Automated Reasoning in Discrete Geometry: Discovery, Verification, and Symmetry

Marijn J.H. Heule

Carnegie Mellon University

scholars

ModRef 2025 in Glasgow August 11, 2025

`sat4math.com`

50 Years of Successes in Computer-Aided Mathematics

- 1976 Four-Color Theorem
- 1998 Kepler Conjecture
- 2014 Boolean Erdős discrepancy problem
- 2016 Boolean Pythagorean triples problem
- 2018 Schur Number Five
- 2019 Keller's Conjecture
- 2021 Kaplansky's Unit Conjecture
- 2022 Packing Number of Square Grid
- 2023 Empty Hexagon in Every 30 Points

50 Years of Successes in Computer-Aided Mathematics

1976 Four-Color Theorem

1998 Kepler Conjecture

2014 Boolean Erdős discrepancy problem (using a SAT solver)

2016 Boolean Pythagorean triples problem (using a SAT solver)

2018 Schur Number Five (using a SAT solver)

2019 Keller's Conjecture (using a SAT solver)

2021 Kaplansky's Unit Conjecture (using a SAT solver)

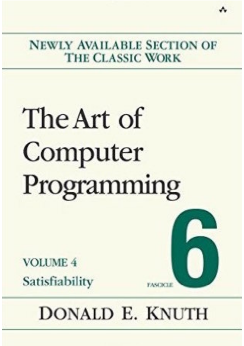
2022 Packing Number of Square Grid (using a SAT solver)

2023 Empty Hexagon in Every 30 Points (using a SAT solver)

Breakthrough in SAT Solving in the Last 30 Years

Satisfiability (SAT) problem: Can a Boolean formula be satisfied?

mid '90s: formulas solvable with thousands of variables and clauses
now: formulas solvable with millions of variables and clauses



Edmund Clarke: *"a key technology of the 21st century"* [Biere, Heule, vanMaaren, Walsh '09/'21]

Donald Knuth: *"evidently a killer app, because it is key to the solution of so many other problems"* [Knuth '15]

Introduction: Abstraction

Not all constraints are easy to encode into propositional logic

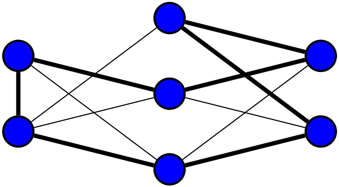
- Abstraction and refinement
- Underapproximation
- Satisfiability modulo theories

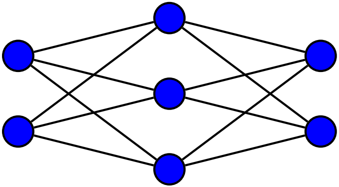
Solution: Only encode a subset of the problem

- Skip the constraints that are hard to encode
- If the subset is UNSAT, the full problem is UNSAT
- If an assignment that satisfies the subset also satisfies the full problem, then SAT
- Otherwise extend the subset (aka refinement)

Hamiltonian Cycles: Two Constraints

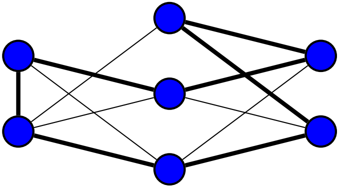
Hamiltonian Cycle Problem (HCP):
Does there exist a cycle that visits all vertices exactly once?

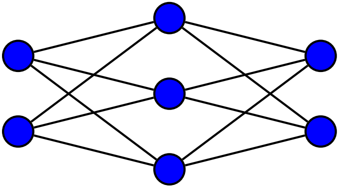




Hamiltonian Cycles: Two Constraints

Hamiltonian Cycle Problem (HCP):
Does there exist a cycle that visits all vertices exactly once?





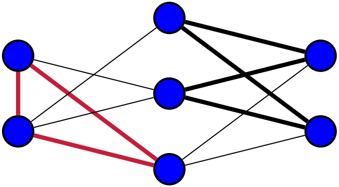
Two constraints:

- Exactly two edges per vertex: easy cardinality constraints
- Exactly one cycle: hard to be compact and arc-consistent
  - One option is to ignore the constraint: incremental SAT.
  - Various encodings use $O(|V|^3)$. Too large for many graphs.
  - Effective encodings are quasi-linear in the number of edges.

Hamiltonian Cycles: Refinement

Only encode: Exactly two edges per vertex

- Problem: Solutions can consist of multiple cycles
- How to implement refinement for a multi-cycle solution?

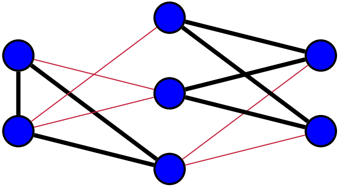


Block at least one subcycle

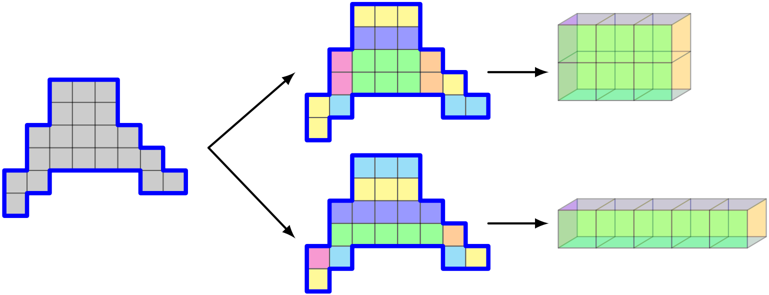
- E.g., block the smallest cycle
- Only a small number of cycles need to be blocked in practice

Constrain the cut edges

- At least 2 cut edges required
- Subcycles are an effective heuristic to pick the cut



Common Unfolding Multiple Boxes



(Un)folding boxes along unit lines of polyominoes only

- Earlier works (non-SAT): Area ~ 90
- Earlier works (SAT full encoding): Area ~ 40
- Our encoding (SAT abstraction): Area ~ 180

Common Unfolidng using Local Constraints [CADE'25]

1. Encode the existence of unfoldings as SAT formulas
2. Use efficient (local) under-approximations for encodings
3. UNSAT $\rightarrow$ no unfoldings exist
4. SAT $\rightarrow$ check satisfying assignments

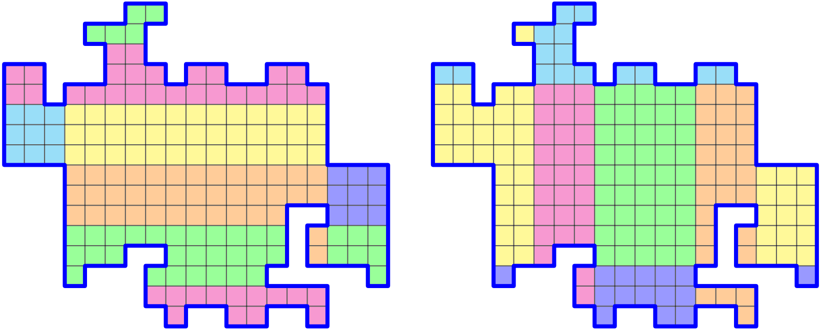
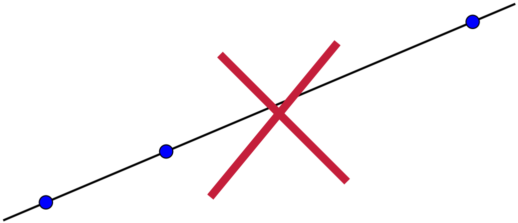


Figure: Common unfolding of $3\times3\times13$ and $3\times5\times9$

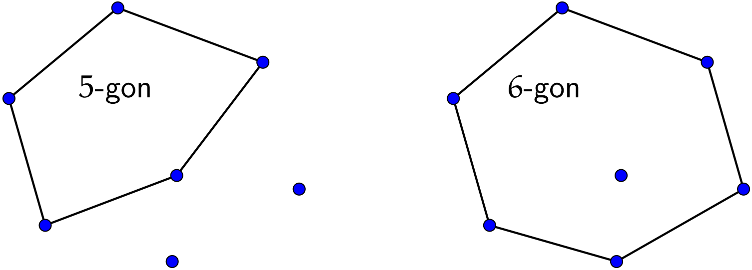
Points in General Position

A finite point set S in the plane is in general position if no three points in S are on a line.



Erdős–Szekeres Numbers

A k-gon (in S) is the vertex set of a convex k-gon

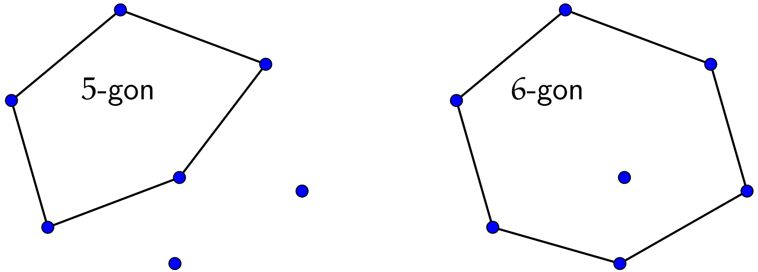


Theorem (Erdős & Szekeres 1935)

$\forall k \in \mathbb{N}, \exists$ *a smallest integer* $g(k)$ *such that every set of* $g(k)$ *points in general position contains a* k*-gon.*

Erdős–Szekeres Numbers

A k-gon (in S) is the vertex set of a convex k-gon



Theorem (Erdős & Szekeres 1935)

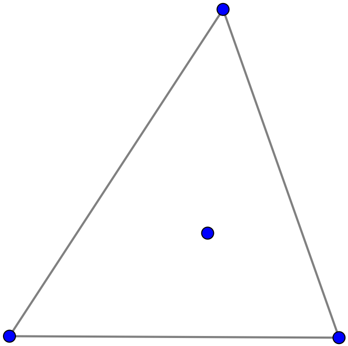
$\forall k \in \mathbb{N}, \exists$ *a smallest integer* $g(k)$ *such that every set of* $g(k)$ *points in general position contains a* k*-gon.*

Is SAT solving suitable to answer such questions? Yes!

Bounds for Small k

Clearly, it takes exactly three points in general position to have a 3-gon (triangle)

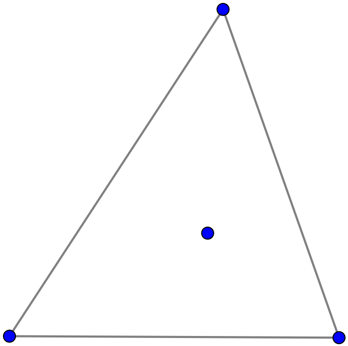
Some sets of 4 points do not for a 4-gon:



Bounds for Small k

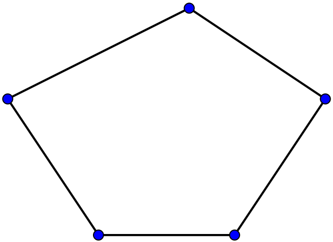
Clearly, it takes exactly three points in general position to have a 3-gon (triangle)

Some sets of 4 points do not for a 4-gon:

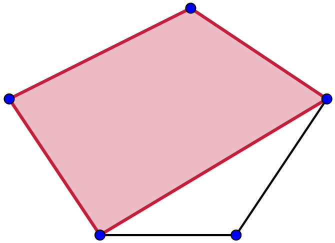


How many points imply a 4-gon?

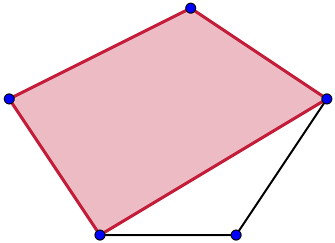
Upperbound for 4-Gon: $g(4) = 5$ [Klein, 1932]

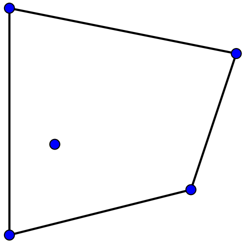


Upperbound for 4-Gon: $g(4) = 5$ [Klein, 1932]

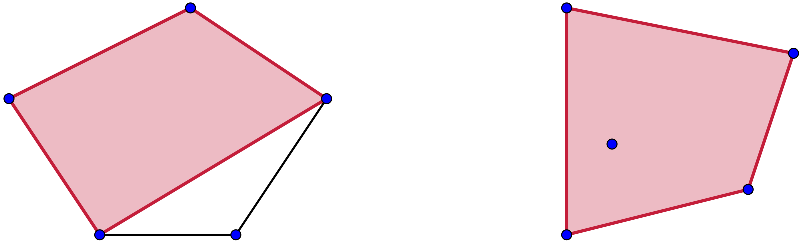


Upperbound for 4-Gon: $g(4) = 5$ [Klein, 1932]

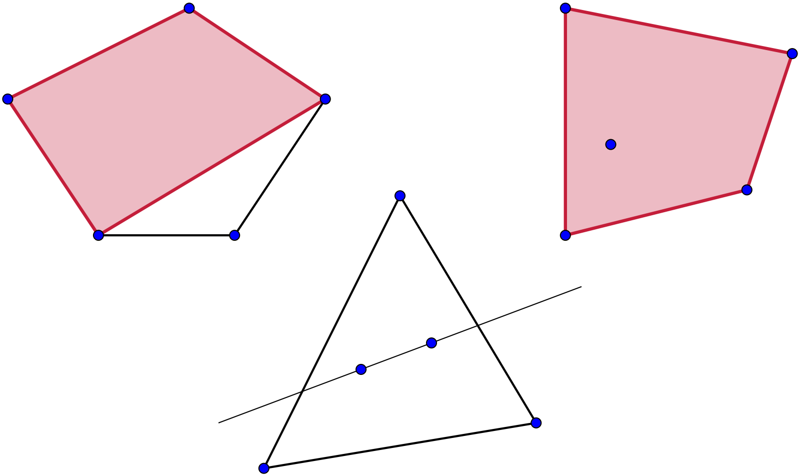




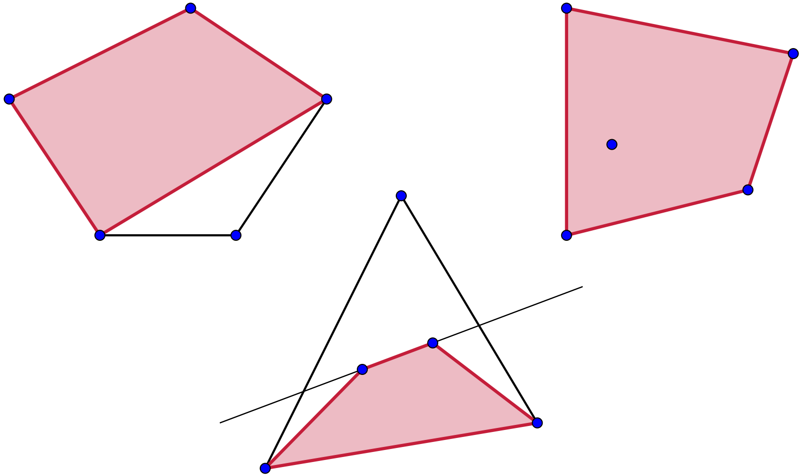
Upperbound for 4-Gon: $g(4) = 5$ [Klein, 1932]



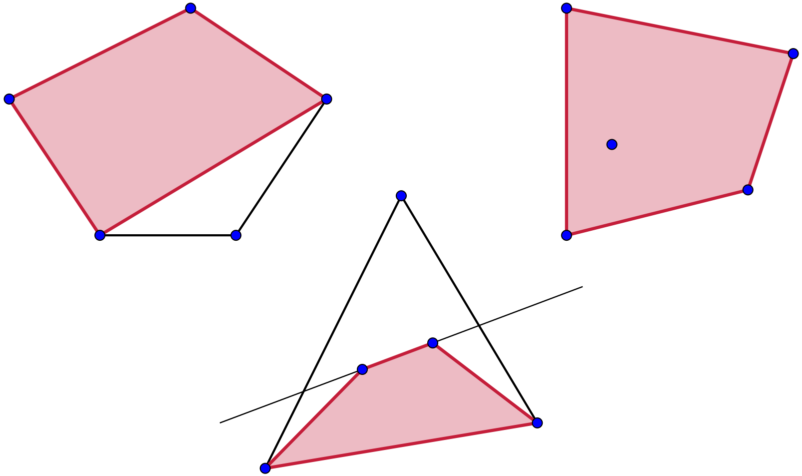
Upperbound for 4-Gon: $g(4) = 5$ [Klein, 1932]



Upperbound for 4-Gon: $g(4) = 5$ [Klein, 1932]

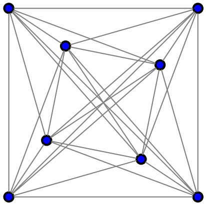


Upperbound for 4-Gon: $g(4) = 5$ [Klein, 1932]



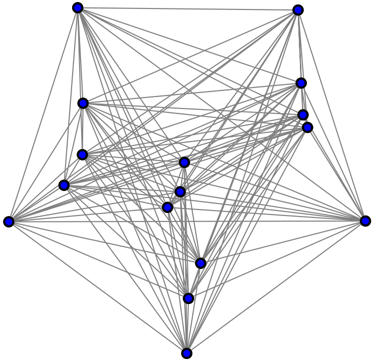
Happy ending problem

Bound Results for 5-Gon and 6-Gon



$g(5) = 9$

- [Kalbfleisch & Stanton '70]

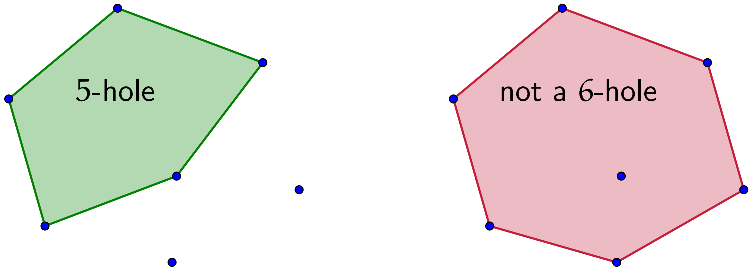


$g(6) = 17$

- Computer-assisted proof, 1500 CPU hours [SzekeresPeters '06]
- One CPU hour using a SAT solver [Scheucher '18]
- Only 10 seconds using new encoding

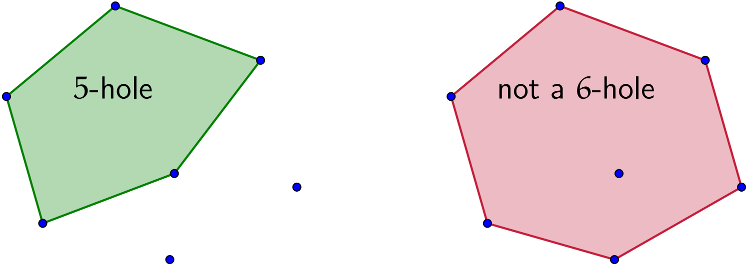
k-Holes

A k-hole (in S) is a k-gon containing no other points of S.



k-Holes

A k-hole (in S) is a k-gon containing no other points of S.



Let $h(k)$ denote the smallest number of points that contain a k-hole.

Erdős, 1970's: For k fixed, does every sufficiently large point set contain k-holes?

k-Holes Overview

A k-hole (in S) is a k-gon containing no other points of S.

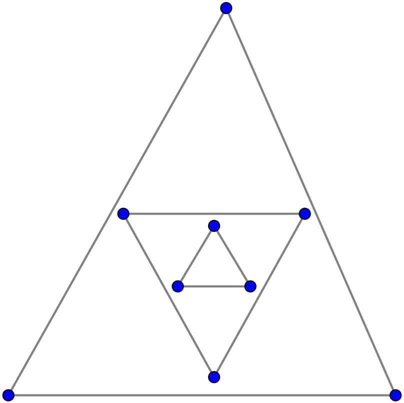
Erdős, 1970's: For k fixed, does every sufficiently large point set contain k-holes?

- 3 points $\Rightarrow \exists$ 3-hole
- 5 points $\Rightarrow \exists$ 4-hole
- 10 points $\Rightarrow \exists$ 5-hole [Harborth '78]
- Arbitrarily large point sets with no 7-hole [Horton '83]

Main open question: what about 6-hole?

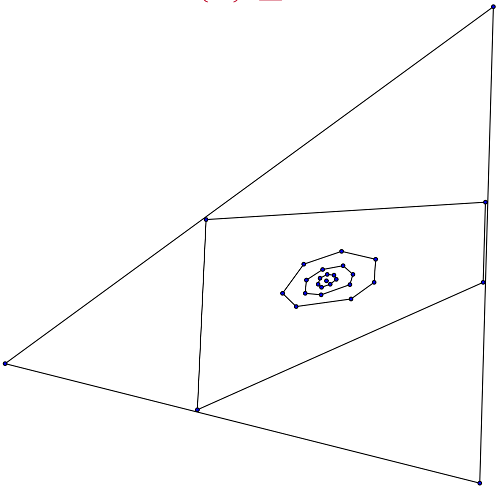
- Lower bound of 30 [Overmars '02]
- Sufficiently large point sets contain a 6-hole [Gerken '08 and Nicolás '07, independently]

Lowerbound for 5-Hole: $h(5) \geq 10$



All 5-gons in these 9 points have an inner point: $h(5) = 10$

Lowerbound for 6-Hole: $h(6) \geq 30$



29 points, no 6-hole [Overmars '02]

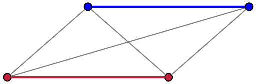
- Found using simulated annealing... is now easy using SAT
- This contains 7-gons. Each 9-gon contains a 6-hole

No Lowerbound for 7-Hole: Horton's Construction

2^1 points, no 7-hole

No Lowerbound for 7-Hole: Horton's Construction

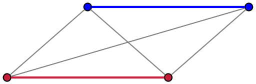
2^1 points, no 7-hole



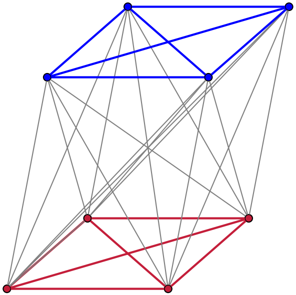
2^2 points, no 7-hole

No Lowerbound for 7-Hole: Horton's Construction

2^1 points, no 7-hole



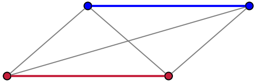
2^2 points, no 7-hole



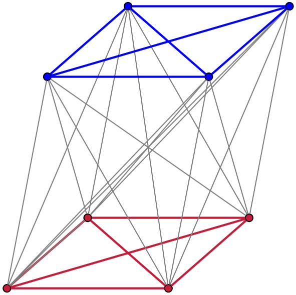
2^3 points, no 7-hole

No Lowerbound for 7-Hole: Horton's Construction

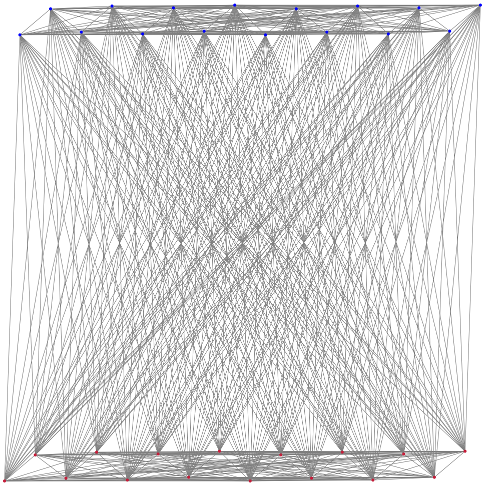
2^1 points, no 7-hole



2^2 points, no 7-hole



2^3 points, no 7-hole



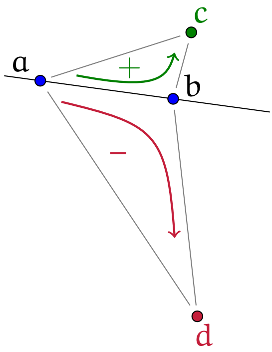
2^5 points, no 7-hole

Orientation Variables [Knuth '92]

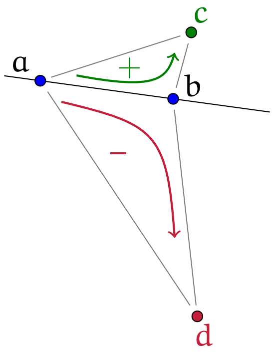
No explicit coordinates of points

Instead for every triple $a < b < c$, one orientation variable O_{abc} to denote whether point c is above the line ab

Triple orientations are enough to express k-gons and k-holes



Orientation Variables [Knuth '92]



No explicit coordinates of points

Instead for every triple $a < b < c$, one orientation variable O_{abc} to denote whether point c is above the line ab

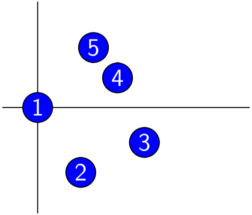
Triple orientations are enough to express k-gons and k-holes

WLOG points are sorted from left to right

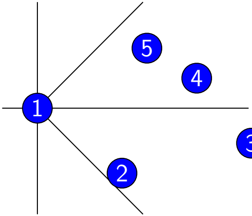
Not all assignments are realizable

- Realizability is hard [Mnëv '88]
- Additional clauses eliminate many unrealizable assignments

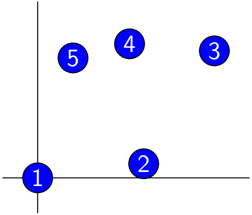
Symmetry Breaking: Sorted & Rotated Around Point 1



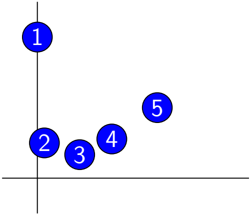
place leftmost point at origin



stretch points to the right to be within $y = x$ and $y = -x$



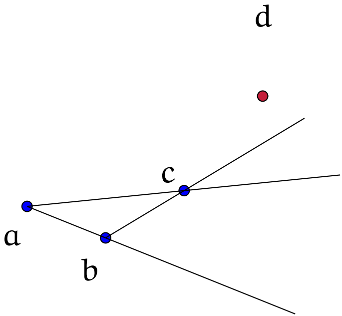
rotate by 45 degrees



projective transformation
$(x, y) \mapsto (y/(x+\epsilon), 1/(x+\epsilon))$

Realizability Constraints [Felsner & Weil '01]

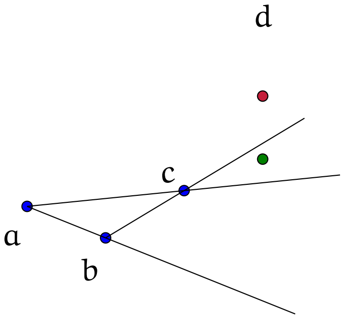
Under the assumption that points are sorted from left to right



O_{abc}	O_{abd}	O_{acd}	O_{bcd}
+	+	+	+

Realizability Constraints [Felsner & Weil '01]

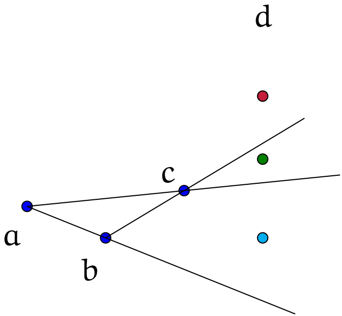
Under the assumption that points are sorted from left to right



O_{abc}	O_{abd}	O_{acd}	O_{bcd}
+	+	+	+
+	+	+	−

Realizability Constraints [Felsner & Weil '01]

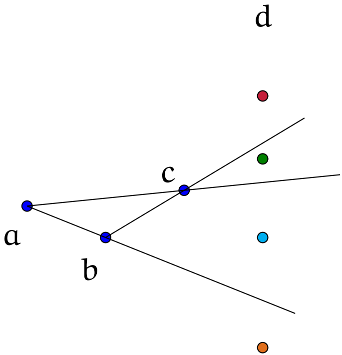
Under the assumption that points are sorted from left to right



O_{abc}	O_{abd}	O_{acd}	O_{bcd}
+	+	+	+
+	+	+	−
+	+	−	−

Realizability Constraints [Felsner & Weil '01]

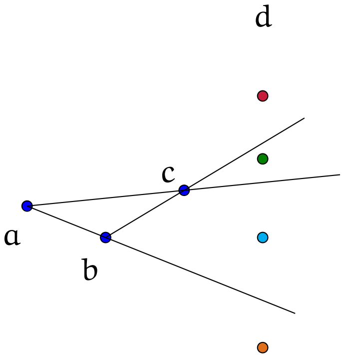
Under the assumption that points are sorted from left to right



O_{abc}	O_{abd}	O_{acd}	O_{bcd}
+	+	+	+
+	+	+	−
+	+	−	−
+	−	−	−

Realizability Constraints [Felsner & Weil '01]

Under the assumption that points are sorted from left to right



O_{abc}	O_{abd}	O_{acd}	O_{bcd}
+	+	+	+
+	+	+	−
+	+	−	−
+	−	−	−
−	−	−	−
−	−	−	+
−	−	+	+
−	+	+	+

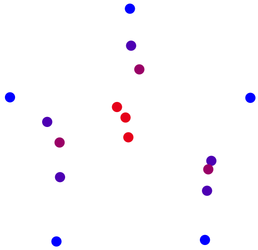
Block multiple sign changes with $\Theta(n^4)$ (ternary) clauses

- $(\overline{O_{abc}} \vee O_{abd} \vee \overline{O_{acd}}) \wedge (O_{abc} \vee \overline{O_{abd}} \vee O_{acd})$
- $(\overline{O_{abc}} \vee O_{acd} \vee \overline{O_{bcd}}) \wedge (O_{abc} \vee \overline{O_{acd}} \vee O_{bcd})$

Encoding Rotational Symmetry

Symmetric unavoidable polygons?

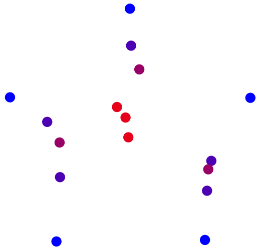
- No three points in a line
- Erdős-Szekeres 1935
- Avoid k-gon: at most 2^{k-2} points
- Happy Ending Problem
- Image from Wikipedia



Encoding Rotational Symmetry

Symmetric unavoidable polygons?

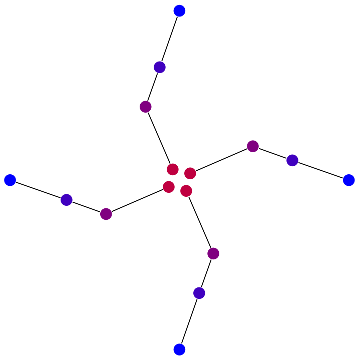
- No three points in a line
- Erdős-Szekeres 1935
- Avoid k-gon: at most 2^{k-2} points
- Happy Ending Problem
- Image from Wikipedia

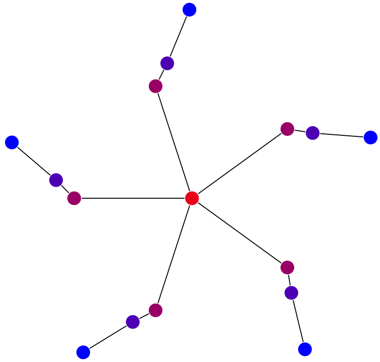


Searching for k-rotational configurations is tricky:

- Points are no longer sorted from left to right
- More complicated axiom clauses
- Let $\pi(p)$ be the point after rotating p by $\frac{360}{k}$ degrees
- Variables O_{abc} and $O_{\pi(a)\pi(b)\pi(c)}$ are equivalent

Two New, Symmetric Point Sets without Hexagons





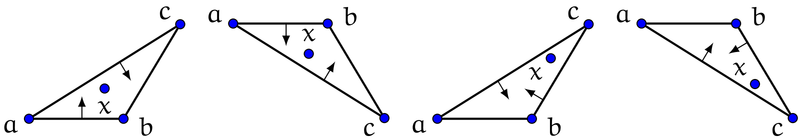
Realized by the `Localizer` tool

- `https://github.com/bsubercaseaux/localizer`

Inside Variables

We introduce inside variables $I_{x;abc}$ which are true if and only if point x is in the triangle abc with $a < x < b$ or $b < x < c$.

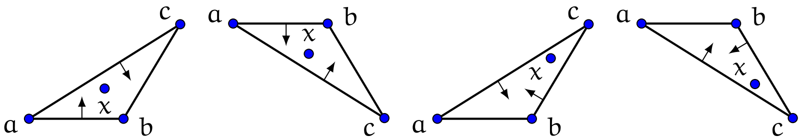
Four possible cases:



Inside Variables

We introduce inside variables $I_{x;abc}$ which are true if and only if point x is in the triangle abc with $a < x < b$ or $b < x < c$.

Four possible cases:



The left two cases with $a < x < b$:

$$I_{x;abc} \leftrightarrow \Big(\big(O_{abc} \rightarrow (\overline{O_{axb}} \land O_{axc})\big) \land \big(\overline{O_{abc}} \rightarrow (O_{axb} \land \overline{O_{axc}})\big) \Big)$$

The right two cases with $b < x < c$:

$$I_{x;abc} \leftrightarrow \Big(\big(O_{abc} \rightarrow (O_{axc} \land \overline{O_{bxc}})\big) \land \big(\overline{O_{abc}} \rightarrow (\overline{O_{axc}} \land O_{bxc})\big) \Big)$$

Hole Variables

We introduce hole variables H_{abc} which are true if and only if no points occur with the triangle abc with $a < b < c$.

$$H_{abc} \vee \bigvee_{a<x<c} I_{x;abc}$$

$$\bigwedge_{a<x<c} \overline{H_{abc}} \vee \overline{I_{x;abc}} \qquad \text{(redundant)}$$

Hole Variables

We introduce hole variables H_{abc} which are true if and only if no points occur with the triangle abc with $a < b < c$.

$$H_{abc} \vee \bigvee_{a<x<c} I_{x;abc}$$

$$\bigwedge_{a<x<c} \overline{H_{abc}} \vee \overline{I_{x;abc}} \qquad \text{(redundant)}$$

Simple 6-hole encoding:

$$\bigvee\nolimits_{a,b,c \in X} \overline{H_{abc}} \quad \forall\, X \subset S \text{ with } |X| = 6$$

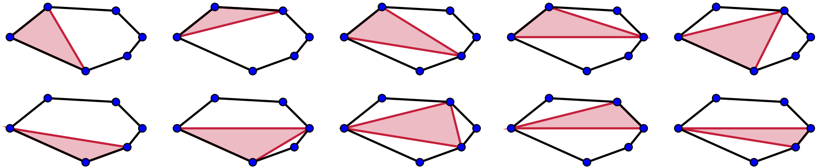
Empty Hexagon Encoding

Given 6 points, how many empty triangles with these points guarantee an empty hexagon (possibly among other points)?

Empty Hexagon Encoding

Given 6 points, how many empty triangles with these points guarantee an empty hexagon (possibly among other points)?

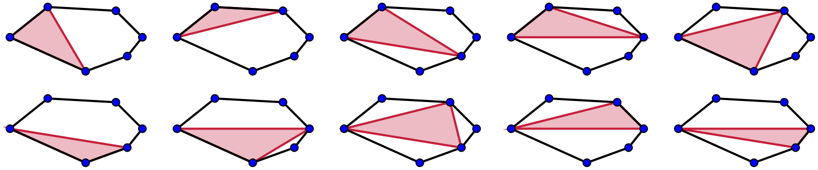
If the points may not be in convex position: 10



Empty Hexagon Encoding

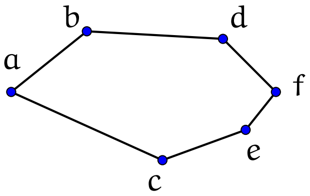
Given 6 points, how many empty triangles with these points guarantee an empty hexagon (possibly among other points)?

If the points may not be in convex position: 10



If the points are in convex position:

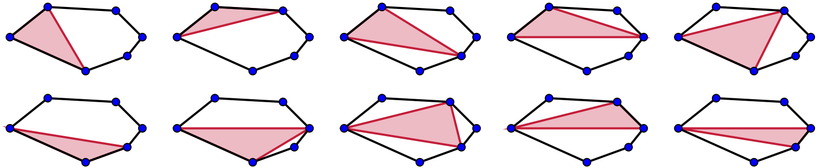
- Requires assignment to four orientation variables
- Includes info which points are above/below the line a to f



Empty Hexagon Encoding

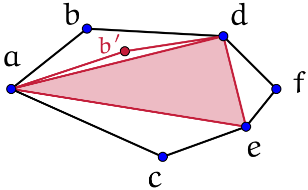
Given 6 points, how many empty triangles with these points guarantee an empty hexagon (possibly among other points)?

If the points may not be in convex position: 10



If the points are in convex position: 1

- Requires assignment to four orientation variables
- Includes info which points are above/below the line a to f



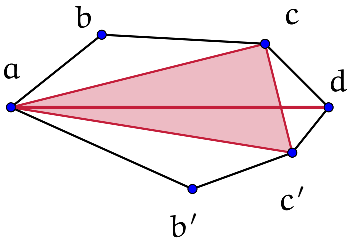
k-Hole Encoding Using $O(n^4)$ Clauses

Shorter clauses, thus more propagation, but still $O(n^6)$

Example

Introduce $O(n^3)$ auxiliary variables:

- A_{acd}: a 4-gon above the line ad
  $\overline{O_{abc}} \wedge \overline{O_{bcd}} \rightarrow A_{acd}$
- $B_{ac'd}$: a 4-gon below the line ad
  $O_{ab'c'} \wedge O_{b'c'd} \rightarrow B_{ac'd}$
- Combine them to block 6-holes
  $\overline{A_{acd}} \vee \overline{B_{ac'd}} \vee \overline{H_{acc'}}$



This reduces the size of the encoding to $O(n^4)$ clauses

Comparison to Existing Work

Szekeres and Peters (2006) solved $g(6) = 17$ in 63 CPU days

- Roughly 40 CPU hours on today's hardware
- https://www.cpubenchmark.net/year-on-year.html

SAT solving, using the same abstraction, is much faster

- The independent SAT approaches by Marić and Scheucher required a few CPU hours
- Their encodings consist of $O(n^k)$ clauses

Our $O(n^4)$ encoding for k-gons and k-holes is even faster

- $g(6) = 17$ can be solved in 10 CPU seconds
- About 4 orders of magnitude faster than the original proof

Verification

The optimization steps are validated or part of the proof

Concurrent solving and proof checking for the first time

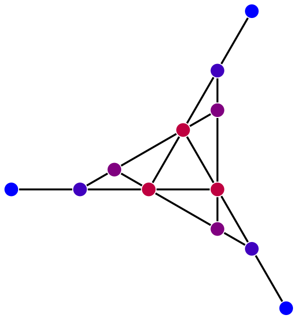
- The solver pipes the proof to a verified checker
- This avoids storing/writing/reading huge files
- Verified checker can easily catch up with the solver

CMU students have formalized and verified all parts in Lean

Everywhere-Unbalanced Point Sets

Everywhere-unbalanced point sets:

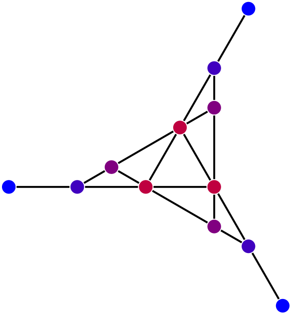
- For each line through 2+ points, unbalanced points by at least k
- $k = 1$ is trivial (a triangle)
- $k = 2$ with 12 points by Noga Alon
- Conjectured for every finite k
- Open: smallest odd configuration



Everywhere-Unbalanced Point Sets

Everywhere-unbalanced point sets:

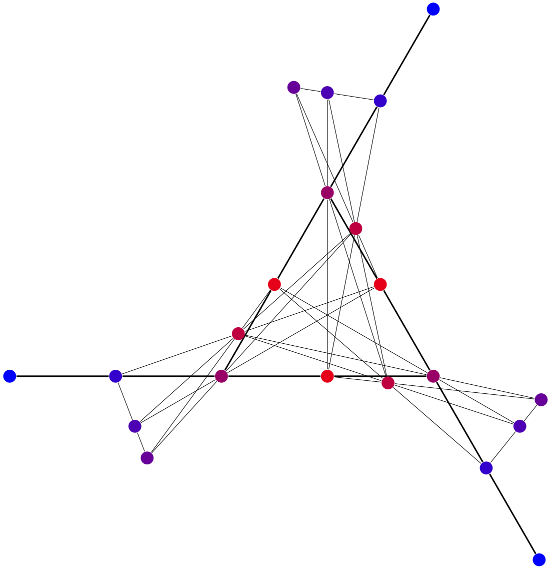
- For each line through 2+ points, unbalanced points by at least k
- $k = 1$ is trivial (a triangle)
- $k = 2$ with 12 points by Noga Alon
- Conjectured for every finite k
- Open: smallest odd configuration



Encoding into SAT:

- Per triple: X_{abc} (c above ab) and Y_{abc} (c below ab)
- Constraints that enforce unbalancedness
- Also realizability constraints

New, Optimal Result: 21 Points and 2-Unbalanced



Conclusions

Theorem
$h(6) = 30$

SAT appears to be the most effective technology to solve a range of problems in computational geometry

Many interesting open problems:

- Minimum number of 4-gons / 5-gons / 6-gons
- Determine whether $g(7) = 33$
- Unbalanced configurations (points can be collinear)