

Folding Polyominoes with Holes into a Cube:
A Computational Approach

Student:

Aaron You

Hong Kong International School, Hong Kong

Supervising Teacher:

Graham Nolan

Hong Kong International School, Hong Kong

August 2023

Abstract

Polyominoes are a well-known subject in recreational mathematics, consisting of plane geometric figures formed by connecting unit squares edge-to-edge. In this study, we explore the problem of folding polyominoes with holes into a cube, which has potential applications in various fields, such as robotics, material science, and 3D printing. In particular, we investigate the conditions under which a given polyomino with holes can be folded into a cube and develop an algorithm to identify such configurations.

My approach begins with the generation of various polyominoes with holes and the identification of their properties, such as their size and number of holes. We then develop a novel algorithm that utilizes graph theory and combinatorial techniques to determine whether a given polyomino with holes can be folded into a cube. The algorithm's performance is evaluated through a series of experiments on a range of polyominoes with varying complexity. This study represents a step towards fuller understanding of the folding problem for polyominoes with holes and offers useful insights for future research in this area.

Keywords: polyominoes, folding, cube, holes, computational geometry, algorithms, graph theory, combinatorics

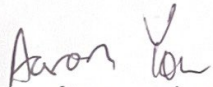
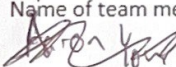
Commitments on Academic Honesty and Integrity

We hereby declare that we

1. are fully committed to the principle of honesty, integrity and fair play throughout the competition.
2. actually perform the research work ourselves and thus truly understand the content of the work.
3. observe the common standard of academic integrity adopted by most journals and degree theses.
4. have declared all the assistance and contribution we have received from any personnel, agency, institution, etc. for the research work.
5. undertake to avoid getting in touch with assessment panel members in a way that may lead to direct or indirect conflict of interest.
6. undertake to avoid any interaction with assessment panel members that would undermine the neutrality of the panel member and fairness of the assessment process.
7. observe the safety regulations of the laboratory(ies) where the we conduct the experiment(s), if applicable.
8. observe all rules and regulations of the competition.
9. agree that the decision of YHSA(Asia) is final in all matters related to the competition.

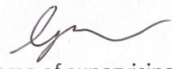
We understand and agree that failure to honour the above commitments may lead to disqualification from the competition and/or removal of reward, if applicable; that any unethical deeds, if found, will be disclosed to the school principal of team member(s) and relevant parties if deemed necessary; and that the decision of YHSA(Asia) is final and no appeal will be accepted.

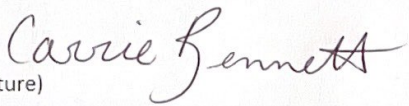
(Signatures of full team below)


Name of team member:
 Aaron You

Name of team member:

Name of team member:


Name of supervising teacher:
Graham Nolan

Noted and endorsed by  (signature) Name of school principal: Carrie Bennett
--

Contents

1	Introduction	4
2	Definitions and Notation	4
3	Methodology	5
3.1	Data Structures and Their Implementation	5
3.1.1	Graphical Representation of Polyominoes	5
3.1.2	Representation of the Unit Cube	8
3.2	Investigation of Algorithms	10
3.2.1	Recursive Graph Traversal (Folding)	10
3.2.2	Implementation of the Greedy Algorithm	11
3.3	Asymptotic Analysis	12
4	Next Steps	13
4.1	Addressing Larger Polyominoes	13
4.2	Exploring Other Folding Models	13
4.3	Implementation and Limitations of the Current Algorithm . .	13
5	Conclusion	14

1 Introduction

This paper explores the intriguing problem of transforming a polyomino — a planar polygon composed of unit squares connected edge-to-edge — into a unit cube through origami-style folding, conforming to standard rules that permit multiple layers of paper on each unit-square face.

A grid model, implicitly suggested by Beluhov’s puzzles, promotes folding along the polyomino’s grid lines, mandates orthogonal fold angles ($\pm 90^\circ$ and $\pm 180^\circ$), and prohibits folding materials strictly inside the cube [4].

However, the polyominoes from Beluhov’s puzzles are not tree-shaped and they possess non-simply connected interiors characterized by holes or slits. These features arguably add to the puzzles’ charm and complexity. Consequently, our paper aims to further understand which polyominoes with holes can be folded into a unit cube within the grid model. While we don’t provide a comprehensive characterization, we do offer numerous intriguing conditions that dictate whether a polyomino can or cannot be transformed into a unit cube.

The problem’s sensitivity to the chosen model is worth noting. The half-grid model, another primary model explored in past studies, permits orthogonal and diagonal folds between half-integral points and has been shown to accommodate the folding of all polyominoes with at least ten unit squares into a unit cube [1, 3]. However, we choose to focus on the grid model, as it aligns with Beluhov’s puzzles and as it has a finite number of polyominoes that fold into a cube unlike the half-grid model [2].

2 Definitions and Notation

A **polyomino** is defined as a connected polygon, denoted as P , within a two-dimensional plane. It is created by combining n unit squares on a square lattice. The vertices of P are referred to as the grid points of P . P is

considered an open region, excluding its boundary, which encompasses the n open unit squares in the form of $(x_i, x_i + 1) \cdot (y_i, y_i + 1)$, along with select shared unit-length edges and grid points among these squares. It is worth noting that the inclusion of the common edge between every adjacent pair of squares is not a requirement for P . In cases where such an edge is missing from P , it is referred to as a **slit edge**. However, P must possess a minimum of $n - 1$ unit-length edges to ensure interior connectivity.

A **hole** in a polyomino P refers to a bounded connected component of P 's exterior, where its boundary corresponds to one of the connected components of P 's boundary, excluding the outermost one. It is assumed that P does not contain holes consisting solely of a single grid point, as these holes do not impact foldability and can be filled in by incorporating them into P . A hole is considered a **slit** if it has zero area (excluding a single point) and is entirely composed of one or more slit edges.

Within this study, our focus lies in the problem of folding a given polyomino P with holes to form a unit. This folding process permits creases along edges of the lattice, with fold angles of $\pm 90^\circ$ or $\pm 180^\circ$.

3 Methodology

3.1 Data Structures and Their Implementation

The inherent merit of polyominoes lies in their intuitive design, uniquely that they are easily representable as a data structure, useful particularly in their capacity for efficient storage and data manipulation. This merit underlies the decision to utilize polyomino structure in crafting data structures for our study. This subsection aims to delve into a detailed scrutiny of the data structures that form the foundation of the proposed algorithm.

3.1.1 Graphical Representation of Polyominoes

To address the intricate problem of evaluating a cube's foldability, it becomes obvious to represent a polyomino in the form of a graph.

Each individual square within the polyomino is a unique structure, holding four variables that denote the squares situated above, below, to the right and left of it. In the absence of a square in any of these four directions, the non-existence of a square is represented as null.

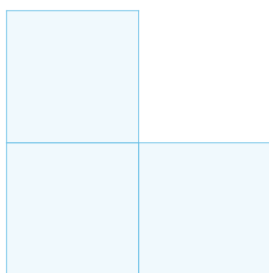


Figure 1: A polyomino

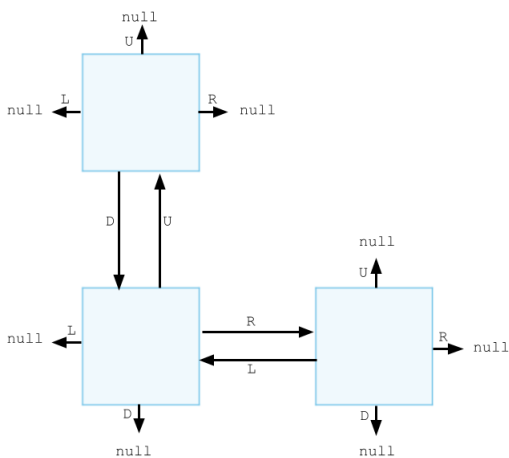


Figure 2: A representation of the polyomino above using a graph. U, D, R, L representing directions Up, Down, Right, Left, respectively.

In the event of a slit or a hole, the algorithm represents it as null as well. This is primarily because the distinction between a non-existent square and

a slit or hole becomes inconsequential when the factor of foldability is under consideration.

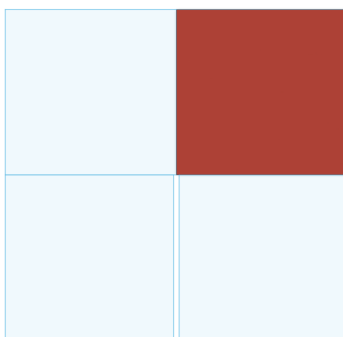


Figure 3: Original polyomino. Red squares indicate a hole.

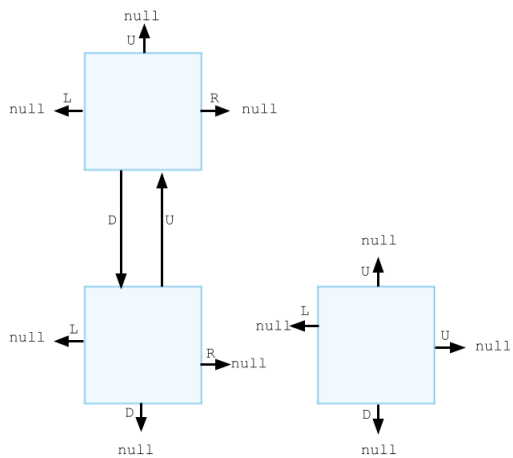


Figure 4: A representation of the polyomino above using a graph. Slits and holes represented with null.

3.1.2 Representation of the Unit Cube

An investigation into whether a polyomino lattice can be folded into a cube requires the incorporation of an additional data structure, designed to keep track of the faces of a cube during the folding progression.

A cube structure can be conceived with references to the top, bottom, front, back, left, and right faces of a cube. Each face retains a map of squares that represent the squares existing on that specific plane.

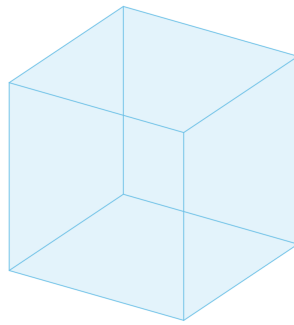


Figure 5: A unit-cube

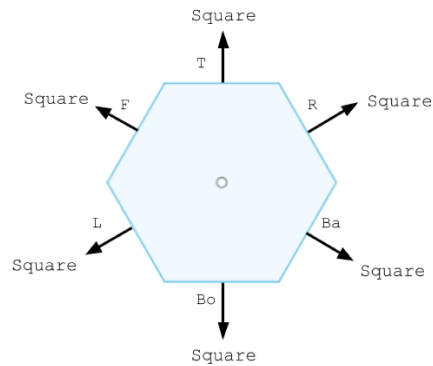


Figure 6: A representation of the unit-cube. Letters T, Bo, R, L, F, Ba, represent top, bottom, right, left, front, and back reference variables respectively.

This structure is not confined to a single square per plane seeing that, an entire lattice of squares can exist on a single face of the cube during the folding process. This calls for the tracking of the graph representation of a polyomino, as opposed to just a single square.

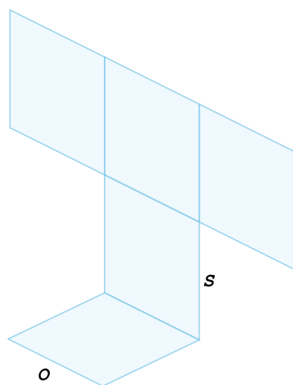


Figure 7: A representation of a polyomino during the folding process. Bottom facing polyomino is labeled as Polyomino *O* and Front facing polyomino is labeled as Polyomino *S*

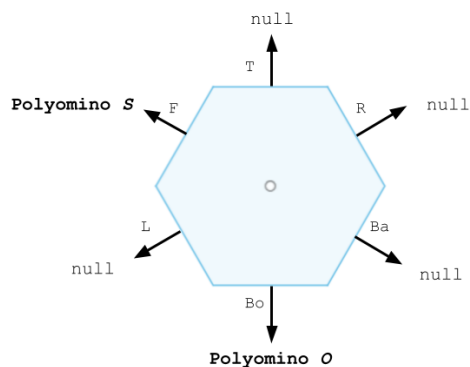


Figure 8: A representation of the polyomino during the folding process. Letters T, Bo, R, L, F, Ba, represent top, bottom, right, left, front, and back reference variables respectively.

3.2 Investigation of Algorithms

In the next subsection, we turn our attention to the algorithms that offer the computational underpinnings necessary for the execution of our study. A deep dive into each algorithm will provide the necessary framework for understanding the operations of the folding process.

3.2.1 Recursive Graph Traversal (Folding)

During the folding of a polyomino, it is crucial to consider that not only one square is being moved to a different plane, but all squares connected to it except for the square the fold originated from. A recursive traversal of the polyomino allows us to store all of the squares that are to be repositioned to a different plane.

Let us represent O as the square lattice not altering planes and S as the square lattice we aim to fold.

The traversal commences on the initial square of S , marking all squares on the side of O along the fold line as visited.

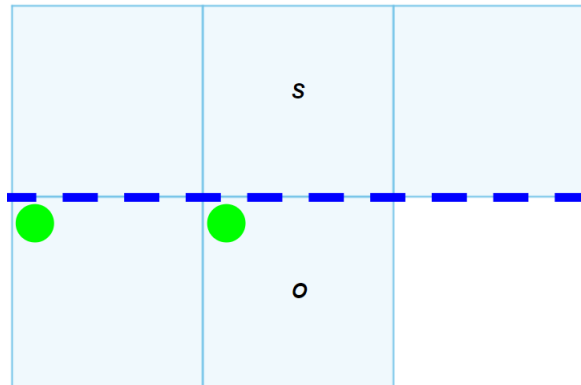


Figure 9: A polyomino. Dotted blue line represents a valley fold. Green dots on the top of a square means that square is marked "visited"

We then traverse through S marking each square we traverse to as visited store it as a square to be moved to another plane. If we traverse to a square

that is already marked as visited, we immediately "return", reducing time complexity by a significant margin.

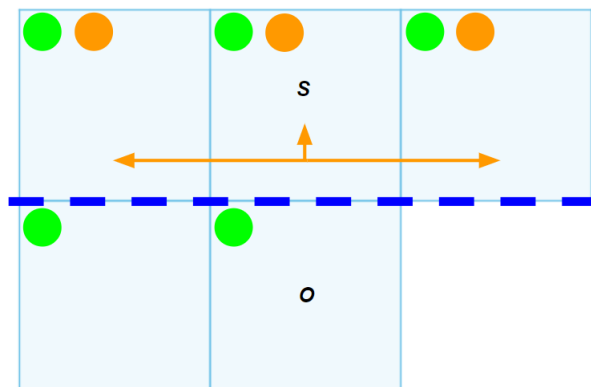


Figure 10: Orange dots show squares marked to move to another plane. Orange arrows show the movement of the traversal.

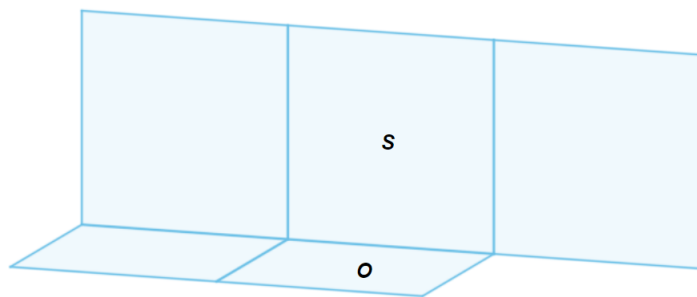


Figure 11: Completed fold.

We iterate through the squares adjacent to the line of the fold and set the referencing variables to null unlinking Polyomino O and Polyomino S . Finally, we copy Polyomino S to the front face of the cube Datastructure.

3.2.2 Implementation of the Greedy Algorithm

Given our objective is to maximize the number of faces our polyomino square lattice can cover in a cube, we can implement a greedy algorithm to

make 90° inwards folds at every edge of the grid.

Initially, a base is randomly determined as the bottom of the cube. Then, branching out from the base, faces of the square lattice are folded 90° inwards in the direction of the center.

All protruding square faces will ultimately be folded into one of the six faces of the unit-cube as multiple layers of paper on each unit-square face is permitted. Hence, we can make our algorithm stop once all 6 faces are occupied by at least 1 square.

3.3 Asymptotic Analysis

This section presents an extensive analysis of the time complexity associated with a particular algorithm, which comprises two primary components: a graph traversal algorithm and a greedy algorithm.

The graph traversal algorithm operates through the method of memoization, a technique that stores the results of expensive function calls and reuses them when the same inputs occur. This approach ensures that the algorithm traverses each square node only once. Consequently, the tight bound, or Theta complexity, of this algorithm is $\Theta(n)$, where n is the total number of squares in the polyomino.

The other component, the greedy algorithm, executes a fold operation at every connecting edge in the polyomino. Given that a polyomino with n squares has $n - 1$ connecting edges, the greedy algorithm performs $n - 1$ folds in total. Thus, similar to the graph traversal algorithm, the Theta complexity of the greedy algorithm is $\Theta(n)$, indicating a linear growth pattern.

Notably, each fold operation performed by the greedy algorithm invokes a call to the graph traversal algorithm. As a result, the time complexity of the combined system becomes $\Theta(n \cdot n) = \Theta(n^2)$. This quadratic growth denotes that the system's time complexity increases quadratically with the size of the input.

However, an optimization has been implemented in the combined system, which terminates the program once all six faces of the cube have been filled. This optimization imposes an upper bound on the time complexity, denoted by Big O notation. Thus, the overall time complexity for the combined system, taking the optimization into account, is $O(n^2)$. This indicates that, in the worst-case scenario, the system's time complexity will not exceed a quadratic function of n .

4 Next Steps

4.1 Addressing Larger Polyominoes

This research focused on relatively small polyominoes. As the size of the polyomino increases, the complexity of the folding problem grows exponentially. Future work could address this by developing more efficient algorithms or using more powerful computing resources to tackle larger polyominoes.

4.2 Exploring Other Folding Models

This research focused on the grid model of folding. However, other folding models, such as the half-grid model, have also been studied. Future research could explore these other models in more depth, comparing their strengths and weaknesses and possibly finding new insights by combining different models.

4.3 Implementation and Limitations of the Current Algorithm

The proposed algorithm, though rigorously conceptualized, remains theoretical due to its lack of implementation. Future work could aim to identify these limitations by implementing the algorithm and putting it through thorough testing. Then, improvements or entirely new algorithms could be proposed to handle cases where our current algorithm falls short. This could involve accounting for more complex slit configurations or developing methods to handle polyominoes with larger numbers of holes.

5 Conclusion

The aim of this research was to explore the intriguing problem of folding polyominoes with holes into a unit cube using conforming to the grid model folding standards. This study proposes a computational analysis that determines whether a polyomino can or cannot be transformed into a unit cube.

We hope that our research contributes to a better understanding of this problem and serves as a stepping-stone for future studies in this field.

References

- [1] Aichholzer, O., Akitaya, H. A., Cheung, K., Demaine, E. D., Demaine, M. L., Fekete, S. P., Kleist, L., Kostitsyna, I., Löffler, M., Masárová, Z., Mundilova, K., & Schmidt, C. (2019). Folding Polyominoes With Holes Into a Cube. *Canadian Conference on Computational Geometry*, 164–170. <http://www.diva-portal.org/smash/record.jsf?pid=diva2:1366092pid=diva2:1366092>.
- [2] Benbernou, N. M., Demaine, E. D., Demaine, M. L., & Lubiw, A. (2020). Universal Hinge Patterns for Folding Strips Efficiently Into Any Grid Polyhedron. *Computational Geometry: Theory and Applications*, 89, 101633. <https://doi.org/10.1016/j.comgeo.2020.101633>
- [3] Czajkowski, K. Y., Demaine, E. D., Demaine, M. L., Epling, K., Kraft, R., Mundilova, K., & Smith, R. (2020). Folding Small Polyominoes Into a Unit Cube. *Canadian Conference on Computational Geometry*, 95–100. <http://dblp.uni-trier.de/db/conf/cccg/cccg2020.html#CzajkowskiDDEKM20>
- [4] Nbeluhov. (2014, October 25). Cube folding. *Puzzled by Titles*. <https://nbpuzzles.wordpress.com/2014/06/08/cube-folding/>