \{a\})$, and by Carathéodory again, a is a convex combination of 3 points in s, and thus lies inside a triangle $t \subseteq s \setminus \{a\}$ (Figure 1b). ◀

In the next section, we show how to use boolean variables to encode which triangles (and, by the above theorem, which k-holes) are empty in a pointset.

4 Triple Orientations

An essential step for obtaining computational proofs of geometric results is *discretization*: problems concerning the existence of an object $\mathcal{O}$ in a continuous search space like $\mathbb{R}^2$ must be reformulated in terms of the existence of a discrete, finitely-representable object $\mathcal{O}'$ that a computer can search for. It is especially challenging to discretize problems in which the desired geometric object $\mathcal{O}$ is characterized by very specific coordinates of points, thus requiring the computer to use floating-point arithmetic, which suffers from numerical instability. Fortunately, this is not the case for Erdős-Szekeres-type problems such as determining the value of $h(k)$, as their properties of interest (e.g., convexity and emptiness) can be described in terms of axiomatizable relationships between points and lines (e.g., point p is above the line $\overrightarrow{qr}$, lines $\overrightarrow{qr}$ and $\overrightarrow{st}$ intersect, etc.) that are invariant under rotation, translation, and even small perturbations of the coordinates. We can discretize these relationships with boolean variables, thus making us agnostic to the specific coordinates of the points. The combinatorial abstraction that has been most widely used in Erdős-Szekeres-type problems is that of *triple orientations* [21, 32], also known as *signotopes* [14, 36], Knuth's *counterclockwise relation* [24], or *signatures* [38]. Given points p, q, r, their *triple-orientation* is defined as:

$$\sigma(p,q,r) = \operatorname{sign}\det\begin{pmatrix} p_x & q_x & r_x \\ p_y & q_y & r_y \\ 1 & 1 & 1 \end{pmatrix} = \begin{cases} 1 & \text{if } p,q,r \text{ are } \textit{oriented} \text{ counterclockwise,} \\ 0 & \text{if } p,q,r \text{ are collinear,} \\ -1 & \text{if } p,q,r \text{ are } \textit{oriented} \text{ clockwise.} \end{cases} .$$

We define σ in Lean using mathlib's definition of the determinant.

```
inductive Orientation : Type where
  | cw          -- Clockwise turn
  | ccw         -- Counter-clockwise turn
  | collinear   -- Collinear

noncomputable def σ (p q r : Point) : Orientation :=
```

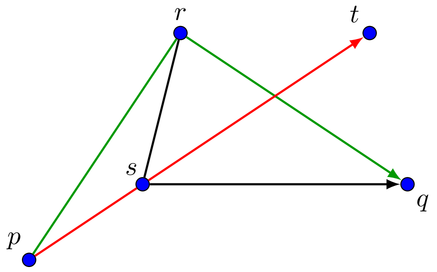


Figure 2 Illustration of triple orientations, where $\sigma(p, r, q) = -1, \sigma(r, s, q) = 1$, and $\sigma(p, s, t) = 0$.

```
  let det := Matrix.det !![p.x, q.x, r.x ; p.y, q.y, r.y ; 1, 1, 1]
  if 0 < det then ccw
  else if det < 0 then cw
  else collinear
```

Using the function σ we can define the notion of *general position* for collections (e.g., finite sets, lists, etc.) of points, simply postulating that $\sigma(p, q, r) \neq 0$ for every three distinct points p, q, r in the collection. Furthermore, we can start formalizing sets of points that are *equivalent* with respect to their triple orientations, and consequently, properties of pointsets that are fully captured by their triple orientations (*orientation properties*).

```
structure σEquiv (S T : Set Point) where
  f : Point → Point
  bij : Set.BijOn f S T
  parity : Bool
  σ_eq : ∀ (p ∈ S) (q ∈ S) (r ∈ S), σ p q r = parity ^^^ σ (f p) (f q) (f r)

def OrientationProperty (P : Set Point → Prop) :=
  ∀ {{S T}}, S ≃σ T → P S → P T -- `≃σ` is infix notation for `σEquiv`
```

Our notion of σ equivalence allows for all orientations to be flipped. The `^^^` (xor) operation does nothing when `parity` is false, and negates the orientation when `parity` is true. See Section 5 for more details.

To illustrate how these notions will be used, let us consider the property $\pi_k(S) \triangleq$ "*pointset* S contains an empty convex k-gon", formalized as `HasEmptyKGon`.

Based on `ConvexEmptyIn.iff_triangles`, we know that $\pi_k(S)$ can be written in terms of whether certain triangles are empty w.r.t S. We can define triangle membership using σ, and prove its equivalence to the geometric definition.

```
/-- `a` is in the triangle `pqr`, possibly on the boundary. -/
def PtInTriangle (a : Point) (p q r : Point) : Prop :=
  a ∈ convexHull ℝ {p, q, r}

/-- `a` is in the triangle `pqr` strictly, not on the boundary. -/
def σPtInTriangle (a p q r : Point) : Prop :=
  σ p q a = σ p q r ∧ σ p a r = σ p q r ∧  σ a q r = σ p q r

theorem σPtInTriangle_iff {a p q r : Point} (gp : InGenPos₄ a p q r) :
  σPtInTriangle a p q r ↔ PtInTriangle a p q r
```

Heule and Scheucher used the orientation-based definition [21], and as it is common in the area, its equivalence to the *ground-truth* mathematical definition was left implicit. This equivalence, formalized in `theorem σPtInTriangle_iff` is not trivial to prove: the forward direction in particular requires reasoning about convex combinations and determinants. Using the previous theorem, we can generalize to k-gons as follows.

```
def σIsEmptyTriangleFor (a b c : Point) (S : Set Point) : Prop :=
  ∀ s ∈ S, ¬σPtInTriangle s a b c

def σHasEmptyKGon (n : Nat) (S : Set Point) : Prop :=
  ∃ s : Finset Point, s.card = n ∧ ↑s ⊆ S ∧ ∀ (a ∈ s) (b ∈ s) (c ∈ s),
  a ≠ b → a ≠ c → b ≠ c → σIsEmptyTriangleFor a b c S

theorem σHasEmptyKGon_iff_HasEmptyKGon {n : Nat} (gp : ListInGenPos pts) :
    σHasEmptyKGon n pts.toFinset ↔ HasEmptyKGon n pts.toFinset
```

Then, because `σHasEmptyKGon` is ultimately defined in terms of σ, we can prove

```
lemma OrientationProperty_σHasEmptyKGon {n : Nat} : OrientationProperty
    (σHasEmptyKGon n)
```

Which in combination with `theorem σHasEmptyKGon_iff_HasEmptyKGon`, provides

```
theorem OrientationProperty_HasEmptyKGon {n : Nat} : OrientationProperty
    (HasEmptyKGon n)
```

The previous theorem is important for two reasons. First, if σ is invariant under certain point transformations (e.g., rotations, translations, etc.), then any orientation property is invariant under the same transformations. This is a powerful tool for performing symmetry breaking (see Section 5). For a concrete example, consider a proof of an Erdős-Szekeres-type result that starts by saying "*we assume without loss of generality that points $p_1, \dots, p_n$ all have positive y-coordinates.*" Since σ is invariant under translation, we can see that this assumption indeed does not impact the validity of the proof.

Second, as introduced at the beginning of this section, SAT encodings for Erdős-Szekeres-type problems use triple orientations to capture properties like convexity and emptiness, thus discretizing the problem. Because we have proved that $\pi_k(S)$ is an orientation property, the values of σ on the points in S contain enough information to determine whether $\pi_k(S)$. Therefore, we have proved that given n points, it is enough to analyze the values of σ over these points, a discrete space with at most 2^{n^3} possibilities, instead of grappling with a continuous search space on n points, $\left(\mathbb{R}^2\right)^n$. This is the key idea that will allow us to transition from the finitely-verifiable statement "*no set of triple orientations over n points satisfies property π_k*" to the desired statement "*no set of n points satisfies property π_k.*"

4.1 Properties of orientations

We now prove, assuming points are sorted left-to-right (which is justified in Section 5), that certain *σ-implication-properties* hold. Consider four points p, q, r, s with $p_x < q_x < r_x < s_x$. If p, q, r are oriented counterclockwise, and q, r, s are oriented counterclockwise as well, then it follows that p, r, s must be oriented counterclockwise (see Figure 3). We prove a number of properties of this form:

```
theorem σ_prop₁ (h : Sorted₄ p q r s) (gp : InGenPos₄ p q r s) :
    σ p q r = ccw → σ q r s = ccw → σ p r s = ccw

  [...]
```

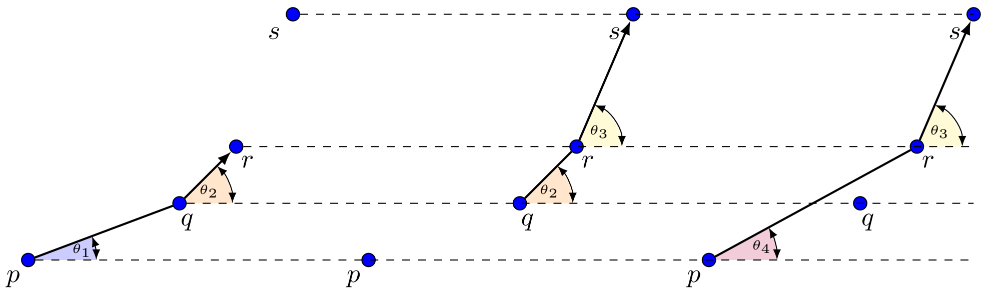


Figure 3 Illustration for $\sigma(p,q,r) = 1 \wedge \sigma(q,r,s) = 1 \implies \sigma(p,r,s) = 1$. As we have assumptions $\theta_3 > \theta_2 > \theta_4$ by the forward direction of the *slope-orientation equivalence*, we deduce $\theta_3 > \theta_4$, and then conclude $\sigma(p,r,s) = 1$ by the backward direction of the *slope-orientation equivalence*.

```
theorem σ_prop₃ (h : Sorted₄ p q r s) (gp : InGenPos₄ p q r s) :
    σ p q r = cw → σ q r s = cw → σ p r s = cw
```

Our proofs of these properties are based on an equivalence between the orientation of a triple of points and the *slopes* of the lines that connect them. Namely, if p, q, r are sorted from left to right, then (i) $\sigma(p,q,r) = 1 \iff \mathsf{slope}(\overrightarrow{pq}) < \mathsf{slope}(\overrightarrow{pr})$ and (ii) $\sigma(p,q,r) = 1 \iff \mathsf{slope}(\overrightarrow{pr}) < \mathsf{slope}(\overrightarrow{qr})$. By first proving these *slope-orientation* equivalences we can then easily prove `σ_prop₁` and others, as illustrated in Figure 3.

These properties will be used in Section 6 to justify clauses (4) and (5) of the SAT encoding; these clauses are commonly added in orientation-based SAT encodings to reduce the search space by removing some "unrealizable" orientations [21,32,36,38].

$$(\neg \mathsf{o}_{a,b,c} \vee \neg \mathsf{o}_{a,c,d} \vee \mathsf{o}_{a,b,d}) \wedge (\mathsf{o}_{a,b,c} \vee \mathsf{o}_{a,c,d} \vee \neg \mathsf{o}_{a,b,d})$$

5 Symmetry Breaking

Symmetry breaking plays a key role in SAT solving by reducing the search space of satisfying assignments for a formula [1,7], thus making a wider range of formulas practical to solve. For example, if one proves that all satisfying assignments to a formula ϕ have either (i) $x_1 = 0, x_2 = 1$, or (ii) $x_1 = 1, x_2 = 0$, and that there is a bijection between satisfying assignments of forms (i) and (ii), then one can assume, *without loss of generality*, that $x_1 = 0, x_2 = 1$, and thus add unit clauses $\overline{x_1}$ and x_2 to the formula ϕ while preserving its satisfiability. There are several techniques that can automatically find symmetry-breaking clauses, such as structured bounded variable addition [18], but it is accepted wisdom in the SAT-solving community that problem-specific symmetry breaking is more effective.

In their proof of the Empty Hexagon Number, Heule and Scheucher showed that for any list of points in general position, there exists a list of points in *canonical position* with the same triple-orientations. Canonical position is defined as follows.

▶ **Definition 1** (Canonical Position). *A list of points $L = (p_1, \ldots, p_n)$ is said to be in* canonical position *if it satisfies all the following properties:*

- ***(General Position)*** *No three points are collinear, i.e., for all $1 \leq i < j < k \leq n$, we have $\sigma(p_i, p_j, p_k) \neq 0$.*
- ***(x-order)*** *The points are sorted with respect to their x-coordinates, i.e., $x(p_i) < x(p_j)$ for all $1 \leq i < j \leq n$.*
- ***(CCW-order)*** *All orientations $\sigma(p_1, p_i, p_j)$, with $1 < i < j \leq n$, are counterclockwise.*

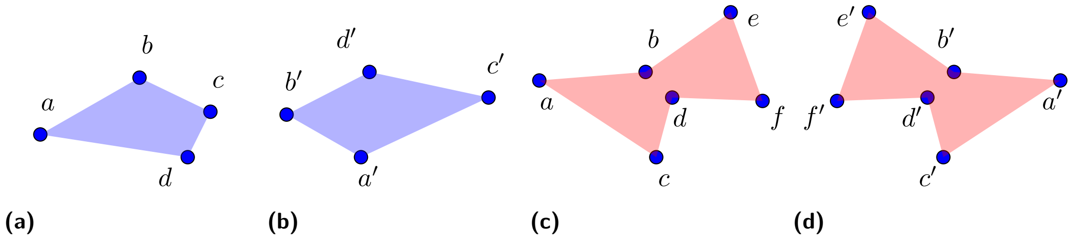


Figure 4 The pointsets depicted in Figures 4a and 4b are σ-equivalent with `parity := false` since the bijection f defined by $(a, b, c, d) \mapsto (b', d', c', a')$ satisfies $\sigma(p_i, p_j, p_k) = \sigma(f(p_i), f(p_j), f(p_j))$ for every $\{p_i, p_j, p_k\} \subseteq \{a, b, c, d\}$. On the other hand, no orientation-preserving bijection exists for Figures 4c and 4d, which are only σ-equivalent with `parity := true`.

- ***(Lex-order)*** *The list of orientations* $\left(\sigma\left(p_{\lceil\frac{n}{2}\rceil-1}, p_{\lceil\frac{n}{2}\rceil}, p_{\lceil\frac{n}{2}\rceil+1}\right), \ldots, \sigma\left(p_2, p_3, p_4\right)\right)$ *is not lexicographically smaller than the list* $\left(\sigma\left(p_{\lfloor\frac{n}{2}\rfloor+1}, p_{\lfloor\frac{n}{2}\rfloor+2}, p_{\lfloor\frac{n}{2}\rfloor+3}\right), \ldots, \sigma\left(p_{n-2}, p_{n-1}, p_n\right)\right)$.

The three ordering properties each break a different symmetry. First, the x-order property breaks symmetry due to how we label the points by ensuring that the points are labeled from left to right. The x-order property also simplifies the encoding of clauses (1)–(5), as they rely on the points being sorted. Second, the CCW-order property breaks symmetry due to *rotation* by fixing the orientations involving the leftmost point p_1.

Third, the lex-order property breaks symmetry due to *reflection*. Reflecting a set of points S over a line (e.g., with the map $(x, y) \mapsto (-x, y)$) preserves the presence of k-holes and convex k-gons. This operation does not quite preserve orientations, but rather flips them (clockwise orientations become counterclockwise and vice versa). Our definition of σ-equivalence includes a *parity* flag for this purpose: `parity := false` corresponds to the case that orientations are the same, and `parity := true` corresponds to the case that all orientations have been flipped. See the point sets in Figure 4 for an example.

The lex-order property, then, picks between a set of points and its reflection over $x = 0$. The vector of consecutive orientations from the middle to the left is assumed to be at least as big as the vector of consecutive orientations from the middle to the right. This constraint is not geometrically meaningful, but is easy to implement in the SAT encoding.

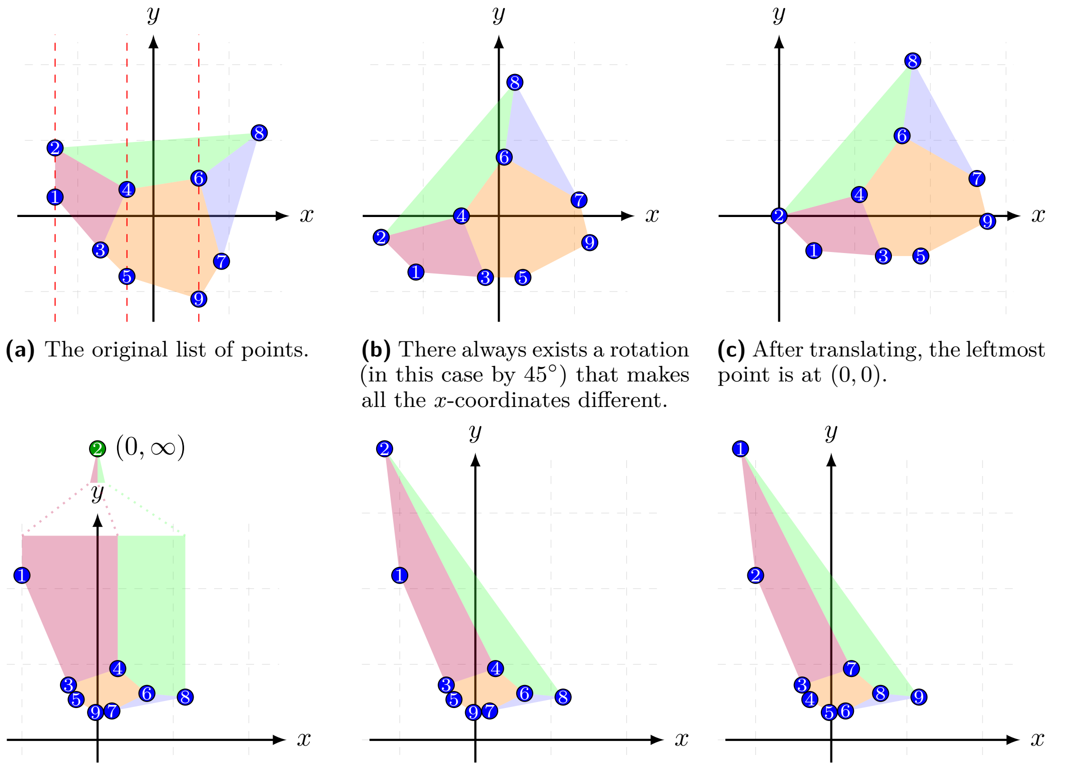
We prove that there always exists a σ-equivalent point set in canonical position.

```
theorem symmetry_breaking : ListInGenPos l →
  ∃ w : CanonicalPoints, Nonempty (l.toFinset ≃σ w.points.toFinset)
```

Proof Sketch. The proof proceeds in 6 steps, illustrated in Figure 5. In each of the steps, we construct a new list of points that is σ-equivalent to the previous one, with the last one being in canonical position.[1] The main justification for each step is that, given that the function σ is defined as a sign of the determinant, applying transformations that preserve (or, when `parity := true`, uniformly reverse) the sign of the determinant will preserve (or uniformly reverse) the values of σ.

For example, given the identity $\det(AB) = \det(A)\det(B)$, if we apply a transformation to the points that corresponds to multiplying by a matrix B such that $\det(B) > 0$, then

[1] Even though we defined σ-equivalence for sets of points, our formalization goes back and forth between sets and lists. Given that symmetry breaking distinguishes between the order of the points e.g., x-order, this proof proceeds over lists. All permutations of a list are immediately σ-equivalent.



(a) The original list of points.

(b) There always exists a rotation (in this case by 45°) that makes all the x-coordinates different.

(c) After translating, the leftmost point is at $(0, 0)$.

(d) Result after applying the map $(x, y) \mapsto (y/x, 1/x)$.

(e) Point 2 is brought back into the real plane.

(f) Points are relabeled from left to right.

Figure 5 Illustration of the proof of the symmetry breaking theorem. Note that the highlighted holes are preserved as σ-equivalence is preserved. For simplicity we have omitted the illustration of the Lex order property.

$\operatorname{sign}(\det(A)) = \operatorname{sign}(\det(AB))$, and thus orientations will be preserved. **Step 1**: we transform the list of points so that no two points share the same x-coordinate. This can be done by applying a rotation to the list of points, which corresponds to multiplying by a rotation matrix. Rotations always have determinant 1. **Step 2**: we translate all points by a constant vector t, which corresponds to multiplying by a translation matrix, to bring the leftmost point p_1 to position $(0, 0)$. As a result, every other point has a positive x-coordinate.

Let L_2 be the list of points excluding p_1 after Step 2. **Step 3**: we apply the projective transformation $f : (x, y) \mapsto (y/x, 1/x)$ to every point in L_2, showing that this preserves orientations within L_2. To see that this mapping is a σ-equivalence consider that

$$\operatorname{sign}\det\begin{pmatrix} p_x & q_x & r_x \\ p_y & q_y & r_y \\ 1 & 1 & 1 \end{pmatrix} = \operatorname{sign}\det\left(\begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}\begin{pmatrix} {}^{p_y}\!/_{p_x} & {}^{q_y}\!/_{q_x} & {}^{r_y}\!/_{r_x} \\ {}^{1}\!/_{p_x} & {}^{1}\!/_{q_x} & {}^{1}\!/_{r_x} \\ 1 & 1 & 1 \end{pmatrix}\begin{pmatrix} p_x & 0 & 0 \\ 0 & q_x & 0 \\ 0 & 0 & r_x \end{pmatrix}\right)$$

$$= \operatorname{sign}\left(1 \cdot \det\begin{pmatrix} {}^{p_y}\!/_{p_x} & {}^{q_y}\!/_{q_x} & {}^{r_y}\!/_{r_x} \\ {}^{1}\!/_{p_x} & {}^{1}\!/_{q_x} & {}^{1}\!/_{r_x} \\ 1 & 1 & 1 \end{pmatrix} \cdot p_x q_x r_x\right) = \operatorname{sign}\det\begin{pmatrix} p_y/p_x & q_y/q_x & r_y/r_x \\ 1/p_x & 1/q_x & 1/r_x \\ 1 & 1 & 1 \end{pmatrix}.$$

To preserve orientations with respect to the leftmost point $(0, 0)$, we set $f((0, 0)) = (0, \infty)$, a special point that is treated separately as follows. As the function σ takes points in $\mathbb{R}^2$ as arguments, we need to define an extension $\sigma_{(0,\infty)}(q,r) = \begin{cases} 1 & \text{if } q_x < r_x \\ -1 & \text{otherwise.} \end{cases}$, We then show that $\sigma((0,0),q,r) = \sigma_{(0,\infty)}(f(q), f(r))$ for all points $q, r \in L_2$.

Step 4: we sort the list L_2 by x-coordinate in increasing order, thus obtaining a list L_3. This can be done while preserving σ-equivalence because sorting corresponds to a permutation, and all permutations of a list are σ-equivalent by definition. **Step 5**: we check whether the Lex order condition above is satisfied in L_3, and if it is not, we reflect the pointset, which preserves σ-equivalence with `parity := true`. Note that in such a case we need to relabel the points from left to right again.

Step 6: we bring point $(0,\infty)$ back into the range by first finding a constant c such that all points in L_3 are to the right of the line $y = c$, and then finding a large enough value M such that (c, M) has the same orientation with respect to the other points as $(0,\infty)$ did, meaning that $\sigma((c,M),q,r) = \sigma_{(0,\infty)}(q,r)$ for every $q, r \in L_3$.

Finally, we note that the list of points obtained in step 6 satisfies the CCW-order property by the following reasoning: if $1 < i < j \leq n$ are indices, then

$$\begin{aligned} \sigma(p_1,p_i,p_j) = 1 &\iff \sigma((c,M),p_i,p_j) = 1 \\ &\iff \sigma_{(0,\infty)}(p_i,p_j) = 1 && \text{(By step 6)} \\ &\iff (p_i)_x < (p_j)_x && \text{(By definition of } \sigma_{(0,\infty)}) \\ &\iff \mathsf{true}. && \text{(By step 4, since points are sorted and } i < j) \end{aligned}$$

This concludes the proof. ◀

Compared to the symmetry-breaking transformation described by Heule and Scheucher, our transformation is simpler. Nonetheless, proving the above theorem in Lean was tedious, as we had to show that the properties from the previous steps were preserved at each new step, which carried substantial proof burden. In particular, steps 3 through 6 required careful bookkeeping and special handling of the distinguished point p_1.

6 The Encoding and Its Correctness

Having established the reduction to orientations, and the symmetry-breaking assumption of canonicity, we now turn to the construction of a CNF formula ϕ_n whose unsatisfiability would imply that every set of n points contains a 6-hole.[2] The formula is detailed in Section 6.

Variables. Let $S = (p_1, \ldots, p_n)$ be the list of points in canonical position. We explain the variables of ϕ_n by specifying their values in the propositional assignment τ_S that is our intended model of ϕ_n corresponding to S. We then have:

- For every $2 \leq a < b < c \leq n$, $\mathsf{o}_{a,b,c}$ is true iff $\sigma(p_a,p_b,p_c) = +1$.[3]
  The first optimization observes that orientations are antisymmetric: if (p,q,r) is counter-clockwise then (q,p,r) is clockwise, etc. Thus one only needs $\mathsf{o}_{a,b,c}$ for ordered triples (a,b,c), reducing the number of orientation variables by a factor of $3! = 6$ relative to using all triples. The second optimization uses the **CCW-order** property of canonical positions: since all $\mathsf{o}_{1,a,b}$ are true, we may as well omit them from the encoding.

[2] Satisfiability of ϕ_n would *not* necessarily imply the existence of a point set without a 6-hole, due to the *realizability problem* (see e.g., [36]).

[3] Since the point set is in general position, we have $\neg\mathsf{o}_{a,b,c} \iff \sigma(p_a,p_b,p_c) = -1$.

$$\mathsf{c}_{i;a,b,c} \to \left((\mathsf{o}_{a,b,c} \leftrightarrow \mathsf{o}_{a,i,c}) \wedge (\mathsf{o}_{a,b,c} \leftrightarrow \overline{\mathsf{o}_{a,i,b}})\right) \text{ for all } 2 \le a < i < b < c \le n \tag{1}$$

$$\mathsf{c}_{i;a,b,c} \to \left((\mathsf{o}_{a,b,c} \leftrightarrow \mathsf{o}_{a,i,c}) \wedge (\mathsf{o}_{a,b,c} \leftrightarrow \overline{\mathsf{o}_{b,i,c}})\right) \text{ for all } 2 \le a < b < i < c \le n \tag{2}$$

$$\Bigg(\bigwedge_{\substack{a<i<c \\ i \neq b}} \overline{\mathsf{c}_{i;a,b,c}} \Bigg) \to \mathsf{h}_{a,b,c} \quad \text{ for all } 2 \le a < b < c \le n \tag{3}$$

$$\mathsf{o}_{a,b,c} \wedge \mathsf{o}_{a,c,d} \to \mathsf{o}_{a,b,d} \quad \text{ for all } 2 \le a < b < c < d \le n \tag{4}$$

$$\overline{\mathsf{o}_{a,b,c}} \wedge \overline{\mathsf{o}_{a,c,d}} \to \overline{\mathsf{o}_{a,b,d}} \quad \text{ for all } 2 \le a < b < c < d \le n \tag{5}$$

$$\left(\mathsf{o}_{\lceil \frac{n}{2} \rceil - 1, \lceil \frac{n}{2} \rceil, \lceil \frac{n}{2} \rceil + 1}, \dots, \mathsf{o}_{2,3,4}\right) \succeq_{\text{lex}} \left(\mathsf{o}_{\lfloor \frac{n}{2} \rfloor + 1, \lfloor \frac{n}{2} \rfloor + 2, \lfloor \frac{n}{2} \rfloor + 3}, \dots, \mathsf{o}_{n-2,n-1,n}\right) \tag{6}$$

$$\overline{\mathsf{o}_{a,b,c}} \wedge \overline{\mathsf{o}_{b,c,d}} \to \mathsf{u}^4_{a,c,d} \quad \text{ for all } 2 \le a < b < c < d \le n \tag{7}$$

$$\mathsf{o}_{a,b,c} \wedge \mathsf{o}_{b,c,d} \to \mathsf{v}^4_{a,c,d} \quad \text{ for all } 2 \le a < b < c < d \le n \tag{8}$$

$$\mathsf{u}^4_{a,b,c} \wedge \overline{\mathsf{o}_{b,c,d}} \wedge \mathsf{h}_{a,b,d} \to \mathsf{u}^5_{a,c,d} \quad \text{ for all } 2 \le a < b < c < d \le n,\ a+1 < b \tag{9}$$

$$\mathsf{u}^4_{a,c,d} \to \overline{\mathsf{o}_{a,c,d}} \quad \text{ for all } 2 \le a < c < d \le n,\ a+1 < c \tag{10}$$

$$\mathsf{v}^4_{a,c,d} \to \mathsf{o}_{a,c,d} \quad \text{ for all } 2 \le a < c < d \le n,\ a+1 < c \tag{11}$$

$$\neg(\mathsf{u}^5_{a,d,e} \wedge \mathsf{o}_{a,p,e}) \quad \text{ for all } 2 \le a < d < e \le n,\ a < p < e,\ a+2 < d \tag{12}$$

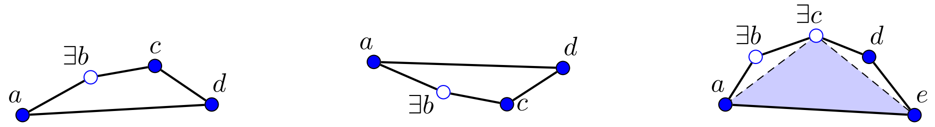
$$\neg(\mathsf{u}^5_{a,d,e} \wedge \overline{\mathsf{o}_{d,e,f}}) \quad \text{ for all } 2 \le a < d < e < f \le n,\ a+2 < d \tag{13}$$

$$\neg(\mathsf{u}^4_{a,c,d} \wedge \mathsf{v}^4_{a,c',d} \wedge \mathsf{h}_{a,c,c'}) \quad \text{ for all } 2 \le a < c < c' < d \le n,\ a+1 < c \tag{14}$$

$$\neg(\mathsf{u}^4_{a,c,d} \wedge \mathsf{v}^4_{a,c',d} \wedge \mathsf{h}_{a,c',c}) \quad \text{ for all } 2 \le a < c' < c < d \le n,\ a+1 < c' \tag{15}$$

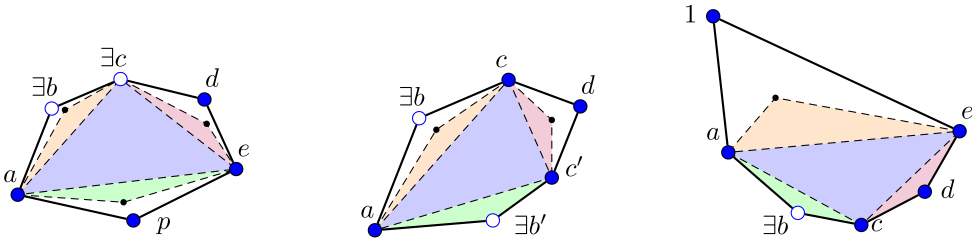
$$\neg(\mathsf{v}^4_{a,c,d} \wedge \mathsf{o}_{c,d,e} \wedge \mathsf{h}_{a,c,e}) \quad \text{ for all } 2 \le a < c < d < e \le n,\ a+1 < c \tag{16}$$

Figure 6 Encoding based on that of Heule and Scheucher for the Empty Hexagon Number [21]. Each line determines a set of clauses. Unsatisfiability of the formula below for $n = 30$ implies $h(6) \le 30$, as detailed throughout the paper.



(a) $\mathsf{u}^4_{a,c,d}$ **(b)** $\mathsf{v}^4_{a,c,d}$ **(c)** $\mathsf{u}^5_{a,d,e}$

Figure 7 Illustration of the 4-cap (7a), 4-cup (7b), and 5-cap (7c) variables. The highlighted region denotes an empty triangle.



(a) $(\mathsf{u}^5_{a,d,e} \land \mathsf{o}_{a,p,e})$ **(b)** $(\mathsf{u}^4_{a,c,d} \land \mathsf{v}^4_{a,c',d} \land \mathsf{h}_{a,c,c'})$ **(c)** $(\mathsf{v}^4_{a,c,d} \land \mathsf{o}_{c,d,e} \land \mathsf{h}_{a,c,e})$

Figure 8 Illustration of some *forbidden configurations* that imply 6-holes. Figure 8a corresponds to the configuration forbidden by clause (12), Figure 8b to the one forbidden by clause (14), and Figure 8c to clause (16). All highlighted regions denote empty triangles.

- Next, for every $a < b < c$ with $a < i < b$ or $b < i < c$, the variable $\mathsf{c}_{i;a,b,c}$ is true iff σ`PtInTriangle S[i] S[a] S[b] S[c]` holds. By σ`PtInTriangle_iff`, this is true exactly iff p_i is inside the triangle $p_a p_b p_c$. The reason for assuming (a, b, c) to be ordered is again symmetry: $p_a p_b p_c$ is the same triangle as $p_a p_c p_b$, etc. Furthermore thanks to the x**-order** property of canonical positions, if p_i is in the triangle then $x(p_a) < x(p_i) < x(p_c)$. This implies that $a < i < c$, leaving one case distinction permuting (i, b).
- For every $a < b < c$, $\mathsf{h}_{a,b,c}$ is true iff σ`IsEmptyTriangleFor S[a] S[b] S[c] S` holds. By a geometro-combinatorial connection analogous to ones above, this is true iff $p_a p_b p_c$ is a 3-hole.
- Finally, one defines 4-*cap*, 5-*cap*, and 4-*cup* variables. For $a + 1 < c < d$, $\mathsf{u}^4_{a,c,d}$ is true iff there is b with $a < b < c$ with $\sigma(p_a, p_b, p_c) = \sigma(p_b, p_c, p_d) = -1$. $\mathsf{v}^4_{a,c,d}$ is analogous, except in that the two orientations are required to be counterclockwise. These are the 4-caps and 4-cups, respectively. The 5-cap variables $\mathsf{u}^5_{a,d,e}$ are defined for $a + 2 < d < e$. We set $\mathsf{u}^5_{a,d,e}$ to true iff there exists c with $a + 1 < c < d$ such that $\mathsf{u}^4_{a,c,d}$, $\mathsf{o}_{c,d,e}$, and $\mathsf{h}_{a,c,e}$ are all true. Intuitively, 4-caps and 4-cups are clockwise and counterclockwise arcs of length 4, respectively, whereas 5-caps are clockwise arcs of length 5 containing a 3-hole. All three are depicted in Figure 7. The usage of these variables is crucial to an efficient encoding: we will show below that a hexagon can be covered by only 4 triangles, so one need not consider all $\binom{6}{3}$ triangles contained within it.

Satisfaction. We now have to justify that the clauses of ϕ_n are satisfied by the intended interpretation τ_S for a 6-hole-free point set S. The variable-defining clauses (1)–(3) and (7)–(11) follow essentially by definition combined with boolean reasoning. The orientation properties (4) and (5) have been established in the family of theorems σ`_prop`$_i$. The lexicographic ordering clauses (6) follow from the **Lex order** property of canonical positions. Thus we are

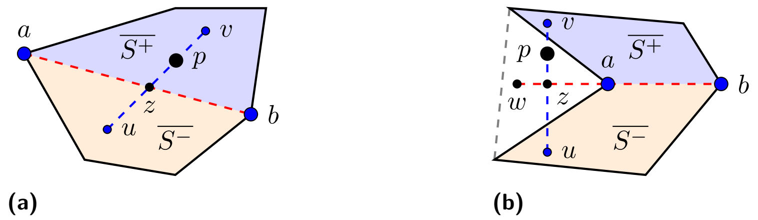


Figure 9 Illustration of the proof for `split_convexHull`. (a) Given point p, we obtain points u and v inside the two halves and z as the point of intersection with the line $\overline{ab}$. (b) In this (contradictory) situation, the point z has ended up outside the segment $\overline{ab}$, because S is not actually convex. In this case we construct w such that z is on the $\overline{wa}$ segment, and observe that w, z, a, b are collinear.

left with clauses (12)–(16) which forbid the presence of certain 6-holes.[4] We illustrate why clause (12) is true. The contrapositive is easier to state: if τ_S satisfies $\mathsf{u}^5_{a,d,e} \wedge \mathsf{o}_{a,p,e}$, then S contains a 6-hole. The intuitive argument is depicted in Figure 8a. The clause directly implies the existence of a convex hexagon *apedcb* such that *ace* is a 3-hole. It turns out that this is enough to ensure the existence of a 6-hole by "flattening" the triangles *ape*, *edc*, and *cba*, if necessary, to obtain empty triangles $ap'e$, $ed'c$, and $cb'a$, which can be assembled into a 6-hole $ap'ed'cb'$.

Justifying this formally turned out to be complex, requiring a fair bit of reasoning about point `Arcs` and `σCCWPoints`: lists of points winding around a convex polygon. Luckily, the main argument can be summarized in terms of two facts: (a) any triangle *abc* contains an empty triangle $ab'c$; and (b) empty shapes sharing a common line segment can be glued together. Formally, (a) can be stated as

```
theorem σIsEmptyTriangleFor_exists (gp : ListInGenPos S)
  (abc : [a, b, c] ⊆ S) : ∃ b' ∈ S, σ a b' c = σ a b c
    ∧ (b' = b ∨ σPtInTriangle b' a b c) ∧ σIsEmptyTriangleFor a b' c S.toFinset
```

Proof. Given points p, q, say that $p \leq q$ iff p is in the triangle *aqc*. This is a preorder. Now, the set $S' = \{x \in S \mid \sigma(a, x, c) = \sigma(a, b, c) \wedge x \leq b\}$ is finite and so has a weakly minimal element b', in the sense that no $x \in S'$ has $x < b'$. Emptiness of $ab'c$ follows by minimality. ◀

Moving on, (b) follows from a *triangulation lemma*: given any convex point set S and a line $\overleftrightarrow{ab}$ between two vertices of S, the convex hull of S is contained in the convex hulls of points on either side of $\overleftrightarrow{ab}$. That is:

```
theorem split_convexHull (cvx : ConvexIndep S) :
  ∀ {a b}, a ∈ S → b ∈ S →
    convexHull ℝ S ⊆ convexHull ℝ {x ∈ S | σ a b x ≠ ccw}
                   ∪ convexHull ℝ {x ∈ S | σ a b x ≠ cw}
```

Proof. Let $S^+ = \{x \in S \mid \sigma(a, b, x) \geq 0\}$ and $S^- = \{x \in S \mid \sigma(a, b, x) \leq 0\}$ be the two sets in the theorem, and let $p \in \overline{S}$, where $\overline{S}$ denotes the convex hull of S. Assume WLOG that $\sigma(a, b, p) \geq 0$. (We would like to show that $p \in \overline{S^+}$.) Now p is a convex combination

[4] They are intended to forbid *all* 6-holes, but proving completeness is not necessary for an unsatisfiability-based result.

of elements of S^+ and elements of S^-, so there exist points $u \in \overline{S^-}$ and $v \in \overline{S^+}$ such that p lies on the $\overline{uv}$ line. Because $\{x \mid \det(a,b,x) \le 0\} \supseteq S^-$ is convex, it follows that $\det(a,b,u) \le 0$, and likewise $\det(a,b,v) \ge 0$, so they lie on opposite sides of the $\overleftrightarrow{ab}$ line and hence $\overline{uv}$ intersects $\overleftrightarrow{ab}$ at a point z. The key point is that z must in fact be on the line segment $\overline{ab}$; assuming that this was the case, we could obtain z as a convex combination of a and b, and p as a convex combination of v and z, and since v is in $\overline{S^+}$ and $a, b \in S^+ \subseteq \overline{S^+}$ we can conclude $p \in \overline{S^+}$. To show that $z \in \overline{ab}$, suppose not, so that a lies between z and b (see Figure 9b). (The case where z is on the b side is similar.) We can decompose z as a convex combination of some $w \in \overline{S \setminus \{a\}}$ and a, which means that w, z, a, b are collinear and appear in this order on the line. Therefore a is a convex combination of w and b, which means that $a \in \overline{S \setminus \{a\}}$ which violates convexity of S. ◀

By contraposition, the triangulation lemma directly implies that if $\{x \in S \mid \sigma(a,b,x) \neq +1\}$ and $\{x \in S \mid \sigma(a,b,x) \neq -1\}$ are both empty shapes in P, then S is an empty shape in P.

6.1 Running the CNF.

Having now shown that our main result follows if ϕ_{30} is unsatisfiable, we run a distributed computation to check its unsatisfiability. We solve the SAT formula ϕ_{30} produced by Lean using the same setup as Heule and Scheucher [21], although using different hardware: the Bridges 2 cluster of the Pittsburgh Supercomputing Center [4]. Following Heule and Scheucher, we partition the problem into 312 418 subproblems. Each of these subproblems was solved using `CaDiCaL` version 1.9.5. `CaDiCaL` produced an LRAT proof for each execution, which was validated using the `cake_lpr` verified checker on-the-fly in order to avoid writing/storing/reading large files. The total runtime was 25 876.5 CPU hours, or roughly 3 CPU years. The difference in runtime relative to Heule and Scheucher's original run is purely due to the difference in hardware. Additionally, we validated that the subproblems cover the entire search space as Heule and Scheucher did [21, Section 7.3]. This was done by verifying the unsatisfiability of another formula that took 20 seconds to solve.

The artifact for this paper includes scripts to validate any individual subproblem, as well as the summary proof that the subproblems cover the search space. However, the unsatisfiability of ϕ_{30} depends on the unsatisfiability of *all* (hundreds of thousands of) subproblems. A skeptical reader might wish to examine the proof files for all subproblems, but we estimated the total proof size to be tens or hundreds of terabytes, far too much to reasonably store and distribute. Instead, the skeptical reader must run the entire 3 CPU year computation. We believe this trust story can be somewhat improved, but we leave such a challenge to future work.

7 Related Work

Our formalization is closely related to a prior development in which Marić put proofs of $g(6) \le 17$ on a more solid foundation [28]. The inequality, originally obtained by Szekeres and Peters [39] using a specialized, unverified search algorithm, was confirmed by Marić using a formally-verified SAT encoding. Marić introduced an optimized encoding based on nested convex hull structures, which, when combined with performance advances in modern SAT solvers, significantly improved the search time over the unverified computation.

Our work focuses on the closely-related problem of determining k-hole numbers $h(k)$. Rather than devise a new SAT encoding, we use essentially the same encoding presented by Heule and Scheucher [21]. Interestingly, a formal proof of $g(6) \le 17$ can be obtained as a corollary of our development. We can assert the hole variables $\mathsf{h}_{a,b,c}$ as true while leaving the remainder of the encoding in Section 6 unchanged, which trivializes constraints about emptiness so that only the convexity constraints remain.[5] The resulting CNF formula asserts the existence of a set of n points with no convex 6-gon. We checked this formula to be unsatisfiable for $n = 17$, giving the same result as Marić:

```
theorem gon_6_theorem : ∀ (pts : Finset Point),
  SetInGenPos pts → pts.card = 17 → HasConvexKGon 6 pts
```

Since both formalizations can be executed, we performed a direct comparison against Marić's encoding. On a personal laptop, we found that it takes negligible time (below 1s) for our verified Lean encoder to output the full CNF. In contrast, Marić's encoder, extracted from Isabelle/HOL code,[6] took 437s to output a CNF. To improve encoding performance, Marić wrote a C++ encoder whose code was manually compared against the Isabelle/HOL specification. We do not need to resort to an unverified implementation.

As for the encodings, ours took 28s to solve, while the Marić encoding took 787s (both using `cadical`). This difference is likely accounted for by the relative size of the encodings, in particular their symmetry breaking strategies. For $k = 6$ and n points, the encoding of Heule and Scheucher uses $O(n^4)$ clauses, whereas the one of Marić uses $O(n^6)$ clauses. They are based on different ideas: the former as detailed in Section 5, whereas the latter on nested convex hulls. The different approaches have been discussed by Scheucher [32]. This progress in solve times represents an encouraging state of affairs; we are optimistic that if continued, it could lead to an eventual resolution of $g(7)$.

Further differences include what exactly was formally proven. As with most work in this area, we use the combinatorial abstraction of triple orientations. We and Marić alike show that point sets in $\mathbb{R}^2$ satisfy orientation properties (Section 4). However, our work goes further in building the connection between geometry and combinatorics: our definitions of convexity and emptiness (Section 3), and consequently the theorem statements, are geometric ones based on convex hulls as defined in Lean's `mathlib` [29]. In contrast, Marić axiomatizes these properties in terms of σ. A skeptical reviewer must manually verify that these combinatorial definitions correspond to the desired geometric concept.

A final point of difference concerns the verification of SAT proofs. Marić fully reconstructs some of the SAT proofs on which his results depend, though not the main one for $g(6)$, in an NbE-based proof checker for Isabelle/HOL. We make no such attempt for the time being, instead passing our SAT proofs through the formally verified proof checker `cake_lpr` [40] and asserting unsatisfiability of the CNF as an axiom in Lean. Thus we trust that the CNF formula produced by the verified Lean encoder is the same one whose unsatisfiability was checked by cake_lpr.

8 Concluding Remarks

We have proved the correctness of the main result of Heule and Scheucher [21], implying $h(6) \leq 30$. Given that the lower bound $h(6) > 29$ can be checked directly (see [21]), we conclude the result $h(6) = 30$ is indeed correct. We believe this work puts a *happy ending* to one line of research started by Klein, Erdős and Szekeres in the 1930s. Prior to formalization,

[5] This modification was performed by an author who did not understand this part of the proof, nevertheless having full confidence in its correctness thanks to the Lean kernel having checked every assertion.

[6] We used Isabelle/HOL 2016. Porting the encoder to more recent versions of the prover would require broader adaptations due to breaking changes in the HOL theories.

the result of Heule and Scheucher relied on the correctness of various components of a highly sophisticated encoding that are hard to validate manually. We developed a significant theory of combinatorial geometry that was not present in mathlib. Beyond the main theorem presented here, we showed how our framework can be used for other related theorems such as $g(6) = 17$, and we hope it can be used for proving many further results in the area.

Our formalization required approximately 300 hours of work over 3 months by researchers with significant experience formalizing mathematics in Lean. The final version of our proofs consists of approximately 4.7k lines of Lean code; about 26% are lemmas that should be moved to upstream libraries, about 40% develops the theory of orientations in plane geometry, and the remaining 34% (1550 LOC) validates the symmetry breaking and SAT encoding.

We substantially simplified the symmetry-breaking argument presented by Heule and Scheucher, and derived in turn from Scheucher [32]. Moreover, we found a small error in their proof, as their transformation uses the mapping $(x, y) \mapsto (x/y, -1/y)$, and incorrectly assumes that x/y is increasing for points in CCW-order, whereas only the slopes y/x are increasing. Similarly, we found a typo in the statement of the Lex order condition that did not match the (correct) code of Heule and Scheucher. Our formalization corrects this.

Future Work We hope to formally verify the result $h(7) = \infty$ due to Horton [23], and other results in Erdős-Szekeres style problems.

We also want to improve the trust story of importing "cube and conquer"-style results into an ITP. Importing these kinds of proofs is a significant engineering task when the proof certificate is hundreds of terabytes in size, as it was for this result (see Section 6.1). Although we are confident that our results are correct, more work needs to be done to strengthen the trust in this connection point.

References

1 A. Biere, M. Heule, H. van Maaren, and T. Walsh. *Handbook of Satisfiability: Volume 185 Frontiers in Artificial Intelligence and Applications*. IOS Press, NLD, 2009.

2 Joshua Brakensiek, Marijn Heule, John Mackey, and David Narváez. The Resolution of Keller's Conjecture, 2023. arXiv:1910.03740.

3 Curtis Bright, Kevin K. H. Cheung, Brett Stevens, Ilias S. Kotsireas, and Vijay Ganesh. A SAT-based resolution of Lam's Problem. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021*, pages 3669–3676. AAAI Press, 2021. URL: https://doi.org/10.1609/aaai.v35i5.16483, doi:10.1609/AAAI.V35I5.16483.

4 Shawn T. Brown, Paola Buitrago, Edward Hanna, Sergiu Sanielevici, Robin Scibek, and Nicholas A. Nystrom. *Bridges-2: A Platform for Rapidly-Evolving and Data Intensive Research*, pages 1–4. Association for Computing Machinery, New York, NY, USA, 2021.

5 Davide Castelvecchi. Mathematicians welcome computer-assisted proof in 'grand unification' theory. *Nature*, 595(7865):18–19, June 2021. URL: http://dx.doi.org/10.1038/d41586-021-01627-2, doi:10.1038/d41586-021-01627-2.

6 Cayden Codel, Marijn J. H. Heule, and Jeremy Avigad. Verified Encodings for SAT Solvers. In Alexander Nadel and Kristin Yvonne Rozier, editors, *Proceedings of the 23rd conference on Formal Methods In Computer-Aided Design*, 2023.

7 James Crawford, Matthew Ginsberg, Eugene Luks, and Amitabha Roy. Symmetry-breaking predicates for search problems. In *Proc. KR'96, 5th Int. Conf. on Knowledge Representation and Reasoning*, pages 148–159. Morgan Kaufmann, 1996.

8 Luís Cruz-Filipe, João Marques-Silva, and Peter Schneider-Kamp. Formally Verifying the Solution to the Boolean Pythagorean Triples Problem. *J. Autom. Reason.*, 63(3):695–722, oct 2019. doi:10.1007/s10817-018-9490-4.

9 Luís Cruz-Filipe and Peter Schneider-Kamp. Formally Proving the Boolean Pythagorean Triples Conjecture. In Thomas Eiter and David Sands, editors, *LPAR-21. 21st International Conference on Logic for Programming, Artificial Intelligence and Reasoning*, volume 46 of *EPiC Series in Computing*, pages 509–522. EasyChair, 2017. URL: https://easychair.org/publications/paper/xq6J, doi:10.29007/jvdj.

10 Leonardo de Moura, Soonho Kong, Jeremy Avigad, Floris van Doorn, and Jakob von Raumer. The Lean Theorem Prover (System Description). In Amy P. Felty and Aart Middeldorp, editors, *Automated Deduction - CADE-25*, pages 378–388, Cham, 2015. Springer International Publishing.

11 Théo Delemazure, Tom Demeulemeester, Manuel Eberl, Jonas Israel, and Patrick Lederer. Strategyproofness and proportionality in party-approval multiwinner elections. In Brian Williams, Yiling Chen, and Jennifer Neville, editors, *Thirty-Seventh AAAI Conference on Artificial Intelligence, AAAI 2023, Thirty-Fifth Conference on Innovative Applications of Artificial Intelligence, IAAI 2023, Thirteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2023, Washington, DC, USA, February 7-14, 2023*, pages 5591–5599. AAAI Press, 2023. URL: https://doi.org/10.1609/aaai.v37i5.25694, doi:10.1609/AAAI.V37I5.25694.

12 Paul Erdős and George Szekeres. On some extremum problems in elementary geometry. *Ann. Univ. Sci. Budapest. Eötvös Sect. Math.*, 3(4):53–62, 1960.

13 Paul Erdős and György Szekeres. A combinatorial problem in geometry. *Compositio Mathematica*, 2:463–470, 1935. URL: http://eudml.org/doc/88611.

14 Stefan Felsner and Helmut Weil. Sweeps, arrangements and signotopes. *Discrete Applied Mathematics*, 109(1):67–94, April 2001. doi:10.1016/S0166-218X(00)00232-8.

15 Tobias Gerken. Empty Convex Hexagons in Planar Point Sets. *Discrete & Computational Geometry*, 39(1):239–272, mar 2008. doi:10.1007/s00454-007-9018-x.

16 Sofia Giljegård and Johan Wennerbeck. Puzzle Solving with Proof. Master's thesis, Chalmers University of Technology, 2021.

17 W. T. Gowers, Ben Green, Freddie Manners, and Terence Tao. On a conjecture of Marton, 2023. arXiv:2311.05762.

18 Andrew Haberlandt, Harrison Green, and Marijn J. H. Heule. Effective auxiliary variables via structured reencoding. In *26th International Conference on Theory and Applications of Satisfiability Testing (SAT 2023)*, volume 271 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 11:1–11:19, Dagstuhl, Germany, 2023. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. URL: https://drops.dagstuhl.de/entities/document/10.4230/LIPIcs.SAT.2023.11, doi:10.4230/LIPIcs.SAT.2023.11.

19 Heiko Harborth. Konvexe Fünfecke in ebenen Punktmengen. *Elemente der Mathematik*, 33:116–118, 1978. URL: http://eudml.org/doc/141217.

20 Marijn J. H. Heule, Oliver Kullmann, and Victor W. Marek. *Solving and Verifying the Boolean Pythagorean Triples Problem via Cube-and-Conquer*, page 228–245. Springer International Publishing, 2016. URL: http://dx.doi.org/10.1007/978-3-319-40970-2_15, doi:10.1007/978-3-319-40970-2_15.

21 Marijn J. H. Heule and Manfred Scheucher. Happy ending: An empty hexagon in every set of 30 points, 2024. arXiv:2403.00737.

22 Andreas F Holmsen, Hossein Nassajian Mojarrad, János Pach, and Gábor Tardos. Two extensions of the erdős-szekeres problem. *arXiv preprint arXiv:1710.11415*, 2017.

23 J. D. Horton. Sets with No Empty Convex 7-Gons. *Canadian Mathematical Bulletin*, 26(4):482–484, 1983. doi:10.4153/CMB-1983-077-8.

24 Donald E. Knuth. Axioms and Hulls. In Donald E. Knuth, editor, *Axioms and Hulls*, Lecture Notes in Computer Science, pages 1–98. Springer, Berlin, Heidelberg, 1992. doi:10.1007/3-540-55611-7_1.

25 Boris Konev and Alexei Lisitsa. A SAT Attack on the Erdos Discrepancy Conjecture, 2014. arXiv:1402.2184.

26 Peter Lammich. Efficient Verified (UN)SAT Certificate Checking. *Journal of Automated Reasoning*, 64(3):513–532, March 2020. doi:10.1007/s10817-019-09525-z.

27 Filip Marić. Formal verification of a modern SAT solver by shallow embedding into Isabelle/HOL. *Theor. Comput. Sci.*, 411(50):4333–4356, 2010. URL: https://doi.org/10.1016/j.tcs.2010.09.014, doi:10.1016/J.TCS.2010.09.014.

28 Filip Marić. Fast formal proof of the Erdős-Szekeres conjecture for convex polygons with at most 6 points. *J. Autom. Reason.*, 62(3):301–329, 2019. URL: https://doi.org/10.1007/s10817-017-9423-7, doi:10.1007/S10817-017-9423-7.

29 The mathlib Community. The Lean mathematical library. In *Proceedings of the 9th ACM SIGPLAN International Conference on Certified Programs and Proofs*, POPL '20. ACM, January 2020. URL: http://dx.doi.org/10.1145/3372885.3373824, doi:10.1145/3372885.3373824.

30 Carlos M. Nicolas. The Empty Hexagon Theorem. *Discrete & Computational Geometry*, 38(2):389–397, September 2007. doi:10.1007/s00454-007-1343-6.

31 Duckki Oe, Aaron Stump, Corey Oliver, and Kevin Clancy. Versat: A Verified Modern SAT Solver. In Viktor Kuncak and Andrey Rybalchenko, editors, *Verification, Model Checking, and Abstract Interpretation*, pages 363–378, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

32 Manfred Scheucher. Two disjoint 5-holes in point sets. *Computational Geometry*, 91:101670, December 2020. doi:10.1016/j.comgeo.2020.101670.

33 Sarek Høverstad Skotåm. CreuSAT, Using Rust and Creusot to create the world's fastest deductively verified SAT solver. Master's thesis, University of Oslo, 2022. URL: https://www.duo.uio.no/handle/10852/96757.

34 Leila Sloman. 'A-Team' of Math Proves a Critical Link Between Addition and Sets. https://www.quantamagazine.org/a-team-of-math-proves-a-critical-link-between-addition-and-sets-20231206/, December 2023.

35 Bernardo Subercaseaux and Marijn J. H. Heule. The Packing Chromatic Number of the Infinite Square Grid is 15. In Sriram Sankaranarayanan and Natasha Sharygina, editors, *Tools and Algorithms for the Construction and Analysis of Systems - 29th International Conference, TACAS 2023, Held as Part of ETAPS 2022, Proceedings, Part I*, volume 13993 of *Lecture Notes in Computer Science*, page 389–406. Springer, 2023. doi:10.1007/978-3-031-30823-9_20.

36 Bernardo Subercaseaux, John Mackey, Marijn J. H. Heule, and Ruben Martins. Minimizing pentagons in the plane through automated reasoning, 2023. arXiv:2311.03645.

37 Andrew Suk. On the erdős-szekeres convex polygon problem. *Journal of the American Mathematical Society*, 30(4):1047–1053, 2017.

38 George Szekeres and Lindsay Peters. Computer solution to the 17-point Erdős-Szekeres problem. *The ANZIAM Journal*, 48(2):151–164, 2006. doi:10.1017/S144618110000300X.

39 George Szekeres and Lindsay Peters. Computer solution to the 17-point erdős-szekeres problem. *The ANZIAM Journal*, 48(2):151–164, 2006.

40 Yong Kiam Tan, Marijn J. H. Heule, and Magnus O. Myreen. Verified Propagation Redundancy and Compositional UNSAT Checking in CakeML. *International Journal on Software Tools for Technology Transfer*, 25(2):167–184, April 2023. doi:10.1007/s10009-022-00690-y.

41 Mark Walters. It Appears That Four Colors Suffice : A Historical Overview of the Four-Color Theorem. 2004. URL: https://api.semanticscholar.org/CorpusID:14382286.

42 Nathan Wetzler, Marijn J. H. Heule, and Warren A. Hunt. DRAT-trim: Efficient checking and trimming using expressive clausal proofs. In Carsten Sinz and Uwe Egly, editors, *Theory and Applications of Satisfiability Testing – SAT 2014*, pages 422–429, Cham, 2014. Springer International Publishing.