Common Nets

Aresh Pourkavoos

July 5, 2024

1 Introduction

A video by Matt Parker [1] explores the following problem: are there multiple boxes which may be cut and unfolded to produce the same 2D net? Unless mentioned otherwise, a *box* is a rectangular prism with positive integer side lengths, and a *net* is a connected shape made of unit squares whose edges are aligned with those of the box when folded. Parker summarizes the following results, mostly from Ryuhei Uehara and collaborators:

- The $1 \times 1 \times 5$ and $1 \times 2 \times 3$ boxes have a common net consisting of 22 unit squares. In fact, all 2263 such nets have been enumerated. [2]
- For all positive integers k, the $1 \times 5 \times 2k$ and $1 \times 1 \times (6k + 2)$ boxes have a common net (which also tiles the plane), so there are infinitely many pairs of distinct boxes with a common net. [3]
- One of the 2263 nets mentioned above also folds into a double-covered 1×11 rectangle, which may be considered a $0 \times 1 \times 11$ "box." However, some of the unit squares are folded in half in the process. To avoid this, we may scale the net by a factor of 2, producing a common net of the $0 \times 2 \times 22$, $2 \times 2 \times 10$, and $2 \times 4 \times 6$ boxes. [4]
- The $1 \times 3 \times 3$ and $1 \times 1 \times 7$ boxes have a common net consisting of 30 unit squares. This net also folds along diagonal lines to produce a $\sqrt{5} \times \sqrt{5} \times \sqrt{5}$ "box." [5]
- For all positive integers k, the following three boxes have a common net:

$$\begin{aligned} &2k &&\times(4k+9) &&\times 2(16k+13) \\ &(4k+3) &&\times 2(k+3) &&\times 8(4k+3) \\ &(4k+3) &&\times 2(4k+3) &&\times 2(7k+12) \end{aligned}$$

  So there are infinitely many triples of distinct boxes with a common net. [6]

Parker concludes with the observation that the smallest surface area shared by three boxes is 46, the area of the $1 \times 1 \times 11$, $1 \times 2 \times 7$, and $1 \times 3 \times 5$ boxes. However, it is unknown whether these boxes have a common net. Here, we present an algorithm to search for these nets.

2 Definitions

First, we define a net more precisely. Consider the infinite square grid as a graph G, where each vertex represents a square and each edge connects a pair of adjacent squares. (In other words, G is the 1-skeleton of the dual of the square grid.) To maintain visual clarity, the vertices of G will be referred to as squares. A

[1]Parker 2022, " Can the Same Net Fold into Two Shapes?", youtu.be/jOTTZtVPrgo
[2]Matsui and Uehara, jaist.ac.jp/ uehara/etc/origami/nets/index-e.html
[3]Uehara 2008, "Polygons Folding to Plural Incongruent Orthogonal Boxes"
[4]Abel et al 2011, "Common Developments of Several Different Orthogonal Boxes"
[5]Horiyama et al 2015, "Common Developments of Three Incongruent Boxes of Area 30"
[6]Shirakawa and Uehara 2013, "Common Developments of Three Different Orthogonal Boxes"

generalized polyomino, or *GP*, is a connected subgraph (S, E) of G, up to isomorphisms of G (i.e. translation, 90-degree rotation, and reflection), where S and E are the GP's square and edge sets, respectively. In other words, it is a polyomino with additional information about which squares are considered adjacent. (It is important to specify that only isomorphisms of G can make two GPs equivalent, since many distinct polyominoes are isomorphic as graphs, e.g. the two trominoes.) Squares which are adjacent in G but not in the GP may be considered to have a cut between them.

A *partial net* of an $a \times b \times c$ box is a GP (S, E) equipped with a map from S to the surface of the box which represents a folding procedure, i.e.

- Each square in S is mapped to one of the $2ab + 2ac + 2bc$ squares on the box, as well as one of 4 orientations. (Since boxes are not chiral, we do not need all 8 ways to lay one square on another. This may be thought of as coloring one side of the plane from which the GP is cut, then requiring that color to face the outside of the box.)
- No two squares in S map to the same square on the box, even if their orientations are different.
- Adjacent squares in S map to adjacent squares on the box with aligned orientations, e.g. if square a is to the left of b in G, and a and b are connected by an edge in E, then the right side of a's corresponding oriented square on the box overlaps with the left side of b's square.

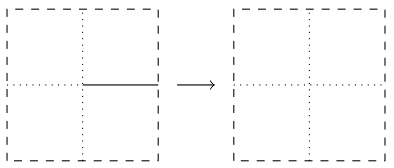
A *net* is a partial net such that all squares on the given box are covered.

We wish to answer the following question:

$\boxed{\text{Does there exist a GP which is a net of all three of the } 1 \times 1 \times 11,\ 1 \times 2 \times 7, \text{ and } 1 \times 3 \times 5 \text{ boxes?}}$

Since the boxes have surface area 46, the shared net would be a generalized 46-omino. However, only recently has the total number of (non-generalized) 46-ominoes been computed, at approximately 8.57×10^{24}. [7] Including the edges of the graph to account for all GPs of size 46 would raise this number significantly, so we should try to restrict the search space.

First, note that removing edges from a net always results in another net (as long as the graph remains connected), since edges only constrain how squares may map to the surface of the box. Also, every GP has a spanning tree, so we only need to consider the set of GPs which are also trees. Additionally, (Uehara 2008) observed that if four squares are connected by three edges to form a larger square with an internal cut, then the fourth edge may be added to the net (i.e. the last pair of adjacent squares may be glued together), as shown below, without affecting its ability to fold onto the box.

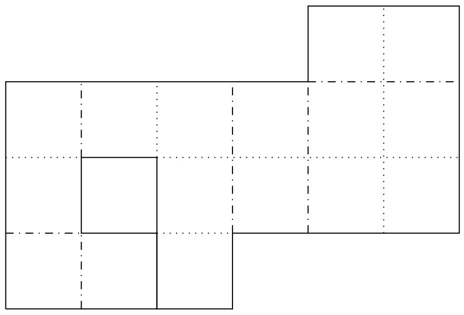


Solid lines are cut edges, dotted lines are joined edges, and dashed lines may be either.

Given a GP, we may repeat this gluing process until the configuration on the left of the above figure no longer exists. Since gluing can map many trees to the same GP, the size of the search space is further reduced. Let us call these glued trees *weakly generalized polyominoes* or *WGPs*. Then the answer to the central question above is not changed if we replace "GP" with "WGP," so from here, the definition of "partial net" (and thus of "net") is restricted to refer to WGPs only.

WGPs have no cuts which terminate on the interior, i.e. all internal cuts terminate on the edges. Then if a given WGP has an underlying polyomino which is simple (i.e. has no holes), then an internal cut would split it in two, so the WGP is the polyomino itself with all adjacent squares joined. For this reason, previous work was focused on simple polyominoes. In particular, the 2263 common nets of the $1 \times 1 \times 5$ and $1 \times 2 \times 3$ boxes listed by (Matsui and Uehara) were found by a search of simple polyominoes. However, there are WGPs whose underlying polyominoes have holes, such as the one below, which is a net of a $1 \times 2 \times 2$ box.

[7]Mason 2023, "Counting polyominoes of size 50", oeis.org/A000105/a000105_1.pdf



Dashdotted lines are joined edges which are folded to produce the box. Note the internal cut edge.

Crucially, the underlying polyomino (with the internal cut removed) is not a net of any box.

3 Algorithm

3.1 Depth-first search

The first algorithm to find common nets is based on a depth-first search function which accepts:

- $\mathcal{N}$, a set of common partial nets which do not contain each other, i.e. for all distinct $Q, R \in \mathcal{N}$, Q does not contain R.
- P, a common partial net which does not contain, and is not contained by, any $Q \in \mathcal{N}$.

The function returns the set of all common nets which contain P but do not contain any $Q \in \mathcal{N}$.

Algorithm 1 Common net search

```
1:  function Search($P$, $\mathcal{N}$)
2:      if $P$ has 46 squares then return $\{P\}$
3:      $\mathcal{R} \leftarrow \{\}$
4:      $\mathcal{M} \leftarrow \{\}$
5:      for $e \in$ {edges from a square in $P$ to a square not in $P$} do
6:          $P' \leftarrow$ Glue($P \cup \{e\}$)
7:          if $P'$ is not a common partial net then continue
8:          if $P'$ contains any $Q \in \mathcal{N} \cup \mathcal{M}$ then continue
9:          $\mathcal{R} \leftarrow \mathcal{R} \cup$ Search($P'$, $\mathcal{N} \cup \mathcal{M}$)
10:         $\mathcal{M} \leftarrow \mathcal{M} \cup \{P'\}$
        return $\mathcal{R}$
```

In other words, the algorithm repeatedly tries to extend P by one square, searches the resulting WGP P', and then eliminates P' for the rest of the search of P. The correctness of Algorithm 1 relies on the fact that every WGP which strictly contains P must contain one of the P'.

Thanks to line 8, each common net will be found exactly once during the search, so there is no duplication of effort. Further refinements should strive to maintain this property. However, depending on $|\mathcal{N} \cup \mathcal{M}|$, line 8 may be very expensive. In this case, the size is bounded by the depth of recursion (46) and the number of P' added at each level i, which is no more than $2i+2$, the maximum perimeter of an i-omino. Thus the extra time spent at each node is quadratic in the recursion depth, whereas the overlaps that are prevented decrease the number of nodes exponentially. Contrast this with a breadth-first search, where $\mathcal{N}$ would contain an entire layer of the search tree, i.e. the set of all common partial nets with i squares.

3.2 Restricting common partial nets

In theory, Algorithm 1 would suffice to find all common nets of the $1 \times 1 \times 11$, $1 \times 2 \times 7$, and $1 \times 3 \times 5$ boxes, when called with P as a single square and $\mathcal{N}$ as the empty set. However, it suffers from two major problems:

- It is called for all common partial nets P at some point during the search. While this guarantees that all common nets will be found, it would be ideal to search only a subset of common partial nets which still includes all common (total) nets.
- Checking whether one partial net A contains another B is expensive, as B must be overlaid on A with all combinations of translation, rotation, and reflection.

To ameliorate these issues, we can try to find a set $\mathcal{S} = \{S_1, \ldots, S_k\}$ of partial nets such that for all common nets P, there exists an $S \in \mathcal{S}$ such that P contains S. Then we can run Algorithm 1 k times on the inputs

$$(S_1, \{\}), (S_2, \{S_1\}), (S_3, \{S_1, S_2\}), \ldots, (S_k, \{S_1, S_2, S_3, \ldots, S_{k-1}\})$$

to obtain each net exactly once.

Edgemap consists of:

- `cx, cy : u8` -
- `rows: Vec<u64>` - array of rows of edges
  - `(rows[2*y]<<x)>>63 =` presence of edge from square `(x, y)` to `(x+1, y)` relative to top left
  - `(rows[2*y+1]<<x)>>63 =` presence of edge from `(x, y)` to `(x, y+1)`

Partial net consists of:

- Edgemap for joined edges
- Horizontal reflection of previous map
- Edgemap for cut edges

SAT formulation (2 boxes) divides each square into 4 square-edges
Variables:

- For each square-edge a of the first box and square-edge b of the second box, a variable M_{ab} for whether a maps to b
  - Preprocessing: sets of 4 variables for same square of both boxes are equivalent
  - Equivalence classes map a square of the first box to a square of the second box in one of 4 orientations
  - Fixes an orientation of the mapping: must be accounted for in symmetry breaking
- For each edge e of the first box and edge f of the second box, a variable N_{efo} for whether e maps to f in one of two orientations o
- For each edge e of the first box, a variable J_e for whether e is joined
- For each edge f of the second box, a variable K_f for whether f is joined

Constraints:

- M_{ab} defines a bijection:
  - For each square-edge a of the first box, exactly one of $\{M_{ab} \mid b \text{ square-edge of second box}\}$
  - For each square-edge b of the second box, exactly one of $\{M_{ab} \mid a \text{ square-edge of first box}\}$
- For each edge e of the first box, exactly one of $\{\neg J_e\} \cup \{N_{efo} \mid f \text{ edge of second box, } o \text{ orientation}\}$
- For each edge f of the second box, exactly one of $\{\neg K_f\} \cup \{N_{efo} \mid e \text{ edge of first box, } o \text{ orientation}\}$
- If N_{efo}, then the corresponding square-edges on either side of e and f are mapped to each other
  - $\neg N_{efo} \vee M_{e_0 f_o}$, $\neg N_{efo} \vee M_{e_1 f_{1-o}}$
- Gluing: if 3 edges around a degree-4 vertex are joined, then so is the 4th one (for both boxes)
- The joined edges of the first box form a connected graph:
  - Original: cs.stackexchange.com/questions/111410
  - Doesn't work for partial assignments: want a contradiction as soon as one part of the graph is cut off from another
  - Graph of squares and edges is planar: dual has vertices and edges
  - Primal graph is connected iff dual complement graph is acyclic
  - Cuts are dual to cycles: want a contradiction if dual non-joined edges form a cycle
  - Given (simple) graph $G = (V, E)$ and $F \subseteq E$, determine if (V, F) acyclic:
    - Order $E = \{e_1, \ldots, e_n\}$, ideally by BFS
    - For $i \in \{0, \ldots, n\}$, define $E_i = \{e_j \in E \mid j \leq i\}$, $F_i = F \cap E_i$
    - For $i \in \{0, \ldots, n\}$, $u, v \in V$, add aux var C_{iuv}: whether u is connected to v by F_i
    - For $v \in V$, C_{0vv}
    - For $i \in \{0, \ldots, n-1\}$, if $e_{i+1} = \{a_i, b_i\}$, for $u, v \in V$:
      $C_{(i+1)uv} \leftrightarrow C_{iuv} \vee (C_{iua} \wedge e_i \in F \wedge C_{ibv}) \vee (C_{iub} \wedge e_i \in F \wedge C_{iav})$
  - Simplification:
    - For $i \in \{0, \ldots, n\}$, C_{ivv}
    -

Unit prop during problem generation: “literal” can be (possibly negated) variable or truth value
Function to create a logic gate accounts for these options: can return one of its arguments back, or a truth value
Clauses must be either a sequence of (possibly negated) variables or “satisfied” (not added to final problem)
Inserting false into a clause does nothing, inserting true makes it satisfied, inserting into a satisfied clause does nothing

Symmetry breaking:
Start with arbitrary polyomino that can fold around all 3 boxes
Fix a “base squedge” on one of the 1×1 faces of box 1
Fix a representative of each of the 29 orbits of the squedges of box 2 under rotation *and reflection*
Ensure that the base squedge maps to a representative by rotating/reflecting the net around box 2
Ensure that the mapping preserves orientation by reflecting the nets around boxes 1 and/or 3 if necessary
Fix a representative of each of the 46 orbits of the squedges of box 3 under rotation
Ensure that the base squedge maps to a representative by rotating the net around box 3
Draw a path following the net from the face of box 1 containing the base squedge to the opposite face (without moving backwards)
Ensure that the path (when interpreted as a sequence of moves) is lexicographically earlier than its reversal by rotating one 1×1 face to the other if necessary
Ensure that the base squedge of box 1 points towards the opposite face when the net is unfolded by rotating the net around the face