Computational Complexities of Folding

David Eppstein

OSME 2024
Melbourne, Australia, July 2024

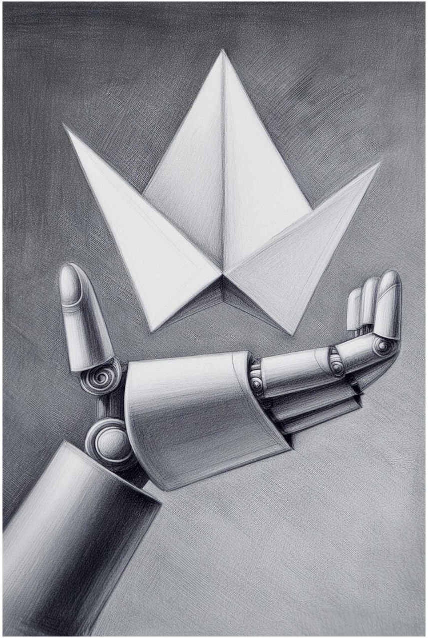
Machine simulation of folding

Given folding pattern and final state, find intermediate motion

Often works pretty well!

Heuristic, with occasional (minor?) issues:

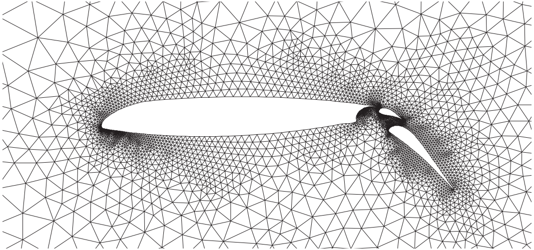
- Unphysical motions
- Inability to use complex folding steps
- Not broken down into simple steps
- Search huge space of step orders [Zhu and Filipov 2019]
- Unclear whether numerical inaccuracy affects validity



Can theory make these methods more robust?

Hope: Provide methods that work as well in practice as current heuristics, but with guarantees on solution quality and algorithmic performance

...as happened in 1990s for finite element mesh generation [Bern et al. 1994]

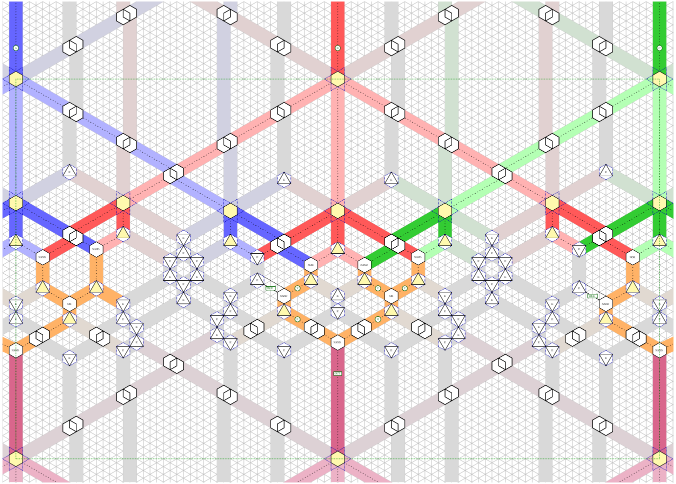


Fear: Hardness results prevent us from providing guaranteed algorithms

Hope: Maybe this tells us something helpful about why we do things the way we do

The main idea

We can design origami structures that perform computations



Simulating these structures cannot be easier than doing the computation!

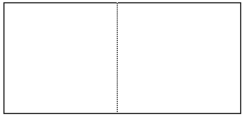
... because if it were, that would be a way of doing the computation

Can it fold?

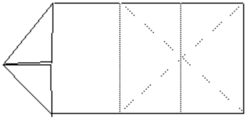
From the 1990s world wide web

Origami Page

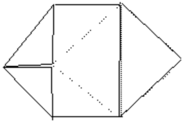
Welcome to the origami page which contains complete and detailed instructions on how to build a simple origami model in 3 easy and one difficult steps. I have entitled this fold "Rabbit style object on geometrical solid".



Step 1:First take a 2:1 rectangular piece off origami paper - ideally rabbit colored on one side and geometric object/rabbit's ear coloured on the other.



Step 2:Fold the corner flaps of the left hand square to the centre and crease the right hand square as shown.



Step 3:Fold the right hand square into this simple base by pinching in the bottom centre crease so that it meets the top centre crease. Crease the left hand square as indicated.

Step 4:Fold all remaining bits of paper on the left hand square into a rabbit-like formation while folding simultaneously the right hand square into a geometric formation.

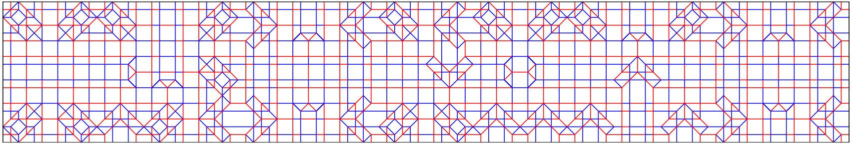
Et voila, as the Japanese say. The completed rabbit-style object on geometric solid. This photo was taken in my office by pointing the Indy Cam and pressing the button.

This model was folded by me from the design worked out by Fred Rohm

So how hard are origami folding instructions?

We need to formalize this as a computational problem

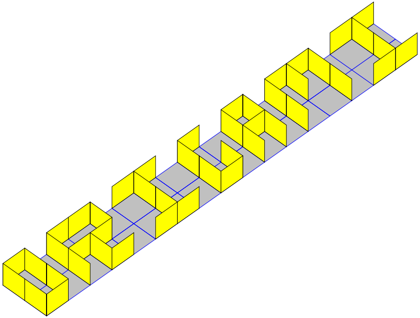
Input: 2d folding pattern (planar straight line graph of folds), possibly labeled with mountain/valley folds



Output: the folded shape

If we can find this we can also answer:

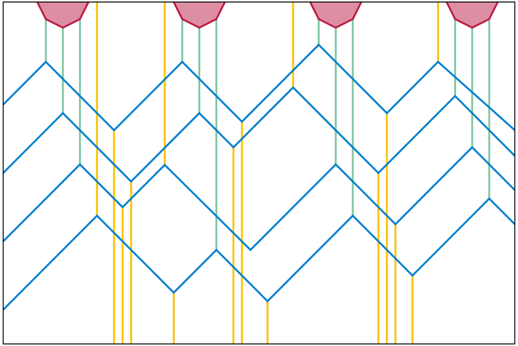
Yes (it folds) or no (it doesn't)



Simplifying assumption: If it folds, it folds flat in the plane (not like this)

Flat foldability is NP-complete

Outline of idea by [Bern and Hayes 1996]: convert Boolean circuit into folding pattern



Blue and green lines: zigzag pleats can fold in two ways, act like wires + binary signals

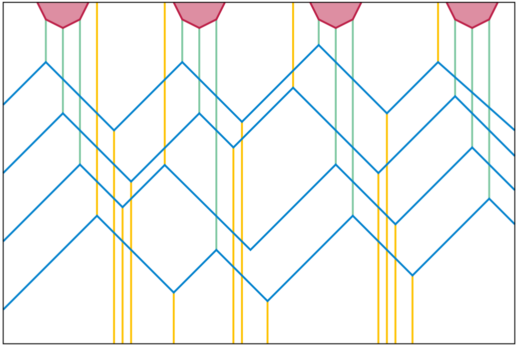
Red shaded regions: logical "gates" that test whether three incoming signals are unequal

Yellow vertical lines: extra creases needed to make the folding pattern work

Original construction is incorrect but was patched by Akitaya et al. [2015]

What does turning circuits into folds mean?

Circuit is satisfiable $\Longleftrightarrow$ can assign values to its wires that make all gate outputs true $\Longleftrightarrow$ the crease pattern can be folded flat without self-intersections



If we could test flat-foldability, applying our test to this crease pattern would tell us whether the circuit is satisfiable

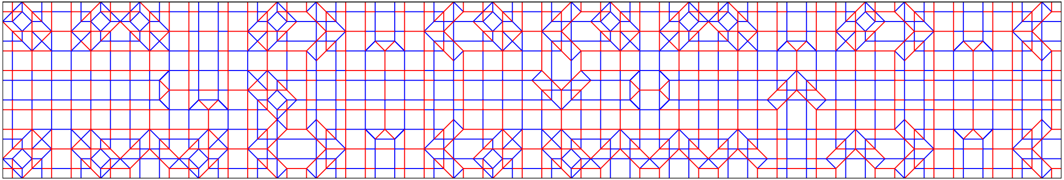
"Not all equal" satisfiability is known to be hard $\Rightarrow$ flat-foldability is also hard

Parameterized complexity:
Analyze difficulty by more than input size

Quantify features of an input that make it difficult by numerical parameters

Separate time bound into function of parameters $\times$ polynomial of input size

Can solve very large problems as long as the parameters stay small

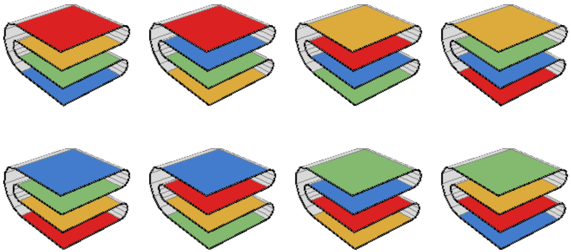


Maybe it's easy to fold grid-like crease patterns in which
one of the two grid dimensions is small?

Grid dimension is the wrong parameter

Reason 1: It doesn't apply to non-grid folding patterns

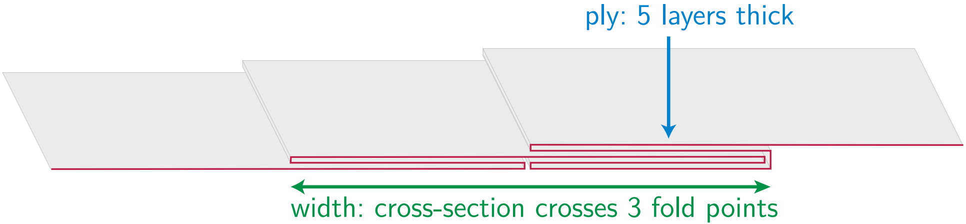
Reason 2: We don't even know how to fold $3 \times n$ square grids



The still-unsolved "map folding problem": flat-fold a grid given as input a mountain–valley assignment on all grid edges

Better parameters

Parameterize by cutting flat-folded state (whatever it is) by vertical lines, and looking at the geometry of the cross-sections



- Ply: how many layers are there, for the cross section with the most layers?
- Width: what is the maximum # fold points that a cross-section passes through?

  (Actual definition uses treewidth of planar graphs, similar but more complicated, does not require cross-sections to be parallel lines)

Fine-grained complexity

Focus on tightness of time bounds rather than the cruder distinction between polynomial and exponential time

Algorithm: Sweep vertical cross-section line left to right

Track cross-sections of folds of left side of line

Fewer cross-sections than folds, $O\big((\mathrm{ply!})^{\mathrm{width}+1}n\big)$

"Fixed-parameter tractable": bad function of parameters but linear in crease pattern size

Exponential time hypothesis $\Rightarrow$ exponential dependence on width cannot be improved

Proof idea: For NP-complete crease pattern, width is proportional to # circuit variables

(But to understand dependence on ply, we need progress on map folding!)

[Eppstein 2023]

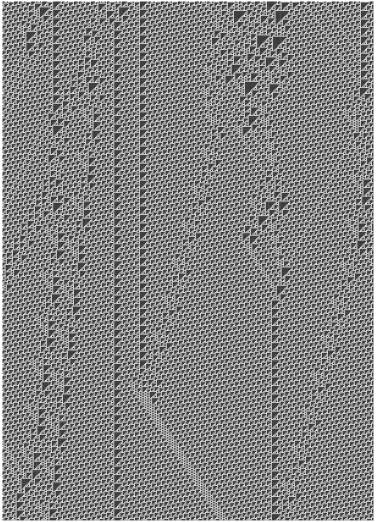
Undecidability

Given as input:

- A repeating crease pattern on an infinite half-plane
- The folded state along the edge of the half-plane, repeating except for a finite perturbation

There is no algorithm to test the existence of a folded state of the entire half-plane in which the perturbation remains finite, even if each fold is uniquely determined and easy to find

[Hull and Zakharevich 2023]

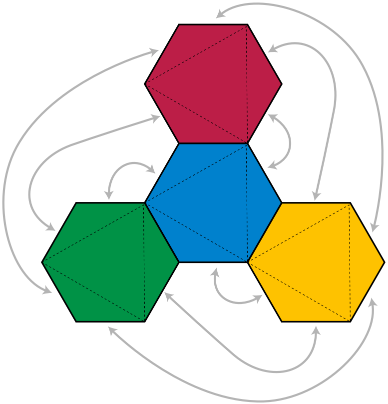


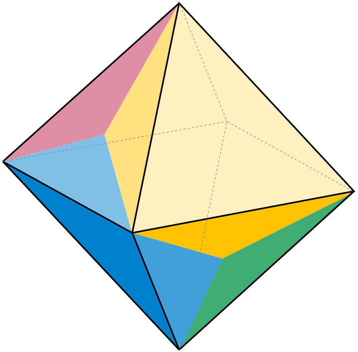
Main idea: simulate "Rule 110" cellular automaton

What does it fold to?

Alexandrov's theorem

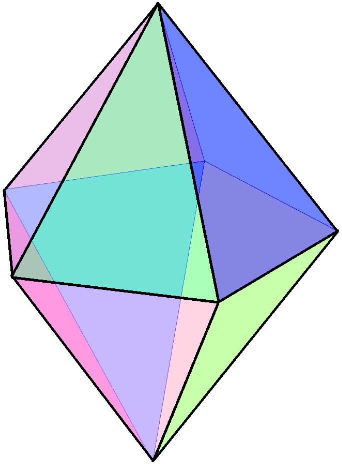
Fold paper to form a surface that is topologically spherical, with finitely many "cone points" of total angle $< 360^\circ \Rightarrow$ exactly one way for it to form a convex polyhedron





But how does the shape of the polyhedron depend on the folding pattern?
And where do the creases go?

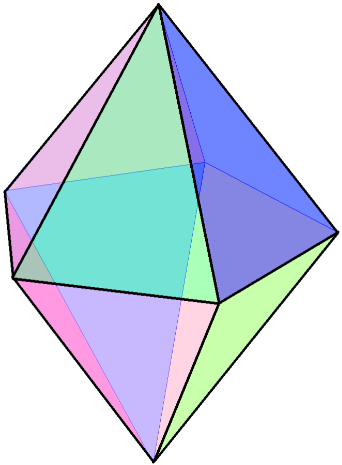
An easy(?) special case



Bipyramid with

- All faces isosceles triangles
- All edge lengths integers
- Same length for the two equal sides of each face
- Base lengths allowed to vary

One dimension lower



Equator must be a cyclic polygon!

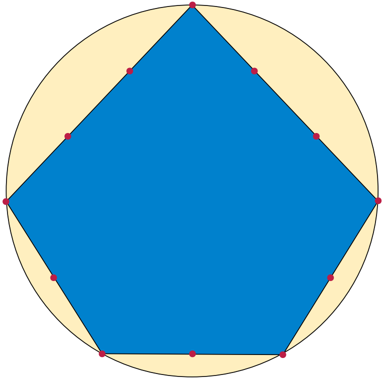
- Symmetry of face dimensions + uniqueness of realization $\Rightarrow$ top and bottom are mirrored $\Rightarrow$ equator is a plane polygon
- All vertices at equal distances from top and bottom apex $\Rightarrow$ they lie on a circle
- Equator edge lengths must be as specified from the folding pattern

Galois complexity

Coordinates for a cyclic polygon with given edge lengths are roots of polynomials with:

High algebraic degree (for regular p-gon with p prime, the polynomial is $x^p - 1$)

Unsolvable Galois groups $\Rightarrow$ no closed-form formula [Varfolomeev 2004]



The same difficulties extend to finding the shape of a bipyramid from its folding pattern

$\Rightarrow$ numerical approximations rather than exact symbolic descriptions may be necessary

How to fold it?

A mismatch between theory and practice

Previous hardness proofs:

Difficult to find folded state of crease pattern

Many simulation tasks:

We know both crease pattern and folded goal

Seek motion from one to the other

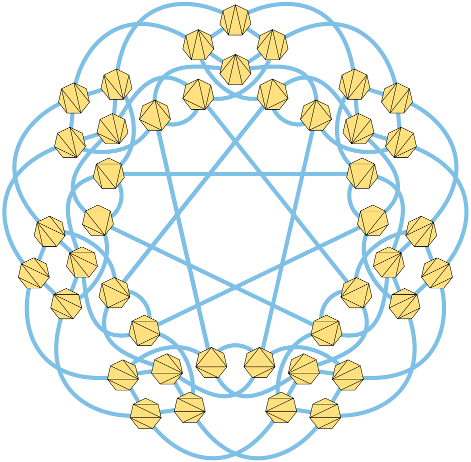
Reconfiguration complexity

Define combinatorial system of states+moves

Study complexity of problems:

- Can I reach one state from another?
- Are all states connected?
- How to reach goal in fewest moves?

Often PSPACE-complete (harder than NP)



Face flips in origami tessellations

States = locally valid mountain-valley assignments on a tessellation:

- Satisfies Kawasaki's & Maekawa's theorems
- Not required to have a global flat folding

Move = reverse the assignment on a single face

On triangular tiling, $O(n)$ flips always suffice, but it is NP-complete to minimize length

[Akitaya et al. 2020b]

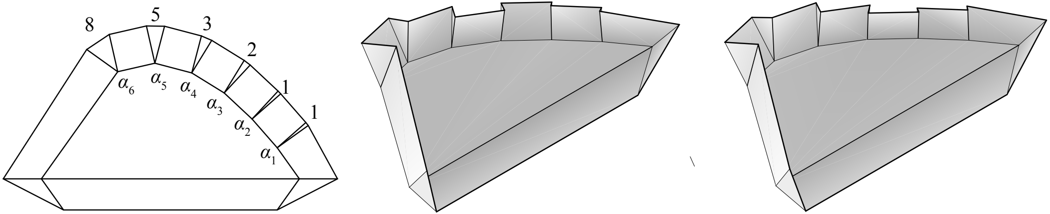
Weaknesses: NP vs PSPACE; local vs global folding; do moves make sense?

When even a single step is hard to find

For flat-foldable unlabeled crease patterns, starting from a completely unfolded state, rigidly flexing every crease simultaneously is weakly NP-complete

Must subdivide certain angles into two subsets with equal sums

Weakly NP-complete ⇒ high numerical precision is necessary for accurate results



Finding a subset of a given crease system that can flex rigidly, starting from a completely unfolded state, is strongly NP-complete

[Akitaya et al. 2020a]

Flaps and flips

A toy problem for origami reconfiguration

State: Equal-sized squares of paper lie flat on a flat table, attached to the table by a hinge along one side

Move: flip one square to new flat position with hinged edge attached in same place

All other squares remain where they were

Moving square can be pulled from or inserted into pockets (non-rigidly); can flip across hinge even if other squares lie above hinge

Example: The green square can flip across its hinge but must remain under the lower red square

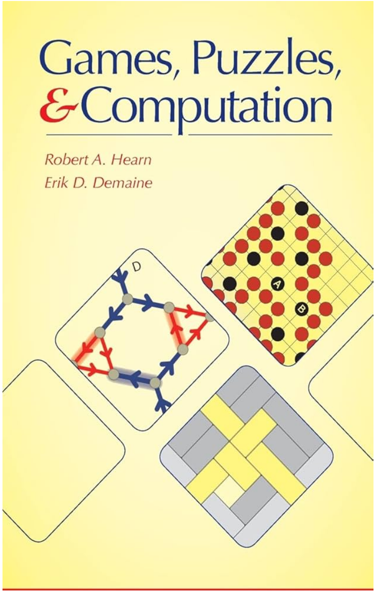
Nondeterministic constraint logic

Circuit-like reconfiguration problem used by [Hearn and Demaine 2009] to prove PSPACE-hardness of many puzzles and games

State: planar diagram of

- Blue and red arrows
- Gates: junction where three arrows meet
- OR (3 blue): ≥ 1 arrow must point inward
- AND (1 blue, 2 red): at least one blue or both red arrows must point inward

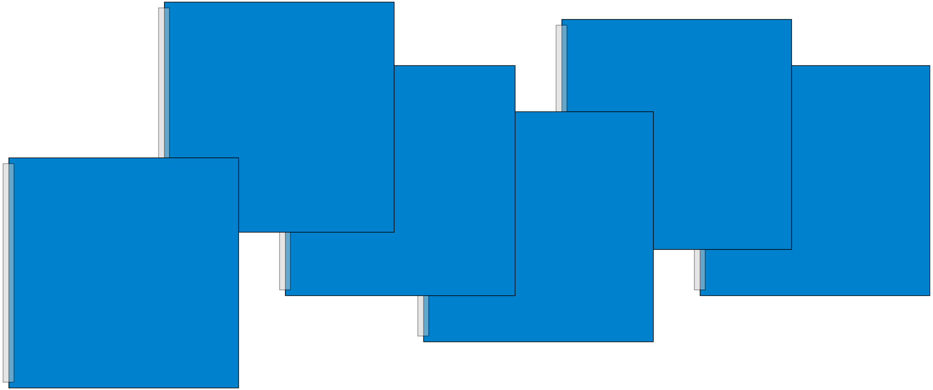
Move: Reverse an arrow!



Simulating an arrow

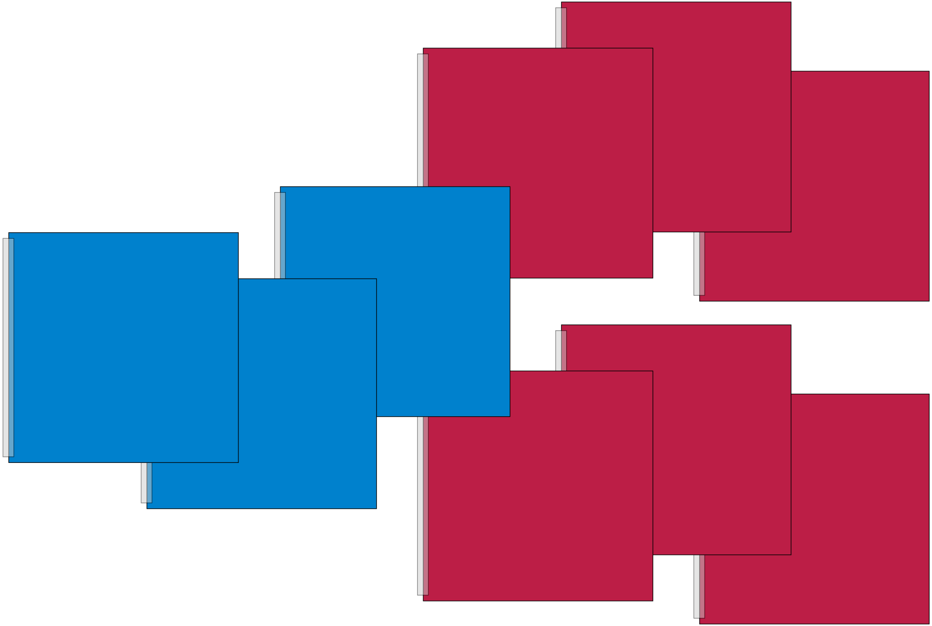
arrow points toward the hinged side of each square

to reverse arrow, flip squares one at a time starting from the arrowhead
(intermediate states have no arrowhead at either end, not problematic)

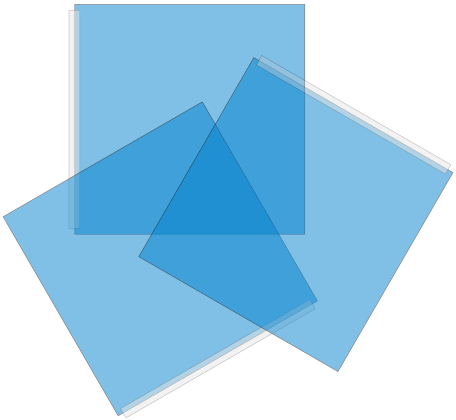


each square overlaps the hinges of its neighbors on both sides,
impossible for squares at both ends of arrow to flip inward (double arrowhead)

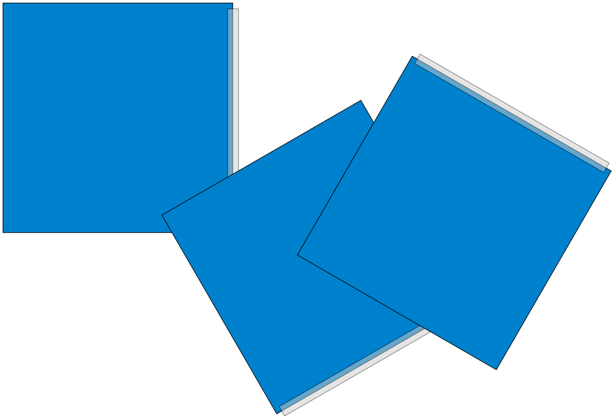
Simulating an and-gate



Simulating an or-gate



three outward-pointing arrows: impossible
cyclic above-below order in central triangle



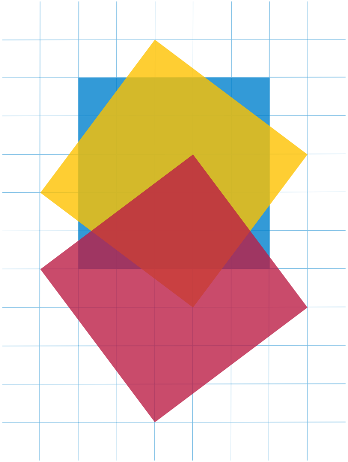
if one arrow points in, all squares lie flat

if two point in, either can be flipped out

PSPACE-completeness and its consequences

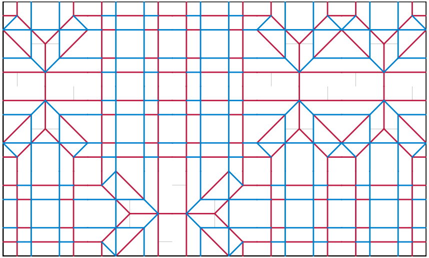
For the "flaps and flips" reconfiguration problem:

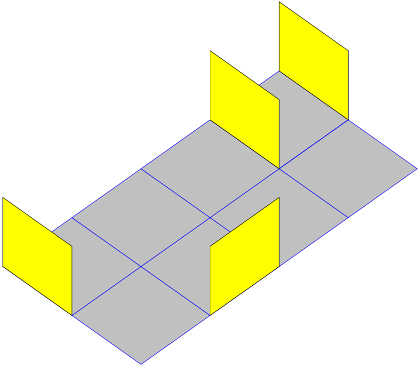
- Testing whether one state can reach another is PSPACE-complete
- Testing whether all states are connected is PSPACE-complete
- Completeness holds even for patterns of bounded ply and bounded width
- Hard configurations can have integer vertex coordinates
- Getting from one state to another may require exponentially many flips



Single-sheet flat-folding

We can make single-sheet crease patterns that produce flaps!





Hardness proof should extend to reconfiguring flat foldings of single-sheet patterns with $O(1)$ changes of crease orientation per move or only allowing refolding along one line segment at a time

End matter

Conclusions

Origami is hard!

... that's part of what makes it interesting

NP-hardness, ETH $\Rightarrow$ if you don't already know what you're folding, figuring it out from the crease pattern is non-obvious

Galois complexity $\Rightarrow$ for fully 3d folds, numerical approximation may be necessary

PSPACE-hardness $\Rightarrow$ repeated folding and unfolding may be necessary

Questions for future research

Hardness for more realistic folding models?

How to quantify ease of folding a design?

How to incorporate ease of folding into the design process?

Image credits I

Robot origami and sand through a magnifying glass created by Adobe Generative AI

Airfoil mesh from [Barth and Jespersen 1989]

Folding pattern for Rule 110 cell from [Hull and Zakharevich 2023]

Nearby star TSP from `http://www.math.uwaterloo.ca/tsp/star/hyg.html`

Force-directed tree drawing from [Bannister et al. 2012]

"Origami" crease pattern and crease pattern for flaps from Origami Maze Font by Erik Demaine, Martin Demaine, and Jason Ku, 2010, `https://erikdemaine.org/fonts/maze/`

Rabbit on box instructions from `https://web.archive.org/web/20050816201015/http://www.richardclegg.org/htdocs/origami.html`

Schematic view of Bern–Hayes circuit simulation from [Eppstein 2023]

Eight ways to fold a 2x2 map along its creases, CC-BY-SA image by Robert Dickau, March 24, 2010, `https://commons.wikimedia.org/wiki/File:MapFoldings-2x2.png`

Rule 110 pattern created with Golly, `https://golly.sourceforge.io/`

Image credits II

Pentagonal bipyramid, CC-BY-SA image by Quatrostein, February 8, 2009, https://commons.wikimedia.org/wiki/File:Pentagonale_bipiramide.png

Rubik's cube, CC-BY-SA image by Famartin, January 14, 2021, https://commons.wikimedia.org/wiki/File:2021-01-14_18_25_51_A_scrambled_Rubik%27s_Cube_in_the_Franklin_Farm_section_of_Oak_Hill,_Fairfax_County,_Virginia_(cropped).jpg

Blue triangle mesh, cropped from CC-BY-NC image by Ryan Robinson, May 4, 2007, "Advanced Concept Car Prototype", https://www.flickr.com/photos/infinite-origami/484625351/

Single-step rigid folding hardness illustration from [Akitaya et al. 2020a]

Games, Puzzles, & Computation: cover of [Hearn and Demaine 2009]

Other images created by the author

References I

Hugo A. Akitaya, Kenneth C. Cheung, Erik D. Demaine, Takashi Horiyama, Thomas Hull, Jason S. Ku, Tomohiro Tachi, and Ryuhei Uehara. Box pleating is hard. In Jin Akiyama, Hiro Ito, Toshinori Sakai, and Yushi Uno, editors, *Discrete and Computational Geometry and Graphs – 18th Japan Conference, JCDCGG 2015, Kyoto, Japan, September 14–16, 2015, Revised Selected Papers*, volume 9943 of *Lecture Notes in Comput. Sci.*, pages 167–179. Springer, 2015. doi: 10.1007/978-3-319-48532-4_15.

Hugo A. Akitaya, Erik D. Demaine, Takashi Horiyama, Thomas C. Hull, Jason S. Ku, and Tomohiro Tachi. Rigid foldability is NP-hard. *J. Comput. Geom.*, 11(1):93–124, 2020a. doi: 10.20382/jocg.v11i1a4.

Hugo A. Akitaya, Vida Dujmović, David Eppstein, Thomas Hull, Kshitij Jain, and Anna Lubiw. Face flips in origami tessellations. *J. Comput. Geom.*, 11(1):397–417, 2020b. doi: 10.20382/jocg.v11i1a15.

References II

Michael J. Bannister, David Eppstein, Michael T. Goodrich, and Lowell Trott. Force-directed graph drawing using social gravity and scaling. In Walter Didimo and Maurizio Patrignani, editors, *Proc. 20th Int. Symp. Graph Drawing*, volume 7704 of *Lecture Notes in Computer Science*, pages 414–425. Springer-Verlag, 2012. doi: 10.1007/978-3-642-36763-2_37.

Timothy Barth and Dennis Jespersen. The design and application of upwind schemes on unstructured meshes. In *27th Aerospace Sciences Meeting*. American Institute of Aeronautics and Astronautics, 1989. doi: 10.2514/6.1989-366.

Marshall Bern and Barry Hayes. The complexity of flat origami. In *Proc. 7th ACM-SIAM Symposium on Discrete Algorithms (SODA '96)*, pages 175–183, Philadelphia, PA, 1996. Society for Industrial and Applied Mathematics. URL https://portal.acm.org/citation.cfm?id=313852.313918.

Marshall W. Bern, David Eppstein, and John R. Gilbert. Provably good mesh generation. *J. Comput. System Sci.*, 48(3):384–409, 1994. doi: 10.1016/S0022-0000(05)80059-5.

References III

David Eppstein. A parameterized algorithm for flat folding. In *Proc. 35th Canadian Conference on Computational Geometry (CCCG 2023)*, pages 35–42, 2023.

Robert A. Hearn and Erik D. Demaine. *Games, Puzzles, and Computation*. A K Peters, 2009.

Thomas C. Hull and Inna Zakharevich. Flat origami is Turing complete. Electronic preprint arxiv:2309.07932, 2023.

V. V. Varfolomeev. Galois groups of the Heron-Sabitov polynomials for pentagons inscribed in a circle. *Matematicheskiĭ Sbornik*, 195(2):3–16, 2004. doi: 10.1070/SM2004v195n02ABEH000798.

Yi Zhu and Evgueni T. Filipov. An efficient numerical approach for simulating contact in origami assemblages. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 475(2230):20190366, October 2019. doi: 10.1098/rspa.2019.0366.