On the Computational Complexity of Generalized Common Shape Puzzles

Mutsunori Banbara[1], Shin-ichi Minato[2], Hirotaka Ono[1], and Ryuhei Uehara[3](✉)

[1] Nagoya University, Furocho, Chikusa-ku, Nagoya, Aichi 464-8601, Japan
banbara@nagoya-u.ac.jp, ono@i.nagoya-u.ac.jp

[2] Kyoto University, Kyoto, Japan
minato@i.kyoto-u.ac.jp

[3] School of Information Science, Japan Advanced Institute of Science and Technology, Asahidai, Nomi, Ishikawa 923-1292, Japan
uehara@jaist.ac.jp

Abstract. In this study, we investigate the computational complexity of some variants of generalized puzzles. We are provided with two sets $\mathcal{S}_1$ and $\mathcal{S}_2$ of polyominoes. The first puzzle asks us to form the same shape using polyominoes in $\mathcal{S}_1$ and $\mathcal{S}_2$. We demonstrate that this is polynomial-time solvable if $\mathcal{S}_1$ and $\mathcal{S}_2$ have constant numbers of polyominoes, and it is strongly NP-complete in general. The second puzzle allows us to make copies of the pieces in $\mathcal{S}_1$ and $\mathcal{S}_2$. That is, a polyomino in $\mathcal{S}_1$ can be used multiple times to form a shape. This is a generalized version of the classical puzzle known as the common multiple shape puzzle. For two polyominoes P and Q, the common multiple shape is a shape that can be formed by many copies of P and many copies of Q. We show that the second puzzle is undecidable in general. The undecidability is demonstrated by a reduction from a new type of undecidable puzzle based on tiling. Nevertheless, certain concrete instances of the common multiple shape can be solved in a practical time. We present a method for determining the common multiple shape for provided tuples of polyominoes and outline concrete results, which improve on the previously known results in the puzzle community.

Keywords: Common shape puzzle · shape logic · least common multiple shape · NP-completeness · polyform compatibility · polypolyomino · SAT-based solver · undecidability

1 Introduction

Research on the computational complexity of puzzles and games has become increasingly important in theoretical computer science (see [13] for a comprehensive survey). Since the 1990s, numerous puzzles have been demonstrated to be NP-complete in general. These results provide a certain amount of some common intuition of the properties of the NP class. However, it has not been possible to capture certain puzzles, among which the sliding block puzzle is representative. The complexity of these kinds of puzzles had remained an open problem since Martin Gardner pointed out in the 1960s that some certain theory is required to understand such puzzles. However, after 40 years, Hearn and Demaine proposed a framework known as constraint logic, and demonstrated that these puzzles are PSPACE-complete [8,9] (some related work was also done by Flake and Baum [5]). Combinatorial reconfiguration problems have been investigated towards an understanding of the PSPACE class [10].

With the developments in theoretical computer science in the past decade, new series of puzzles have been developed in the puzzle community. In comparison to classical packing puzzles, one major property of these puzzles is that the target shape is not explicitly stated. The first example is the *symmetric shape puzzle*. This puzzle asks us to form a symmetric shape using a given set of pieces. It is extremely challenging to solve such a puzzle because we cannot be sure whether or not we are approaching the goal. This property makes the puzzle very difficult, and in fact, only a few pieces are sufficient to cause this difficulty [4]. The second example is the *anti-slide puzzle*. This puzzle asks us to interlock a given set of pieces. A typical instance asks us to pack the given pieces into a frame so that no piece can be slid in the frame. This puzzle is also difficult because the goal is not explicitly stated. The computational complexity of this puzzle was recently investigated by [11].

Fig. 1. Shape Logic (commercial product by ThinkFun).

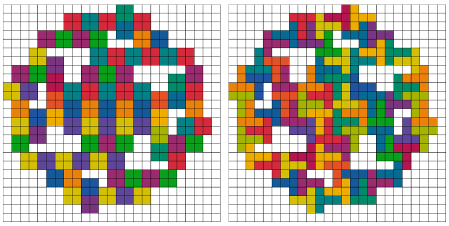


Fig. 2. Copies of a pentomino and a tetromino share a large common shape.

In this study, we focus on such a puzzle that is known as the *common shape puzzle*. Many instances of this puzzle are available in the puzzle community, and a commercial product named "Shape Logic" exists. (The authors confirmed that this puzzle was named "Top This!" in 2008 (Fig. 1), "ShapeOmetry" in 2012, and "Shape Logic" more recently by the same puzzle maker. The puzzle "Top This!" won three awards in 2008.[1] However, in this paper, we use the most recent name.) In the shape logic puzzle, we are provided with two sets $\mathcal{S}_1$ and

[1] https://www.thinkfun.com/about-us/awards/.

$\mathcal{S}_2$ of polygons. We must find a polygon X that can be formed by the pieces in $\mathcal{S}_1$ and $\mathcal{S}_2$, respectively, as in the classic silhouette puzzle Tangram. The main difference between the Tangram and the shape logic puzzle is that the target shape X is not provided, which drastically increases the difficulty of the puzzles.

Hereafter, we suppose that $\max\{|\mathcal{S}_1|/|\mathcal{S}_2|, |\mathcal{S}_2|/|\mathcal{S}_1|\}$ is bounded from above by a constant. We note that if $\mathcal{S}_1$ contains only one piece, the target shape X is fixed to it. Therefore, it is equivalent to the classic puzzle Tangram for $\mathcal{S}_2$, and it is NP-complete even if all pieces in $\mathcal{S}_1$ and $\mathcal{S}_2$ are rectangles [3].

We first demonstrate that it is polynomial-time solvable if $|\mathcal{S}_1| + |\mathcal{S}_2|$ is a constant. Subsequently, we show that the shape logic puzzle is strongly NP-complete even if all the pieces in $\mathcal{S}_1$ and $\mathcal{S}_2$ are small rectangles. We state that a rectangle in $\mathcal{S}_1 \cup \mathcal{S}_2$ is small if its size is polynomial in $(|\mathcal{S}_1| + |\mathcal{S}_2|)$.

Next, we focus on a similar puzzle named the *common multiple shape puzzle*, which has been investigated in the puzzle community for a long time under a few different names such as "polypolyomino"[2] and "polyform compatibility"[3]. We propose the term "the (least) common multiple shape" based on the term "the least common multiple," which is the corresponding Japanese name used in Japanese puzzle community.[4] An instance of this puzzle is as follows: We are provided with two polygons P and Q. The puzzle asks us to find the (smallest area) shape X that can be tiled by P, and also tiled by Q. That is, we can use any number of copies of P and Q, and find the common shape X that can be filled by only copies of P, as well as only copies of Q. This puzzle aims at finding the minimum shape, however, it is known that some pairs result in a huge solution (e.g., Fig. 2). The problem of finding a small common multiple shape of the T-pentomino and O-tetromino (shown in Fig. 2) was first proposed by Robert Wainright as a problem at the conference of games and puzzles competitions on computers[5] in 2005 and 2011. A solution with an area of 600 was found in 2011, and it was improved to 340 in 2011.[6] However, it remains open whether or not this shape with an area of 340 in Fig. 2 is the smallest.

We naturally consider the (least) common multiple shape variant of the shape logic puzzle. That is, for given sets $\mathcal{S}_1$ and $\mathcal{S}_2$ of polygons, the puzzle asks us to find a small common shape X that can be filled by copies of pieces in $\mathcal{S}_1$ (and $\mathcal{S}_2$, respectively). We show that this puzzle is undecidable even if each set of $\mathcal{S}_1$ and $\mathcal{S}_2$ contains small polyominoes. As a corollary, we also demonstrate that the following problem is undecidable: For a given set $\mathcal{S}$ of small polyominoes, determine whether a rectangle can be formed using copies of the pieces in $\mathcal{S}$.

In this study, we also present a formulation of these puzzles and verify the feasibility with a computer. We recently discovered that such puzzles can be solved by SAT-based solvers with sophisticated modeling far more efficiently

[2] https://www.iread.it/Poly/.
[3] https://sicherman.net/polycur.html.
[4] In Japan, we use 最小公倍<u>図形</u> (least common multiple <u>shape</u>) following 最小公倍<u>数</u> (least common multiple <u>number</u>).
[5] http://hp.vector.co.jp/authors/VA003988/gpcc/gpcc.htm.
[6] http://deepgreen.game.coocan.jp/MCFG/MCFG_index.htm.

than when using other methods [1]. By determining an efficient formulation of this puzzle and using a SAT-based solver, we also improve several known instances of the common multiple shapes that have been investigated in the puzzle community.

2 Preliminaries

A *polyomino* is a polygon that can be obtained by joining one or more unit squares edge to edge [7]. In this study, we only consider simple polyominoes (without holes) as polygons. (We note that even if all pieces are simple, the solution may have holes, as indicated in Fig. 2.) If a polyomino P is formed by k unit squares, we refer to P as a *k-omino*. For a specific k, we also refer to it as a *monomino*, *domino*, *tromino*, *tetromino*, *pentomino*, and *hexomino* for $k = 1, 2, 3, 4, 5, 6$, respectively.

A set $\mathcal{S}_1$ of polyominoes is said to be a set of *small* polyominoes when the maximum polyomino in $\mathcal{S}_1$ is a k-omino for $k = O(|\mathcal{S}_1|^c)$ for a positive constant c. In this case, we assume that the input size of the problem is bounded by $O(p(n))$ for a polynomial function p, where $n = |\mathcal{S}_1| + |\mathcal{S}_2|$.

In this study, we consider two problems on polyominoes. The first problem is the *shape logic puzzle*. Given two sets of $\mathcal{S}_1$ and $\mathcal{S}_2$ of small polyominoes, the puzzle asks us to form a common polyomino X using all pieces in $\mathcal{S}_1$ and all pieces in $\mathcal{S}_2$, respectively. The goal shape X is not provided. Clearly, the shape logic puzzle is in NP when all pieces are small as we can guess X and verify the feasibility of the given packing of $\mathcal{S}_1$ and $\mathcal{S}_2$ on X in polynomial time.

The second problem is the *common multiple shape puzzle*. Given two finite sets $\mathcal{S}_1$ and $\mathcal{S}_2$ of polyominoes, the puzzle asks us to form a common polyomino X with a positive area using copies of the pieces in $\mathcal{S}_1$ and copies of the pieces in $\mathcal{S}_2$, respectively. This puzzle generalizes both of the shape logic puzzle and the puzzle known as the *polypolyomino* (also referred to as *polyform compatibility*). The latter puzzle is the case in which $|\mathcal{S}_1| = |\mathcal{S}_2| = 1$. It can be extended from two sets to three or more sets naturally. (See Sect. 5 in this case).

For a finite set $\mathcal{S}$ of polyominoes, we define a set $\hat{\mathcal{S}}$ of polyominoes P such that P can be formed by copies of the pieces in $\mathcal{S}$. Clearly, $\hat{\mathcal{S}}$ is infinite and countable. That is, the common multiple shape puzzle asks whether or not $\hat{\mathcal{S}}_1 \cap \hat{\mathcal{S}}_2 \neq \emptyset$. When $\hat{\mathcal{S}}_1 \cap \hat{\mathcal{S}}_2 \neq \emptyset$, we refer to an element in $\hat{\mathcal{S}}_1 \cap \hat{\mathcal{S}}_2$ as a *common multiple shape*. Among the common multiple shapes, the smallest one is the *least common multiple shape*.

3 Complexity of Shape Logic Puzzle

In this section, we focus on the generalized shape logic puzzle. That is, given two sets $\mathcal{S}_1$ and $\mathcal{S}_2$ of polyominoes, we need to decide if all pieces in $\mathcal{S}_1$ (and in $\mathcal{S}_2$) can form a common polyomino X.

Observation 1. *When $|\mathcal{S}_1| + |\mathcal{S}_2|$ is a constant k, the generalized shape logic puzzle can be solved in time polynomial in n, where n is the total number of vertices in $\mathcal{S}_1 \cup \mathcal{S}_2$.*

Proof (Outline). We solve the problem by brute force using the same technique as in [4, Section 3]. In [4, Section 3], they presented a method for solving the symmetric assembly puzzle, which asks us to form a symmetric shape by using the pieces in a set of (general) simple polygons in polynomial time.

The generalized shape logic puzzle can be reduced to the symmetric shape puzzle in polynomial time as follows: Suppose that the generalized shape logic puzzle has a solution and the pieces in $\mathcal{S}_1$ forms a polygon P, which can also be formed by the pieces in $\mathcal{S}_2$. Then, without loss of generality, P can be placed so that its rightmost vertex v is on ∂P; that is, any point in P is not right of v. At this point, we obtain a symmetric shape by joining P and its mirror image P^R at vertex v with its mirror image on ∂P^R. That is, when the shape logic puzzle with $\mathcal{S}_1$ and $\mathcal{S}_2$ has a solution, the symmetric shape puzzle also has a solution for $\mathcal{S}_1 \cup \mathcal{S}_2$ such that the left half of the symmetric shape consists of the pieces in $\mathcal{S}_1$ and the right half of the symmetric shape consists of the pieces in $\mathcal{S}_2$. The proof in [4, Section 3] is based on brute force. Therefore, we can restrict our search to a symmetric shape that also provides a solution for the shape logic puzzle in $\mathcal{S}_1 \cup \mathcal{S}_2$. As the original brute force algorithm for the symmetric shape puzzle runs in time polynomial in n, so does our algorithm. $\square$

We note that the brute force algorithm in the proof of Observation 1 runs in $O(n^{f(k)})$ for some polynomial function f. That is, the generalized shape logic puzzle problem is fixed parameter tractable.

Theorem 2. *The shape logic puzzle is strongly NP-complete even if all pieces in $\mathcal{S}_1$ and $\mathcal{S}_2$ are small rectangles.*

Proof. According to the definition of a small polyomino, the problem is in NP. Thus, we demonstrate the NP-hardness using a reduction from the 3-partition problem. In the 3-partition problem, we are provided with a multiset of $3m$ positive integers $A = \{a_1, a_2, \ldots, a_{3m}\}$, where the a_is are bounded from above by a polynomial of m. The goal is to partition the multiset A into m triples such that every triple has the same sum $B = (\sum_{i=1}^{3m} a_i)/m$. It is known that the 3-partition problem is strongly NP-complete even if every a_i satisfies $B/4 < a_i < B/2$ [6, SP16]. Without loss of generality, we assume that $a_i > 3$ for each i and $B = 3mB'$ for a positive integer B'. Then, the set $\mathcal{S}_1$ consists of $3m$ rectangles of size $1 \times (a_i + 3m^2)$ for each $i = 1, 2, \ldots, 3m$. Furthermore, $\mathcal{S}_2$ consists of $3m$ congruent rectangles of size $m \times (B/(3m) + 3m)$. The construction can be computed in time polynomial in m.

Subsequently, we observe that $3m < B/(3m) + 3m < B/4 + 3m^2 < a_i + 3m^2$ for each i, as $3m$ pieces exist in $\mathcal{S}_1$, $a_i > 3$, and $a_i > B/4$. That is, (1) $B/(3m) + 3m$ is larger than $3m$, which is the number of long and slender rectangles in $\mathcal{S}_1$, and (2) each length $a_i + 3m^2$ cannot fit into any rectangle that is formed by the pieces in $\mathcal{S}_2$ except if a rectangle with a width of m is created. Therefore, the only means of forming the same shape using the pieces in $\mathcal{S}_1$ and the pieces in $\mathcal{S}_2$ is to form a rectangle with a size of $m \times (B + 9m^2)$ using the pieces in $\mathcal{S}_2$ and to pack long rectangles with a size of $1 \times (a_i + 3m^2)$ into this frame. The arrangement of pieces in $\mathcal{S}_1$ directly provides the solution to the original instance of the 3-partition problem. $\square$

4 Undecidability of Common Multiple Shape Puzzle

In this section, we demonstrate that the common multiple shape puzzle is undecidable. We first show that a generalized jigsaw puzzle is undecidable. In this puzzle, each edge is colored, which will be modified to polygonal shapes without color.

4.1 Undecidability of a Generalized Jigsaw Puzzle

We first consider a generalized jigsaw puzzle. We borrow several notions from [2]. Each piece is a square with four edges and has its own color. We denote the set of colors as $C = \{0, 1, 2, \ldots, c, \bar{1}, \bar{2}, \ldots, \bar{c}\}$. In our jigsaw puzzle, we tile the pieces into a rectangular frame so that each edge is shared by two adjacent pieces with colors i and $\bar{i}$, except for the boundary of the frame. A special color 0 exists, which should match to the frame. That is, when we tile the pieces, the outer boundary has the color 0, and no inside edge has the color 0.

In our jigsaw puzzle, we are allowed to use copies of a piece in $\mathcal{S}$ multiple times, which is the significant difference between our puzzle and that in [2]. Therefore, for a given finite set $\mathcal{S}$, we have infinitely countable means of tiling the pieces. Subsequently, the jigsaw puzzle problem is defined as follows:

Input: A set $\mathcal{S}$ of unit square pieces such that each piece has four colors in C on its four edges.

Output: Decide if there is a polyomino region R such that R can be tiled by copies of pieces in $\mathcal{S}$ in which each inner edge is shared by two adjacent pieces of colors i and $\bar{i}$ (with $i > 0$), and each edge on the boundary ∂R has the color 0.

We first present the following lemma. Intuitively, the rectangle in the following lemma will be used as a template for the other set of pieces.

Lemma 1. *There exists a finite set $\mathcal{S}$ of jigsaw puzzle pieces such that the area R is tiled by copies of the pieces in $\mathcal{S}$ with the boundary color 0 along ∂R if and only if R is a rectangle with a size of at least 3×3.*

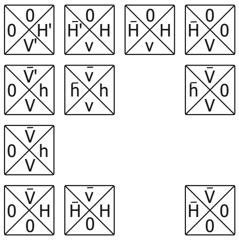


Fig. 3. Jigsaw puzzle in a rectangle.

Proof. We consider the set of 11 jigsaw puzzle pieces depicted in Fig. 3. For ease of reference, we use some letters such as H, V, etc. instead of the numbers 1, 2, etc. in the figure. As every piece contains H,

$\bar{H}$, H', $\bar{H}'$, h, or $\bar{h}$, without loss of generality, we can assume that one piece is placed so that its $\bar{H}$, $\bar{H}'$, or $\bar{h}$ is on its left, and all the pieces are then aligned in the same direction, as indicated in the figure. The boundary of the jigsaw puzzle is labeled by 0. We observe that we cannot form a rectangle with a size of $2 \times n$ (and $n \times 2$) for any n because V' and $\bar{V}$ (and H' and $\bar{H}$) do not match. Furthermore, we observe that an edge is colored by h or v if and only if it is not incident to a vertex on the boundary. In particular, we cannot create a corner with the angle 270°; to achieve this, we must place one "corner boundary" piece inside, which results in the color 0 being inside the shape, and this is not permitted. □

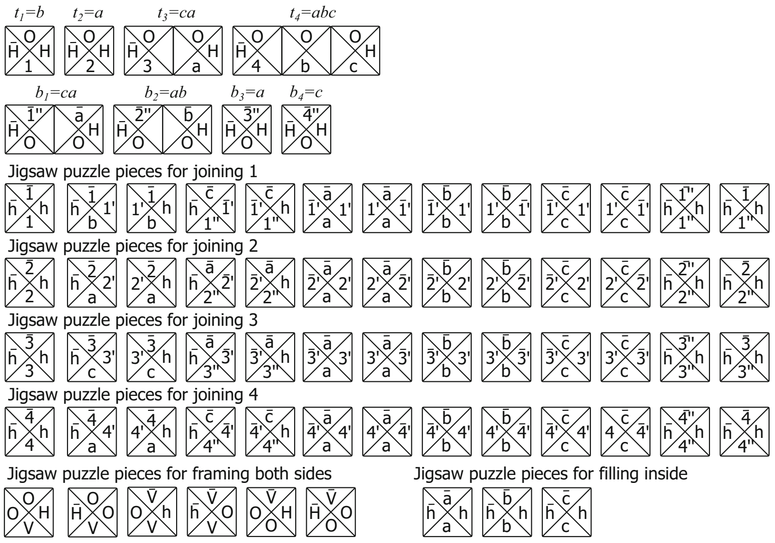
We show that our jigsaw puzzle problem is undecidable.

Lemma 2. *There exists a finite set $\mathcal{S}$ of pieces of the jigsaw puzzle such that the jigsaw puzzle is undecidable.*

Proof (Outline). We present a polynomial-time reduction from the following Post Correspondence (PC) problem:

Input: A sequence of pairs of strings $s_1 = (t_1; b_1), s_2 = (t_2; b_2), \ldots, s_n = (t_n, b_n)$. We define $T(s_i) = t_i$,$B(s_i) = b_i$ for a pair $s_i = (t_i; b_i)$.
Question: Decide if there exists a sequence of pairs $s_{i_1}, s_{i_2}, s_{i_3}, \ldots, s_{i_k}$ of strings such that $T(s_{i_1})T(s_{i_2})T(s_{i_3}) \cdots T(s_{i_k}) = B(s_{i_1})B(s_{i_2})B(s_{i_3}) \cdots B(s_{i_k})$.

Let Σ be an alphabet, namely the set of letters that is used in the sequence. We note that we can use each pair s_i can be used any number of times. It is well known that the PC problem is undecidable even if $|\Sigma|$ is a constant [12].



$t_1=b$
$t_2=a$
$t_3=ca$
$t_4=abc$
$b_1=ca$
$b_2=ab$
$b_3=a$
$b_4=c$
Jigsaw puzzle pieces for joining 1
Jigsaw puzzle pieces for joining 2
Jigsaw puzzle pieces for joining 3
Jigsaw puzzle pieces for joining 4
Jigsaw puzzle pieces for framing both sides
Jigsaw puzzle pieces for filling inside

Fig. 4. A reduction from the PC problem to the jigsaw puzzle problem.

We demonstrate the reduction by using a concrete example $\Sigma = \{a, b, c\}$, $s_1 = (b; ca), s_2 = (a; ab), s_3 = (ca; a), s_4 = (abc; c)$ (Fig. 4). We prepare one piece, one piece, two pieces, and three pieces of jigsaw puzzle for each string $t_1 = b$, $t_2 = a$, $t_3 = ca$, and $t_4 = abc$, respectively. We set each string to be uniquely constructible: two pieces for $t_3 = ca$ have their own color (distinct from any other color) between them, and three pieces for $t_4 = abc$ have their own two colors between a and b, and b and c (these are blank in Fig. 4). The top color is 0, the left color is $\bar{H}$, and the right color is H for each piece. (As in the proof of Lemma 1, we regard certain letters as numbers greater than n). Hereafter, we consider these strings as rectangular pieces that are represented by sizes of 1×1, 1×1, 2×1, and 3×1, respectively. The leftmost bottom color of the rectangular piece t_i is the color i, which is referred to as the *ID* of this rectangular piece. The color of the other edge corresponds to the letter in the string. That is, the second and the third pieces of the rectangle representing $t_4 = abc$ have the colors b and the color c, respectively. (We regard these letters as unique numbers in the color set C. As the size of the alphabet Σ is a constant, regarding these letters as numbers has no influence on our arguments).

Subsequently, we prepare two, two, one, and one pieces for the strings $b_1 = ca$, $b_2 = ab$, $b_3 = a$, and $b_4 = c$, respectively. As with the strings t_i, b_1, b_2, b_3, b_4 each corresponds to a rectangular piece with a size of 2×1, 2×1, 1×1, and 1×1, respectively. The bottom color is 0, the left color is $\bar{H}$, and the right color is H for these rectangular pieces. The top colors of the rectangular piece are represented by the letters, except for the leftmost edge, which has the color ID $\bar{i''}$ for the string b_i.

Next, we prepare to join two pieces with the IDs i and i''. Hereafter, we use (c_u, c_b, c_l, c_r) to denote the top color c_u, bottom color c_b, left color c_l, and right color c_r of a piece. Furthermore, we assume that the top letter of t_i is x_i and the top letter of b_i is y_i. We first prepare a piece with colors $(\bar{i}, i, \bar{h}, h)$, which is a wire of the ID in the vertical direction. We also prepare a piece with colors $(\bar{i}, x_i, \bar{h}, i')$, which turns the ID to the right, and a piece with colors $(\bar{i}, x_i, \bar{i'}, h)$, which turns the ID to the left. The ID is turned to the right or left using one of these pieces and runs horizontally. The prime symbol $'$ means that the ID turns once. Thereafter, we prepare two pieces with the colors $(\bar{y_i}, i'', \bar{h}, i')$ and $(\bar{y_i}, i'', \bar{i'}, h)$ to turn the ID downwards. In this case, the symbol $''$ means that the ID turns twice. Furthermore, we prepare a piece with the color $(\bar{j}, j, \bar{i'}, i')$ for each letter $j \in \Sigma$ to propagate the ID in the horizontal direction. We also add a piece with the color $(\bar{i''}, i'', \bar{h}, h)$ to pass the ID downwards after turning twice. In a special case, an ID can directly move from top to bottom without turning. We prepare a piece with the color $(\bar{i}, i'', \bar{h}, h)$ to deal with this case. Thus, we prepare a total of eight pieces for the IDs i and i''.

Subsequently, we prepare pieces to form the left and right sides of the rectangular frame. We prepare six pieces with the colors $(0, V, 0, H)$, $(0, V, \bar{H}, 0)$, $(\bar{V}, V, 0, h)$, $(\bar{V}, V, \bar{h}, 0)$, $(\bar{V}, 0, 0, H)$, and $(\bar{V}, 0, \bar{H}, 0)$. Finally, we prepare pieces $(\bar{j}, j, \bar{h}, h)$ for each $j \in \Sigma$ to fill the holes in the frame.

We prepare $\sum_{i=1}^{n}(|t_i| + |b_i|) + 8n + 6 + |\Sigma|$ pieces in total. Therefore, the jigsaw puzzle can be constructed in time polynomial in the size of a given instance of the PC problem.

We demonstrate that the instance $s_1 = (t_1; b_1), s_2 = (t_2; b_2), \ldots, s_n = (t_n, b_n)$ of the PC problem has a solution if and only if the jigsaw puzzle has a solution such that a rectangular area R is filled with copies of the pieces with the color 0 only on ∂R.

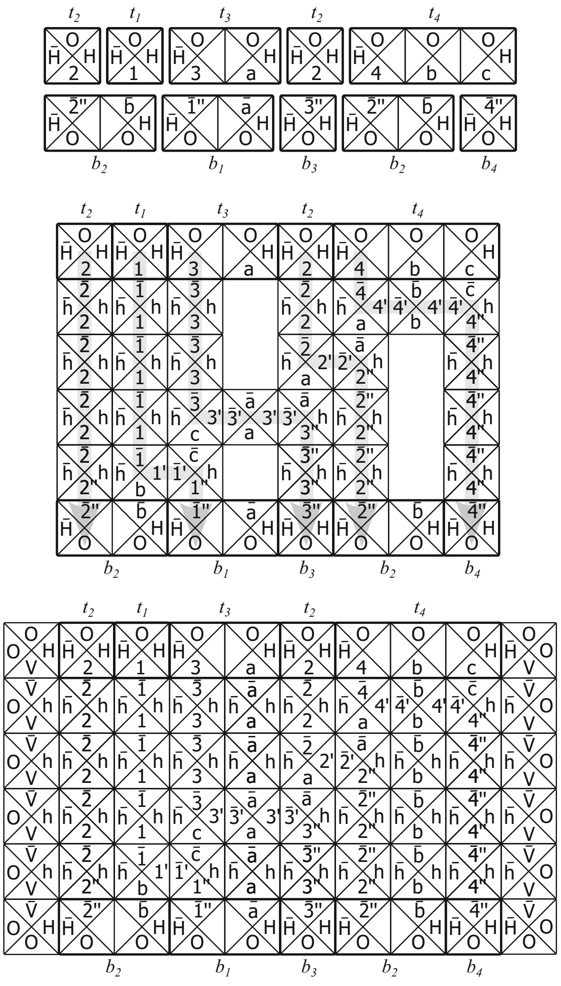


Fig. 5. Construction of a solution of the jigsaw puzzle from a solution of the instance of the PC problem.

We first assume that the sequence $s_{i_1}, \ldots, s_{i_k}$ is a solution, from which we construct a rectangular shape using the set of pieces of the jigsaw puzzle. We use $s_1 = (b; ca), s_2 = (a; ab), s_3 = (ca; a), s_4 = (abc; c)$ as an example (Fig. 5). Intuitively, we verify that two corresponding IDs are joined by a zig-zag path with two (or zero) turns, these zig-zag paths do not cross one another, and the corresponding letters are joined by vertical matching pieces.

We first align the rectangular pieces $t_{i_1}, t_{i_2}, \ldots, t_{i_k}$ on the top row and $b_{i_1}, b_{i_2}, \ldots, b_{i_k}$ on the bottom row following the solution $s_{i_1}, \ldots, s_{i_k}$ of the PC problem. (As a reminder, each string t_i (and s_i) produces unique rectangular pieces.) Thus, we obtain 0s on the top and bottom boundaries, and we can join all rectangular pieces by matching h and $\bar{h}$. Subsequently, we join all corresponding pairs of IDs using the prepared pieces. When the gap between the top and bottom rows is sufficiently large, each joining path for each ID i can be created in one of the following manners:

(1) The pair of the corresponding IDs i and i'' in the same column is directly joined vertically,
(2) When the ID i'' of b_i is left of the ID i of t_i, the ID i first moves down vertically, turns left once, moves horizontally, turns right once, and moves downwards to the ID i'', and
(3) When the ID i'' of b_i is right of the ID i of t_i, the ID i first moves down vertically, turns right once, moves horizontally, turns left once, and moves downwards to the ID i''.

Any of these procedures can be performed using the pieces prepared as above. Note that in the case (2), the first letter x_i of the string t_i appears at the corner when we use the piece $(\bar{i}, x_i, \bar{i}', h)$ is used to turn left, and the first letter y_i of the string b_i appears at the corner when $(\bar{y}_i, i'', \bar{h}, i')$ is used. In the case (3), the pieces $(\bar{i}, x_i, \bar{h}, i')$ and $(\bar{y}_i, i'', \bar{i}', h)$ are used for this purpose.

Following all the above steps, we can observe that each corresponding pair of IDs is joined by either a straight vertical path in case (1) or a zig-zag path with two turns in cases (2) or (3). Moreover, the ith letter in the common string that is produced by the sequence $s_{i_1}, \ldots, s_{i_k}$ appears on all the horizontal edges of the ith piece (from left), except its boundary and pieces on the vertical line that join two corresponding IDs. At this holds even for the holes, we can fill all of the holes using the pieces that have been prepared for filling. Finally, we can complete the frame by arranging the pieces that have been prepared for the frame with the color 0 on the boundary.

We assume that the jigsaw puzzle has a solution. The pieces that correspond to t_i and b_i form the respective rectangles as they have their unique colors. As all pieces have the color h or $\bar{h}$, therefore, every piece is arranged so that $\bar{h}$ appears on the left side and h appears on the right side. Because the color 0 matches no other colors, the rectangles for t_i and the corner pieces of color 0 on the upper edges are arranged on the top row, as are the rectangles for b_i and the corner pieces of color 0 on the lower edges. We need to form a rectangle using the pieces of the color 0 on the left or right side. (Although it may appear that we can form any polyomino other than rectangles, we cannot create any concave corner of 270° because an edge with the color 0 cannot be placed inside the polyomino).

According to the color properties, the ID color of each rectangle corresponding to t_i should be connected to the ID color of each rectangle for b_i, and these k paths cannot cross. If a path has no turn, it is necessary to use some copies of the piece with the color $(\bar{i}, i, \bar{h}, h)$, one copy of the piece $(\bar{i}, i'', \bar{h}, h)$, and some copies of the piece $(\bar{i''}, i'', \bar{h}, h)$. If a path has turns, the only possible solution is that the color i of t_i starts vertically, is changed to i' after one 90° turn, moves horizontally, is changed to i'' after one 90° turn, and moves down to the piece in the rectangle corresponding to b_i. The colors of t_i and b_i appear at each turn on a horizontal edge. Thus, the remaining holes should be packed using the pieces with the color $(\bar{j}, j, \bar{h}, h)$ for the matching color j, and each pair of IDs of t_i and b_i should match.

Therefore, when the jigsaw puzzle has a solution, the pieces form a rectangle, the pieces for t_i are arranged on the top, the pieces for b_i are arranged on the bottom, the corresponding pairs of IDs of t_i and b_i match, and a consistent letter is obtained along each vertical line of the pieces. Thus, s_i can be arranged following the sequence, and the same sequence of letters that is produced by the sequences of t_i and b_i can be obtained, which provides a solution to the PC problem, thereby completing the proof. □

4.2 Undecidability of the Common Multiple Shape Puzzle

We now turn to the common multiple shape puzzle. Lemmas 1 and 2 imply the following Theorem.

Theorem 3 *For two finite sets $\mathcal{S}_1$ and $\mathcal{S}_2$ of small polyominoes, the common multiple shape puzzle for $\mathcal{S}_1$ and $\mathcal{S}_2$ is undecidable.*

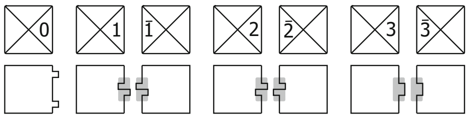


Fig. 6. Colored jigsaw piece for a polyomino. Each color i corresponds to a zigzag pattern that represents the integer i in the binary system. The color $\bar{i}$ is its negative.

Proof (Outline). We first demonstrate how to represent each piece of the jigsaw puzzle in Lemmas 1 and 2 using a small polyomino. The basic concept is explained in [3, Fig. 7]. Each color is represented by its original zig-zag pattern. See Fig. 6 for an example of the representation. Using the binary system, the size of the polyomino is $O((\log |C|)^2)$, where C is the set of colors.

We consider the set $\mathcal{S}_1$ of jigsaw pieces in Lemma 1 and the set $\mathcal{S}_2$ of jigsaw pieces in Lemma 2. Different colors can be used for each set, except the common color 0. Subsequently, according to Lemma 1, a solution to the common multiple shape puzzle is a shape that corresponds to a rectangle. Moreover, according to Lemma 2, whether it can be constructed using the pieces in $\mathcal{S}_2$ is undecidable. The number of colors used in $\mathcal{S}_1 \cup \mathcal{S}_2$ is linear with the size n of the input. Thus, each polyomino has an area of $O((\log n)^2)$, which means that it is small. This completes the proof. □

5 Improved Solutions for Common Multiple Shapes

In this section, we provide a brief formulation of generalized common shape puzzles. The rep-tile problem,[7] which is a type of packing puzzle on polyominoes, has been formulated and examined using several different computer methods [1] recently. In [1], the authors demonstrated that the rep-tile problem can be formulated in a natural form that can be handled using various methods. They compared a well-known puzzle solver, a few algorithms based on dancing links, an MIP solver, and a SAT-based solver with respect to for solving the packing puzzles. In [1], the authors concluded that the SAT-based solver is significantly faster than the other methods. The common shape puzzle has similar properties to the rep-tile problem. Therefore, we examined several instances of the common shape puzzle that are available online,[8] and improved some of the known results by using the SAT-based solver used in [1].

For example, the previous best known shape for F5Q4T4 on https://www.iread.it/Poly/ was a 760-omino, and our new shape is only 160-omino (Fig. 7). (Here, "F5Q4T4" means that this problem asks for finding a common shape of using copies of F-pentomino, Q-tetromino, and T-tetromino, respectively, which are commonly used in the puzzle society. See https://www.iread.it/lz/pttomini.html for details.) The previous best known shape for T5L4Q4 on https://sicherman.net/n445com/n445com.html, which was a 560-omino, is improved to 480-omino (Fig. 8). In addition to them, we improved the following cases: The previous best known shapes for I5P5T5, I5P5Z5, L5P5X5, and P5U5V5 on https://sicherman.net/rosp/triplep.html were 120-omino, 200-omino, 400-omino, and 160-omino, respectively. We obtain new better shapes of 110-omino for I5P5T5, 150-omino for I5P5Z5, 360-omino for L5P5X5, 120-omino for P5U5V5, respectively.

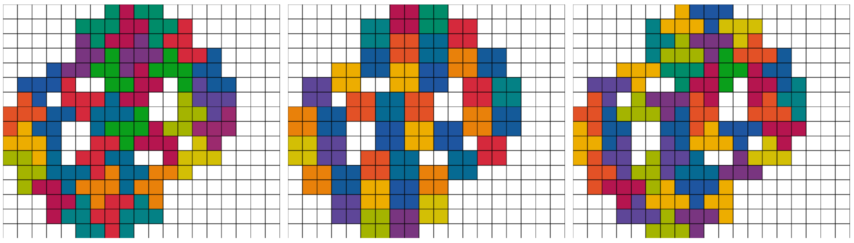


Fig. 7. Tiling patterns for F5Q4T4 improved from 760-omino to 160-omino.

Two main differences exist between the formulations of the common shape puzzle and the packing puzzle in [1]. The first one is that the goal shape is not

[7] A polygon P is called a *rep-tile* if it can be divided into congruent polygons with each other similar to P.

[8] https://www.iread.it/Poly/ and https://sicherman.net/polycur.html.

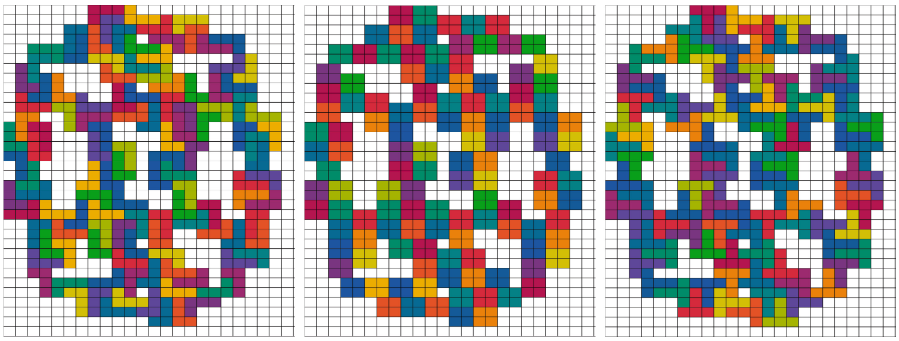


Fig. 8. Tiling patterns for T5L4Q4 improved from 560-omino to 480-omino.

provided in the common shape puzzle, whereas it is provided in the packing puzzle. The second is that we must create a common (or congruent) shape using two different sets $\mathcal{S}_1$ and $\mathcal{S}_2$ of pieces in the common shape puzzle, whereas we have only one set of pieces in the packing puzzle.

To address the first point, we fix the bounding box of the goal shape. We first fix the number of pieces (or $|\mathcal{S}_1|$ and $|\mathcal{S}_2|$), and we attempt to create possible bounding boxes that contains these pieces.

In the packing puzzle, we can assume that each unit square of a goal shape is covered exactly once by a piece. However, in the common shape puzzle, each unit square of a bounding box is covered by either 0 or 2 pieces. Moreover, when the square is covered by 2 pieces, these should be in $\mathcal{S}_1$ and $\mathcal{S}_2$.

We can modify the formulation of the packing puzzle in [1] to that for the common shape puzzle using these concepts. Furthermore, it is straightforward to extend the problem from two sets $\mathcal{S}_1$ and $\mathcal{S}_2$ to three sets $\mathcal{S}_1$, $\mathcal{S}_2$, and $\mathcal{S}_3$ (and more).

6 Concluding Remarks

We have considered the computational complexities of generalized common shape puzzles, in which the goal shapes are not provided. The puzzle is tractable when the number of pieces is a constant; however, it is strongly NP-complete even if the piece sets consist of small rectangles. Moreover, if we are allowed to use the copies of the pieces repeatedly, the problem becomes undecidable. It is possible to formulate the puzzle for several different solvers in a natural form, and we improved some known records for concrete instances using a SAT-based solver. However, we have not yet succeeded in confirming that the results are the minimum solutions. For example, we verified the pattern in Fig. 2 for each boundary box with a size of $i \times \lfloor 625/i \rfloor$ using $1 \leq i \leq 25$ and confirmed that there are no smaller patterns in these boundary boxes. However, this does not imply that the pattern in Fig. 2 is the smallest area pattern. Thus, efficient searching for the minimum solution remains open. We have only considered the polyominoes in this study, and thus, the extension to general polygons is a natural topic for future work.

References

1. Banbara, M., et al.: Solving rep-tile by computers: performance of solvers and analyses of solutions. arXiv:2110.05184 (2021)
2. Bosboom, J., Demaine, E.D., Demaine, M.L., Hesterberg, A., Manurangsi, P., Yodpinyanee, A.: Even $1 \times n$ edge matching and jigsaw puzzles are really hard. J. Inf. Process. **25**, 682–694 (2017)
3. Demaine, E.D., Demaine, M.L.: Jigsaw puzzles, edge matching, and polyomino packing: connections and complexity. Graphs Combin. **23**(Suppl 1), 195–208 (2007). https://doi.org/10.1007/s00373-007-0713-4
4. Demaine, E.D., et al.: Symmetric assembly puzzles are hard, beyond a few pieces. Comput. Geom.: Theory Appl. **90**, 101648, 1–11 (2020). https://doi.org/10.1016/j.comgeo.2020.101648
5. Flake, G.W., Baum, E.B.: Rush hour is PSPACE-complete, or "why you should generously tip parking lot attendants". Theoret. Comput. Sci. **270**, 895–911 (2002)
6. Garey, M.R., Johnson, D.S.: Computers and Intractability—A Guide to the Theory of NP-Completeness. Freeman (1979)
7. Golomb, S.W.: Polyominoes. Princeton University Press, Princeton (1994)
8. Hearn, R.A., Demaine, E.D.: Games, Puzzles, and Computation. A K Peters Ltd. (2009)
9. Hearn, R.A., Demaine, E.D.: PSPACE-completeness of sliding-block puzzles and other problems through the nondeterministic constant logic model of computation. Theoret. Comput. Sci. **343**(1–2), 72–96 (2005)
10. Ito, T., et al.: On the complexity of reconfiguration problems. Theoret. Comput. Sci. **412**, 1054–1065 (2011)
11. Minamisawa, K., Uehara, R., Hara, M.: Mathematical characterizations and computational complexity of anti-slide puzzles. Theor. Comput. Sci. **939**, 216–226 (2023). https://doi.org/10.1016/j.tcs.2022.10.026
12. Post, E.L.: A variant of a recursively unsolvable problem. Bull. Amer. Math. Soc. **52**, 264–268 (1946). https://doi.org/10.1090/S0002-9904-1946-08555-9
13. Uehara, R.: Computational complexity of puzzles and related topics. Interdisc. Inf. Sci. **29**(2), 119–140 (2023). https://doi.org/10.4036/iis.2022.R.06