Noname manuscript No.
(will be inserted by the editor)

Complexity of Tiling a Polygon with Trominoes or Bars

Takashi Horiyama · Takehiro Ito · Keita Nakatsuka · Akira Suzuki · Ryuhei Uehara

Received: date / Accepted: date

Abstract We study the computational hardness of the tiling puzzle with polyominoes, where a polyomino is a rectilinear polygon (i.e., a polygon made by connecting unit squares.) In the tiling problem, we are given a rectilinear polygon P and a set S of polyominoes, and asked whether P can be covered without any overlap using translated copies of polyominoes in S. In this paper, we focus on trominoes and bars as polyominoes; a tromino is a polyomino consisting of three unit squares, and a bar is a rectangle of either height one or width one. Notice that there are essentially two shapes of trominoes, that is, I-shape (i.e., a bar) and L-shape. We consider the tiling problem when restricted to only L-shape trominoes, only I-shape trominoes, both L-shape and I-shape trominoes, or only two bars. In this paper, we prove that the tiling problem

T. Horiyama
Graduate School of Science and Engineering, Saitama University, 255 Shimo-Okubo, Sakura, Saitama 338-8570, Japan.
E-mail: horiyama@al.ics.saitama-u.ac.jp

T. Ito
Graduate School of Information Sciences, Tohoku University, 6-6-05 Aramaki Aza Aoba, Aoba-ku, Sendai, Miyagi 980-8579, Japan.
CREST, JST, 4-1-8 Honcho, Kawaguchi, Saitama 332-0012, Japan.
E-mail: takehiro@ecei.tohoku.ac.jp

K. Nakatsuka
Faculty of Engineering, Saitama University, 255 Shimo-Okubo, Sakura, Saitama 338-8570, Japan.
E-mail: nakatsuka@al.ics.saitama-u.ac.jp

A. Suzuki
Graduate School of Information Sciences, Tohoku University, 6-6-05 Aramaki Aza Aoba, Aoba-ku, Sendai, Miyagi 980-8579, Japan.
CREST, JST, 4-1-8 Honcho, Kawaguchi, Saitama 332-0012, Japan.
E-mail: a.suzuki@ecei.tohoku.ac.jp

R. Uehara
School of Information Science, JAIST, 1-1 Asahidai, Nomi, Ishikawa 923-1292, Japan.
E-mail: uehara@jaist.ac.jp

remains NP-complete even for such restricted sets of polyominoes. All reductions are carefully designed so that we can also prove the #P-completeness and ASP-completeness of the counting and the another-solution-problem variants, respectively. Our results answer two open questions proposed by Moore and Robson (2001) and Pak and Yang (2013).

Keywords tiling problem · polyominoes · NP-complete · #P-complete · ASP-complete

1 Introduction

Since Golomb [5] introduced the notion of polyominoes, many puzzles with polyominoes have been considered and solved from the viewpoint of theoretical computer science. In this paper, we focus on the tiling puzzle which is one of the most famous and classical puzzles with polyominoes.

A *polyomino* is a rectilinear polygon (i.e., a polygon made by connecting unit squares along their edges.) Then, we study the problem of tiling a rectilinear polygon with polyominoes, defined as follows: Given a rectilinear polygon P and a set S of polyominoes, determine whether P can be covered without any overlap using translated copies of polyominoes in S. (See Fig. 1 and 2 as examples.) To avoid the confusion, while both P and polyominoes in S are rectilinear polygons, P is called a polygon and each rectilinear polygon in S is called a polyomino. In the rest of this paper, we denote by (P, S) an instance of the tiling problem.

1.1 Known results

The tiling problem has been studied for several decades, and there are many known results for the problem. We here summarize some of them classified by whether a given polygon P can have a hole or not. In the following, we simply denote by $x \times y$ the polyomino of a rectangle with x rows and y columns.

1.1.1 Polygons without any hole

For the case where a given polygon P contains no hole, the tiling problem is solvable in polynomial time if $S = \{1 \times 2,\ 2 \times 1,\ 1 \times 3,\ 3 \times 1\}$ [11] or $S = \{k \times k',\ m \times m'\}$ for any fixed positive integers k, k', m, m' [8,13].

It is a natural question to ask whether the tiling problem (without any hole in P) is solvable in polynomial time or not, if S contains only a constant number of polyominoes. Pak and Yang [10] negatively answered the question: there exists a set S consisting of 10^6 polyominoes such that the tiling problem is NP-complete and its counting variant is #P-complete.

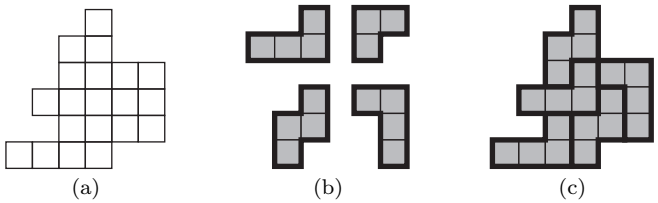

Fig. 1 A Yes-instance (P, S) of the tiling problem consisting of (a) a rectilinear polygon P and (b) a set S of polyominoes. (c) Tiling P with polyominoes in S.

Fig. 2 A No-instance (P, S) of the tiling problem consisting of (a) a rectilinear polygon P and (b) a set S of a polyomino.

Fig. 3 (a) L-trominoes and (b) I-trominoes.

1.1.2 Polygons with holes

For the case where a given polygon P may contain holes, the complexity has been analyzed more precisely.

First, consider the case where S contains only bars (i.e., $1 \times m$ or $m \times 1$ rectangles for a positive integer m.) If S consists of dominoes, that is, $S = \{1 \times 2,\ 2 \times 1\}$, then the tiling problem can be solved in polynomial time by reducing it to the matching problem on bipartite graphs. Rémila [12] gave a linear-time algorithm for $S = \{1 \times 2,\ 2 \times 1,\ 1 \times 3,\ 3 \times 1\}$. On the other hand, Beauquier et al. [1] showed that the tiling problem is NP-complete even if $S = \{1 \times m,\ k \times 1\}$, for any fixed integers $m \geq 2$ and $k \geq 3$.

Second, consider the case where S contains polyominoes which are not necessarily bars. Such a polyomino with the smallest size is a *tromino*, that is, a polyomino consisting of three unit squares. (See Fig. 3.) Trominoes have only two shapes essentially; four polyominoes in Fig. 3(a) are called L-trominoes, and two polyominoes in Fig. 3(b) are called I-trominoes. Let L and I be the sets of all L-trominoes and I-trominoes, respectively. The result by Beauquier et al. [1], mentioned above, implies that the tiling problem is NP-complete if $S = I$. Moore and Robson [9] showed that the tiling problem is NP-complete if $S = L$. They also showed that if $S = L \cup \{2 \times 2\}$, the tiling problem is NP-complete and its counting variant is #P-complete. However, the complexity status was open for $S = L \cup I$.

1.2 Our contribution

In this paper, we precisely analyze the computational hardness of the tiling problem and its variants for polygons P with holes when the polyominoes in S are restricted to trominoes or only two bars. We denote by L-TILING, I-TILING and LI-TILING the tiling problem when restricted to $S = L$, $S = I$ and $S = L \cup I$, respectively; we also denote by 2BAR-TILING the tiling problem when restricted to $S = \{1 \times m,\ k \times 1\}$ for any fixed integers $m \geq 2$ and $k \geq 3$.

In this paper, we prove the NP-completeness of L-TILING, I-TILING, LI-TILING and 2BAR-TILING. As we mentioned in Section 1.1.2, the NP-completeness of L-TILING, I-TILING and 2BAR-TILING are already known. However, our proofs are different from the known ones; we will construct a polynomial-time reduction from ONE-IN-THREE 3SAT (which will be defined later) to each of the four problems. We emphasize that our reductions are carefully designed so that (valid) tilings of the corresponding instance have the one-to-one correspondence with (valid) solutions of a given instance of ONE-IN-THREE 3SAT. Therefore, our reductions also prove that the counting variants and the another-solution-problem variants of the four problems are #P-complete and ASP-complete, respectively. Note that the another-solution-problem variant of a problem is defined as follows [15]: Given an instance of the problem and its (valid) solution s, find a solution s' of the instance other than s if exists.

We note that, while many complexity results are known for the tiling problem, there are only a few results about #P-completeness. Indeed, showing the #P-completeness of L-TILING was an open problem proposed by Moore and Robson [9]. In addition, Pak and Yang [10] conjectured that the counting variant of the tiling problem is #P-complete even if S consists of only a small number of rectangles. Our results answer both of them.

An early version of this paper has been presented in [7].

2 Reductions

To prove our results, we introduce a graph orientation problem, called the ONE-IN-THREE GRAPH ORIENTATION problem (1-IN-3 GO for short), and give reductions from ONE-IN-THREE 3SAT to our problems via 1-IN-3 GO.

We first define the ONE-IN-THREE 3SAT problem, which is known to be NP-complete [14], #P-complete [2] and ASP-complete [15].

Definition 1 ONE-IN-THREE 3SAT.

We are given a conjunctive normal form (CNF) formula φ consisting of m clauses with n variables such that each clause C_j contains three literals (i.e., variables or their negations.) Then, ONE-IN-THREE 3SAT is to determine whether there exists a satisfying truth assignment to the variables such that each clause in φ has exactly one true literal.

For example, $\varphi = (x \vee y \vee \overline{z}) \wedge (x \vee z \vee w)$ is a yes-instance, because there is a satisfying truth assignment $(x, y, z, w) =$ (False, False, False, True). On the

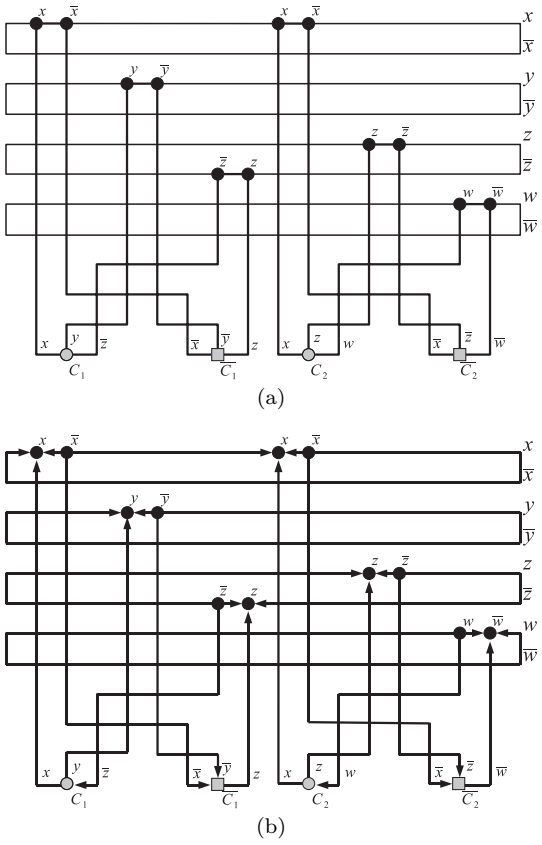

Fig. 4 (a) Instance G_φ of 1-IN-3 GO corresponding to an instance $\varphi = (x \vee y \vee \overline{z}) \wedge (x \vee z \vee w)$ of ONE-IN-THREE 3SAT, and (b) a valid orientation of G_φ corresponding to a satisfying truth assignment $(x, y, z, w) =$ (False, False, False, True).

other hand, $\varphi' = (x \vee y \vee z) \wedge (\overline{x} \vee \overline{y} \vee \overline{z})$ is a no-instance, even though there is a truth assignment which satisfies all clauses in φ'.

2.1 Reduction to 1-IN-3 GO

We now introduce the ONE-IN-THREE GRAPH ORIENTATION problem, as follows:

Definition 2 1-IN-3 GO (ONE-IN-THREE GRAPH ORIENTATION).

We are given an undirected 3-regular graph $G = (V, E)$ such that V is partitioned into three (disjoint) node-subsets V_ℓ, V_C and $V_{\overline{C}}$; the nodes in V_ℓ, V_C, and $V_{\overline{C}}$ are called *literal nodes*, *clause nodes*, and *negated-clause nodes*, respectively. Then, 1-IN-3 GO is to determine whether there exists an orientation of G (i.e., an assignment of directions to all edges in G) such that

(1) every literal node in V_ℓ has in-degree either zero or three;
(2) every clause node in V_C has exactly one inbound edge, i.e., in-degree one; and
(3) every negated-clause node in $V_{\overline{C}}$ has exactly one outbound edge, i.e., out-degree one.

For example, the undirected 3-regular graph in Fig. 4(a) is a yes-instance, because it has a valid orientation as illustrated in Fig. 4(b), where each literal node in V_ℓ is depicted by a black circle, each clause node in V_C by a gray circle, and each negated-clause node in $V_{\overline{C}}$ by a gray square.

In this subsection, we prove the following theorem.

Theorem 1 1-IN-3 GO *and its counting and another-solution-problem variants are NP-complete, #P-complete and ASP-complete, respectively.*

Observe that 1-IN-3 GO is in NP, in #P and in ASP. Therefore, as a proof of Theorem 1, we construct a polynomial-time reduction from ONE-IN-THREE 3SAT to 1-IN-3 GO. We note that, to prove the #P-hardness and ASP-hardness, our reduction satisfies that valid orientations to our corresponding instance of 1-IN-3 GO have the one-to-one correspondence with satisfying truth assignments to a given instance of ONE-IN-THREE 3SAT. In addition, although the corresponding graph of 1-IN-3 GO is not required to be planar, we will embed it (with edge-crossings but without any edge-overlap) on a grid whose size is bounded by a polynomial size (see Fig. 4 as an intuitive illustration); this embedding will be useful for the reductions from 1-IN-3 GO to the tiling problem.

2.1.1 Construction

Let φ be a given 3-CNF formula consisting of m clauses with n variables. Then, we construct an undirected 3-regular graph G_φ (with an embedding on a grid) as the corresponding instance of 1-IN-3 GO. (See Fig. 4(a) for $\varphi = (x \vee y \vee \overline{z}) \wedge (x \vee z \vee w)$ as an example of the construction.)

First, for each variable x in φ, we add a cycle to G_φ. (See the upper half of G_φ in Fig. 4(a).) The cycle consists of some pairs of literal nodes labeled x and $\overline{x}$, and the number of such pairs in the cycle equals to the number of occurrences of the variable in φ. In addition, the labels x and $\overline{x}$ of the literal nodes appear alternatively when we traverse the cycle in the clockwise direction. Each pair $(x, \overline{x})$ will correspond to one occurrence of the literals of the variable x in φ. As an embedding of the cycle on a grid, we place a pair $(x, \overline{x})$ on the upper side if it corresponds to an occurrence of the positive literal x so that the left literal node is labeled with x; otherwise we place the pair on the lower side so that the left literal node is labeled with $\overline{x}$.

Second, for each clause $C_j = \ell_1 \vee \ell_2 \vee \ell_3$ with three literals ℓ_1, ℓ_2 and ℓ_3 in φ, we add two nodes to G_φ; one is a clause node labeled with C_j and the other is a negated-clause node labeled with $\overline{C_j}$. (See the bottom of G_φ in Fig. 4(a).) In the embedding, we place the clause node C_j on the left side

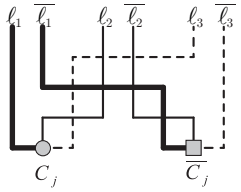

Fig. 5 A pair of a clause node C_j and a negated-clause node $\overline{C_j}$ which corresponds to a clause $C_j = \ell_1 \vee \ell_2 \vee \ell_3$ with three literals ℓ_1, ℓ_2 and ℓ_3 in φ.

and the negated-clause node $\overline{C_j}$ on the right side. Recall that each of the three literals ℓ_1, ℓ_2 and ℓ_3 has its corresponding pair of literal nodes in the upper half of G_φ. Then, as illustrated in Fig. 5, we can connect the clause node C_j with three literal nodes labeled with ℓ_1, ℓ_2 and ℓ_3; and connect the negated-clause node $\overline{C_j}$ with three literal nodes labeled with $\overline{\ell_1}$, $\overline{\ell_2}$ and $\overline{\ell_3}$.

This completes the construction of the corresponding instance G_φ of 1-IN-3 GO. By the construction, G_φ is 3-regular and contains $8m$ nodes and $12m$ edges in total. Thus, G_φ can be embedded on the grid of width $O(m)$ and height $O(n)$. Therefore, we can obtain G_φ together with its embedding in $O(mn)$ time.

2.1.2 Correctness of the reduction

We now prove that the reduction in Section 2.1.1 is correct, and indeed satisfies the one-to-one correspondence between valid orientations of G_φ and satisfying truth assignments of φ. As an example, a valid orientation in Fig. 4(b) of the graph G_φ corresponds to exactly one satisfying truth assignment $(x, y, z, w) =$ (False, False, False, True), and vice versa.

Consider a satisfying truth assignment of φ. Then, we assign the outward direction to all edges incident to the literal nodes which corresponds to True, while assign the inward direction to all edges incident to the literal nodes which corresponds to False; thus, Condition (1) of Definition 2 is satisfied. Note that, since the literal nodes x and $\overline{x}$ appear alternatively in the cycle for a variable x, these assignments do not conflict with each other. Since each edge in G_φ is incident to at least one literal node, we have already assigned directions to all edges in G_φ; this implies that a satisfying truth assignment of φ corresponds to exactly one orientation of G_φ. Since each clause in φ has exactly one true literal by this truth assignment, Conditions (2) and (3) of Definition 2 are satisfied and hence this orientation of G_φ is valid.

Conversely, consider a valid orientation of G_φ. Then, each literal node has in-degree either zero or three. Since the literal nodes x and $\overline{x}$ appear alternatively in the cycle for each variable x, all literal nodes with the same label x (or $\overline{x}$) have the same in-degree. Thus, we set $x =$ True if the literal nodes with label x have in-degree zero; otherwise set $x =$ False. Since Condition (2) of Definition 2 ensures that each clause node C_j has in-degree one, the corresponding clause C_j in φ has exactly one true literal. Thus, the obtained truth

assignment is satisfiable for φ. Furthermore, this truth assignment is uniquely decided from the orientation of G_φ.

This completes the proof of Theorem 1.

2.2 Reduction to L-TILING

In this subsection, we consider the tiling problem with only L-trominoes and prove the following theorem.

Theorem 2 L-TILING *and its counting and another-solution-problem variants are NP-complete, #P-complete and ASP-complete, respectively.*

Observe that L-TILING is in NP, in #P and in ASP. Therefore, as a proof of Theorem 2, we construct a polynomial-time reduction from 1-IN-3 GO to L-TILING with preserving the one-to-one correspondence between valid solutions.

2.2.1 Idea of the reduction

Since the problem is restricted to the case where $S = L$ (see Fig. 3(a)), we can construct only a polygon P so that it corresponds to a given instance of 1-IN-3 GO. Recall that, in the previous subsection, we have constructed the graph G_φ of 1-IN-3 GO with an embedding on a grid of polynomial size. Thus, we indeed construct the polygon P by replacing G_φ on a grid embedding with our gadgets so that a valid way of tiling the polygon P with L-trominoes corresponds to a valid orientation of G_φ, and vice versa.

More specifically, we explain our idea of the reduction using our simplest gadget, called a *line gadget* for L-TILING, which is illustrated in Fig. 6(a); this gadget corresponds to (a part of) an edge in G_φ. The line gadget for L-TILING consists of both the solid and dotted unit squares in Fig. 6(a), although each dotted unit square will be used as a "connector" and will be shared by another gadget to propagate the way of tiling. If we want to cover all solid unit squares of this gadget by L-trominoes, there are only two ways of tilings as illustrated in Fig. 6(b) and (c). We regard the ways of tilings (b) and (c) as two corresponding directions of (the part of) the edge in G_φ: The tiling (b) corresponds to assigning the direction from left to right to the edge, while the tiling (c) corresponds to assigning the direction from right to left to the edge. Covering (resp., not covering) the dotted unit square by an L-tromino represents that the direction of the edge is outbound (resp., inbound.)

If we need to bend an edge in G_φ, we use a *corner gadget* for L-TILING illustrated in Fig. 7(a). Similar to the line gadget above, if we want to cover all solid unit squares of the corner gadget by L-trominoes, there are only two ways of tilings; the dotted unit square covered (resp., not covered) by an L-tromino represents that the direction of the corresponding part of the edge is outbound (resp., inbound.) Therefore, these dotted unit squares work as connectors; by combining link and corner gadgets as in Fig. 7(b), we can correctly propagate the direction of the edge.

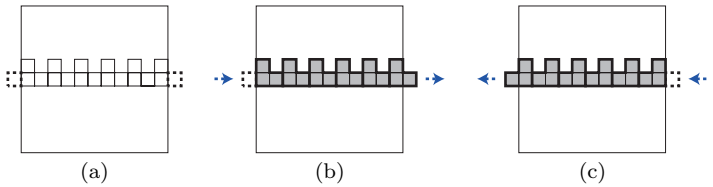

Fig. 6 A line gadget for L-TILING and LI-TILING, and its two ways of tiling.

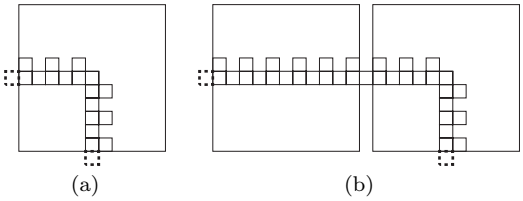

Fig. 7 (a) A corner gadget for L-TILING and LI-TILING, and (b) a combination of line and corner gadgets.

We respectively replace the edge-crossing points, pairs of literal nodes, clause nodes, and negated-clause nodes in G_φ by cross, duplicator, clause, negated-clause gadgets for L-TILING that will be defined in Section 2.2.2. All but the duplicator gadget are of the same size. The size of the duplicator gadget equals to that of a combination of two other gadgets, since we replace two literal nodes by one duplicator gadget. As a result of these replacements, we can obtain a "patchwork" of the gadgets as the polygon P of L-TILING which corresponds to a given instance G_φ of 1-IN-3 GO.

Recall that the graph G_φ is embedded on a grid of size $O(mn)$, where m is the number of clauses in φ and n is the number of variables in φ. Therefore, we use $O(mn)$ gadgets for replacing the elements in G_φ. Furthermore, each of our gadgets consists of a constant number of unit squares, and hence the corresponding polygon P is of size $O(mn)$ and can be constructed in polynomial time.

Therefore, to complete the proof of Theorem 2, it suffices to give the other gadgets (i.e., cross, duplicator, clause, and negated-clause gadgets) with noting that they preserve the one-to-one correspondence between valid tilings and valid orientations.

2.2.2 Other gadgets for L-TILING

For each edge-crossing in the embedding of G_φ, we replace it with a *cross gadget* for L-TILING given in Fig. 8(a). There are only four ways, as illustrated in Fig. 8(b)–(e), to cover all solid unit squares of this gadget by L-trominoes. Note that the left and right connectors (dotted unit squares) always preserve the direction of the horizontal edge in G_φ, that is, if the left connector is covered, then the right connector is not covered, and vice versa. The same

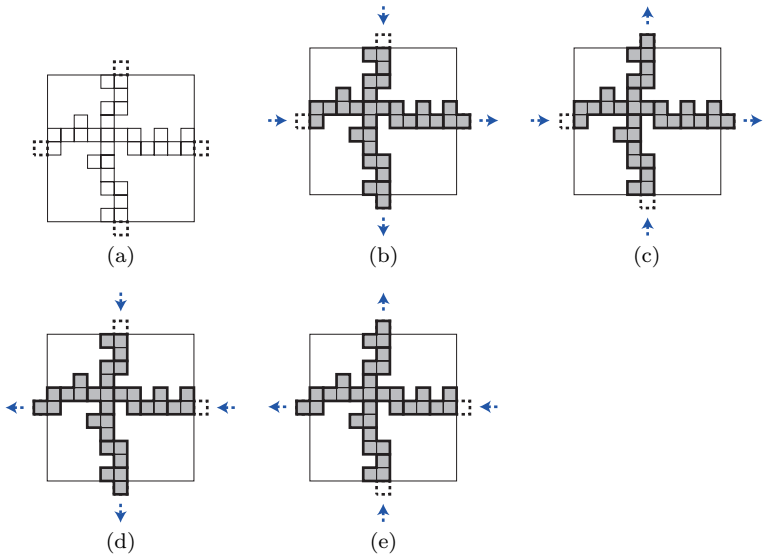

Fig. 8 A cross gadget for L-TILING and its four ways of tilings.

property holds for the upper and bottom connectors, and hence the direction of the vertical edge in G_φ is also preserved. Note that, however, we can independently choose the directions of the vertical and horizontal edges.

Each pair of literal nodes in G_φ is replaced with a *duplicator gadget* for L-TILING given in Fig. 9(a). As mentioned before, the width of the duplicator gadget equals to that of a combination of two other gadgets. There are only two ways to cover all solid unit squares of the duplicator gadget by L-trominoes, as shown in Fig. 9(b) and (c). The upper-left and bottom-left (resp., the upper-right and bottom-right) connectors always have the same direction (inbound or outbound), while the upper-left and upper-right connectors always take the opposite directions. This property preserves Condition (1) of Definition 2.

Each clause node in G_φ is replaced with a *clause gadget* for L-TILING given in Fig. 10(a). There are only three ways to cover all solid unit squares of this gadget, as shown in Fig. 10(b)–(d), each of which has exactly one uncovered connector (i.e., the inbound direction.) This property preserves Condition (2) of Definition 2.

Each negated-clause node in G_φ is replaced with a *negated-clause gadget* for L-TILING given in Fig. 11. Similar to the clause gadget above, there are only three ways to cover all solid unit squares of this gadget; however, each of them has exactly one covered connector (i.e., the outbound direction.) This property preserves Condition (3) of Definition 2.

This completes the proof of Theorem 2.

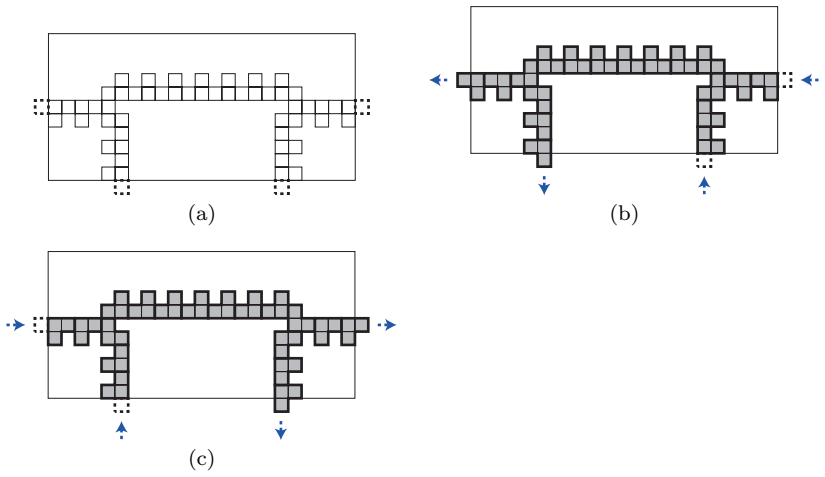

Fig. 9 A duplicator gadget for L-TILING and LI-TILING, and its two ways of tilings.

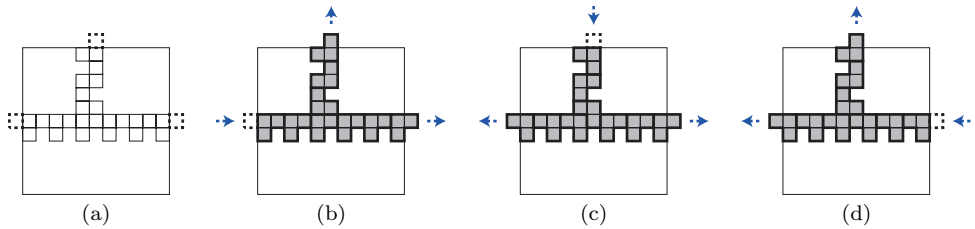

Fig. 10 A clause gadget for L-TILING and LI-TILING, and its three ways of tilings.

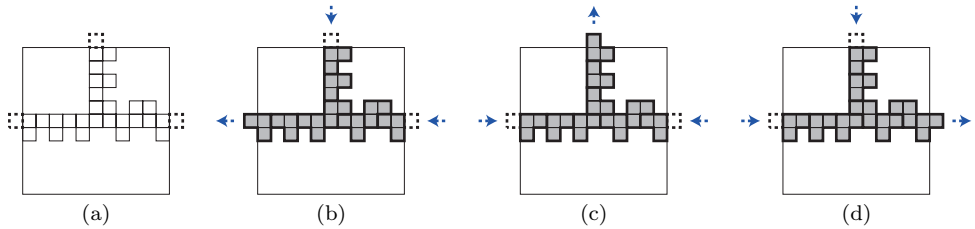

Fig. 11 A negated-clause gadget for L-TILING and LI-TILING, and its three ways of tilings.

2.3 Reductions to I-TILING, LI-TILING and 2BAR-TILING

In this subsection, we consider the tiling problem with only I-trominoes, both L-trominoes and I-trominoes, or only two bars. By the same arguments as in Section 2.2.1, it suffices to construct six gadgets for each variant, that is, line, corner, cross, duplicator, clause and negated-clause gadgets. Thus, we only give those gadgets for each variant such that

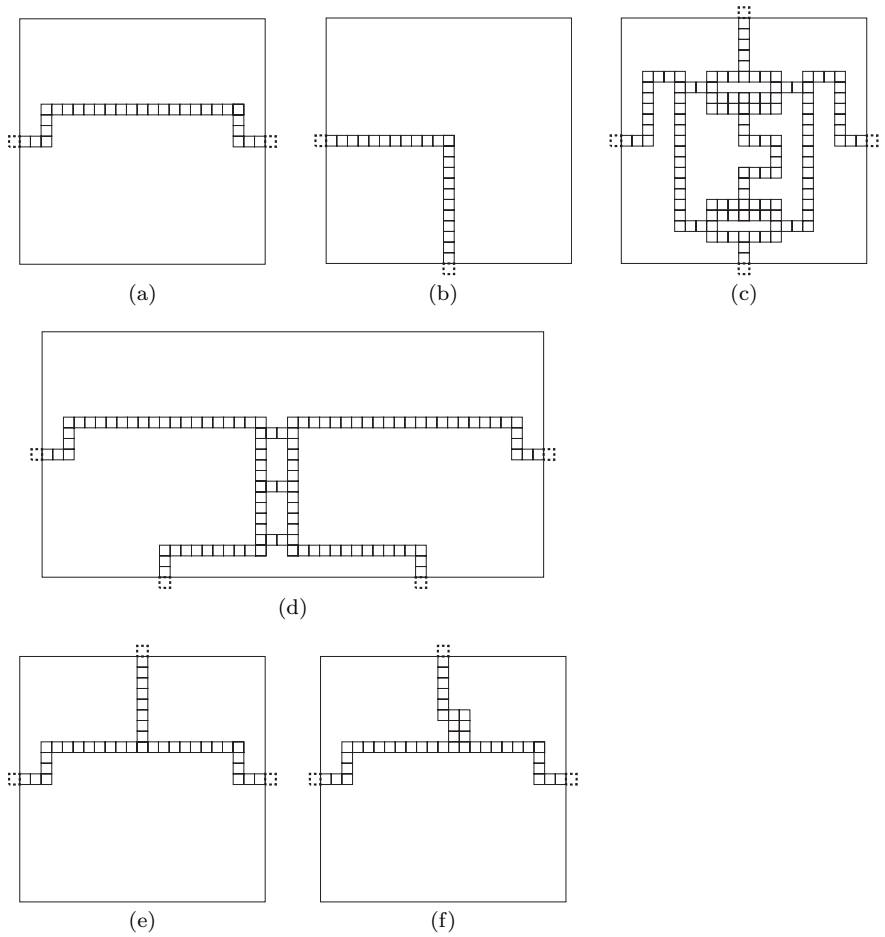

Fig. 12 Gadgets for I-TILING: (a) line, (b) corner, (c) cross, (d) duplicator, (e) clause and (f) negated-clause gadgets.

(a) the size of each gadget equals to that of a combination of a constant number of the smallest-size gadget in the variant;
(b) covering a connector (i.e., a dotted unite square) represents the outward direction, and not covering it represents the inward direction; and
(c) the gadgets preserve the one-to-one correspondence between valid orientations and valid tilings.

In the following, we give such gadgets for each variant, and prove the following theorem.

Theorem 3 I-TILING, LI-TILING *and* 2BAR-TILING *and their counting and another-solution-problem variants are NP-complete, #P-complete and ASP-complete, respectively.*

2.3.1 Gadgets for I-TILING

The line, corner, cross, duplicator, clause and negated-clause gadgets for I-TILING are given in Fig. 12(a)–(f), respectively. For line, corner and clause gadgets in Fig. 12(a), (b) and (e), respectively, we can straightforwardly find the ways to cover all solid unit squares, and can check their correctness. We note that a crank in the line gadget in Fig. 12(a) guarantees that there are exactly two ways of tilings, which correspond to two directions of (a part of) a corresponding edge in G_φ. The other non-trivial gadgets work as follows.

*(c) Cross gadget.*In general, constructing a gadget for an edge-crossing is the most difficult part to design in these kinds of reductions, as mentioned in [6]. This difficulty also occurs in I-TILING, too. (See Fig. 12(c) and 13.)

Observe that there are only two ways to cover each of the left and right connectors, because of the crank structure; recall the similar structure in the line gadget in Fig. 12(a). Since there are 127 solid unit squares in the gadget, we need 43 I-trominoes to cover all of them and hence two unit squares of I-trominoes remain. These two remaining unit squares can cover two of the four connectors. Carefully tracing all possible ways of tilings, we can check that both of the top and bottom connectors cannot be covered at the same time, and both of the right and left connectors cannot be covered at the same time. Therefore, we only have four ways of tilings as shown in Fig. 13.

*(d) Duplicator gadget.*In the middle part of the duplicator gadget in Fig. 12(d), notice that there are three "bridges" of length four which join the right and left halves. Then, there are only two ways to cover each bridge; putting an I-tromino on either the left side or the right side. Since there are 106 solid unit squares in the gadget, we need 36 I-trominoes to cover all of them and hence two unit squares of I-trominoes remain. These two remaining unit squares can cover two of the four connectors. The crank structure near each connector controls the possible ways of tilings, and indeed there are only two ways to cover all solid unit squares of the gadget as shown in Fig. 14.

*(f) Negated-clause gadget.*In the negated-clause gadget given in Fig. 12(f), the rectangle of size 4×2 placed on the center plays an important role. Consider the ways to put two I-trominoes on the rectangle. Note that, since the width of the rectangle is two, both the I-trominoes must be of size 3×1. If the two I-trominoes are put on different heights, then we cannot cover all solid unit squares of the rectangle. Therefore, there are only two ways to put the I-trominoes on the rectangle. This gives us only three valid tilings of the gadget, as illustrated in Fig. 15.

2.3.2 Gadgets for LI-TILING

Notice that, if we use even one I-tromino, there is no way to cover all solid unit squares of the gadgets for L-TILING except for the cross gadget. Therefore, we

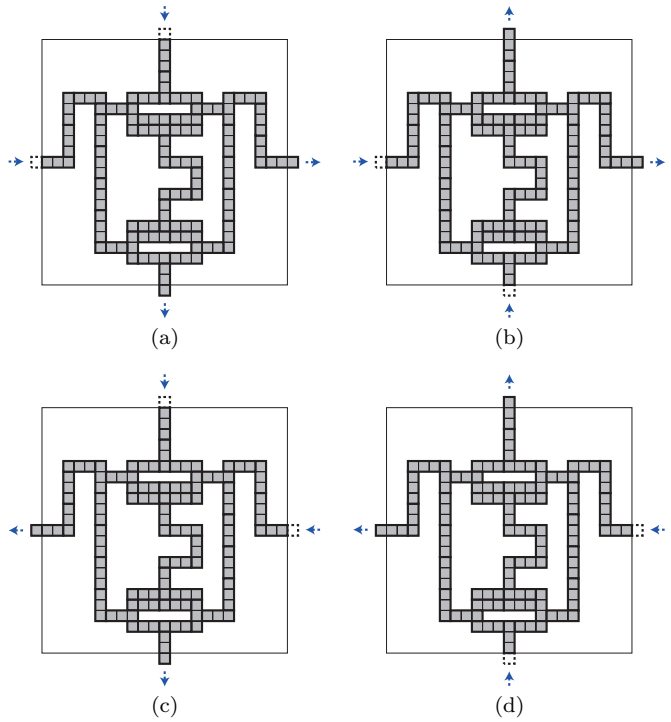

Fig. 13 Four ways to cover the cross gadget for I-TILING.

can use the gadgets for L-TILING in Fig. 6, 7, 9, 10 and 11 as line, corner, duplicator, clause and negated-clause gadgets for LI-TILING, respectively. The cross gadget for LI-TILING is given in Fig. 16. There are only four ways to cover all solid unit squares of the gadget, as shown in Fig. 17.

2.3.3 Gadgets for 2BAR-TILING

Notice that I-TILING is a special case of 2BAR-TILING. Indeed, all gadgets for I-TILING can be extended straightforwardly to those for 2BAR-TILING with $S = \{1 \times m,\ k \times 1\}$, for any fixed integers $m \geq 2$ and $k \geq 3$. As examples, we give the gadgets for $S = \{1 \times 2,\ 3 \times 1\}$ in Fig. 18, and give the cross and duplicator gadgets for $S = \{1 \times 5,\ 4 \times 1\}$ in Fig. 19 and 20, respectively.

Recall that, in the negated-clause gadget for I-TILING in Fig. 12(f) and 15, the rectangle of size 4×2 at the center played an important role. For 2BAR-TILING with $S = \{1 \times m,\ k \times 1\}$, we define the size of such a rectangle as $(k+1) \times (m-1)$. Then, to cover all unit squares in the rectangle, we have to place $(m-1)$ bars of size $k \times 1$ with the same height.

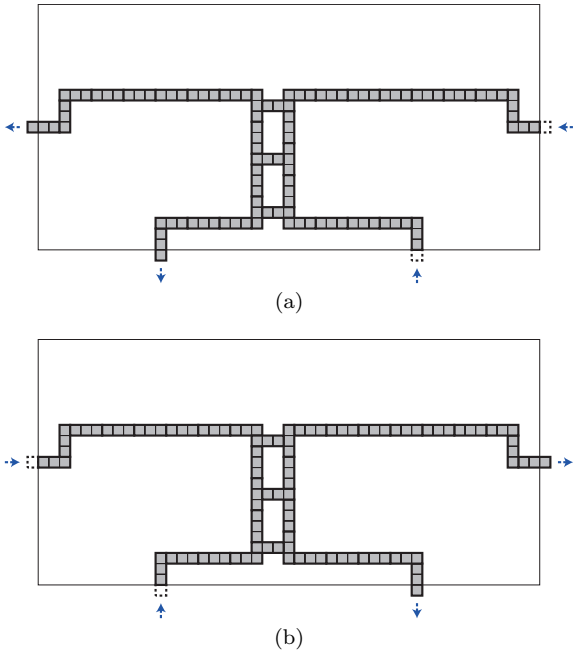

Fig. 14 Two ways to cover the duplicator gadget for I-TILING.

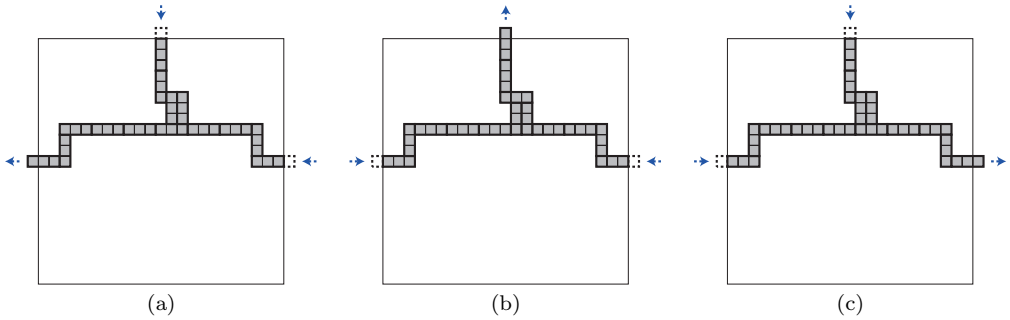

Fig. 15 Three ways to cover the negated-clause gadget for I-TILING.

3 Concluding Remarks

We have studied the computational hardness of L-TILING, I-TILING, LI-TILING and 2BAR-TILING, and clarified that the tiling problem remains computationally intractable even for a few (at most six) polyominoes. We emphasize that our results gave answers to two open questions proposed by [9,10].

In this paper, we focused on the tiling problem for a polygon P with holes. It remains open to clarify whether the tiling problem and its variants remain

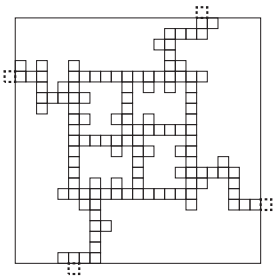

Fig. 16 A cross gadget for LI-TILING.

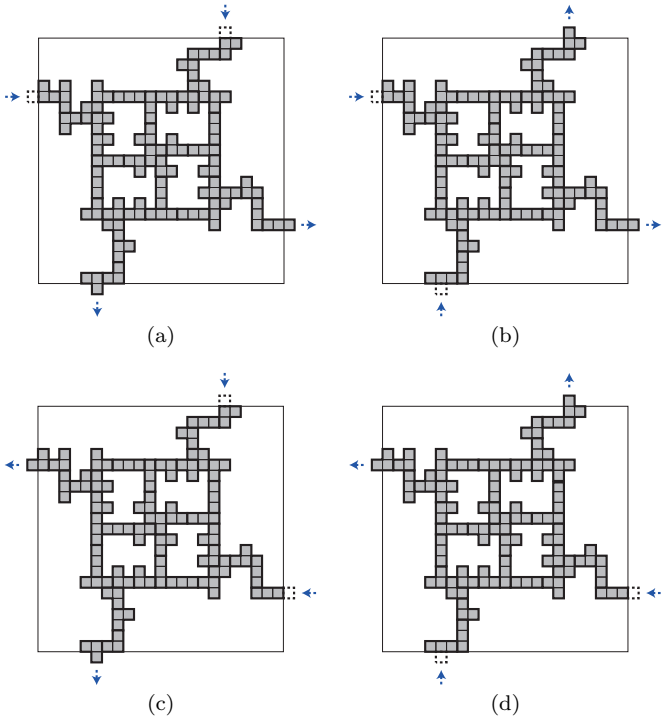

Fig. 17 Four ways to cover the cross gadget for LI-TILING.

NP-complete, #P-complete and ASP-complete even if a given polygon P has no hole and a given set S consists of only a few polyominoes.

Finally, we note that many of our reductions can be extended to the following related problems.

(1) Packing problem
Given a polygon P, a set S of polyominoes and an integer k, determine whether k copies of polyominoes in S can be placed on P without any overlap. Note that, in the packing problem, we can rotate the copy of a

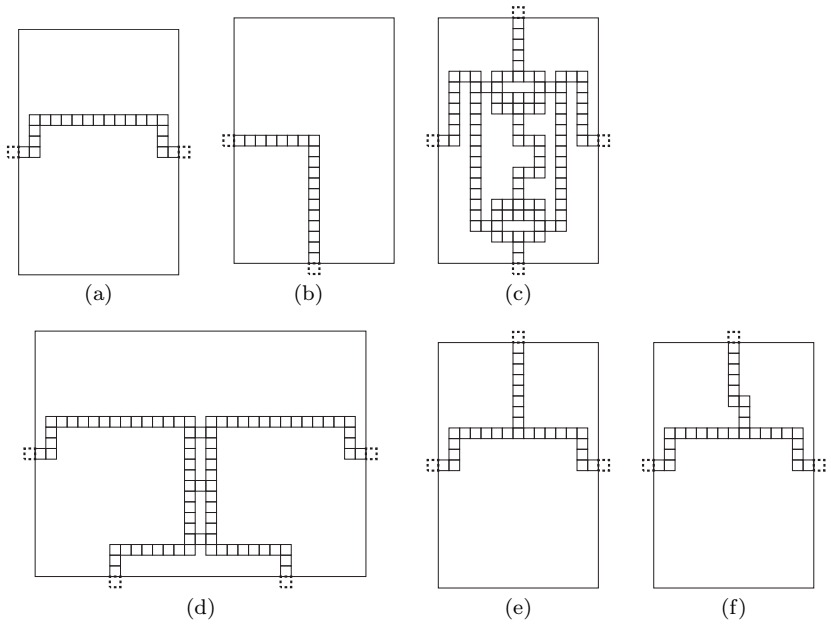

Fig. 18 Gadgets for 2BAR-TILING with $S = \{1 \times 2,\ 3 \times 1\}$: (a) line, (b) corner, (c) cross, (d) duplicator, (e) clause and (f) negated-clause gadgets.

polyomino in S. It is known that this problem is solvable in polynomial time if $S = \{1 \times 2\}$, while it is NP-complete if $S = \{2 \times 2\}$ [3,4].

(2) Covering problem
Given a set P of points in the plane, a set S of polyominoes and an integer k, determine whether k translated copies of polyominoes in S can cover all points in P. In this problem, the copies of polyominoes are allowed to mutually overlap each other.

(3) Unique covering problem
Given a set P of points in the plane, a set S of polyominoes and an integer k, determine whether k translated copies of polyominoes in S can uniquely cover all points in P, that is, no overlap of the copies of polyominoes is allowed.

Similar to L-TILING, I-TILING, LI-TILING and 2BAR-TILING, consider the three problems above when restricted to only L-trominoes, only I-trominoes, both L-trominoes and I-trominoes, and only two bars.

Notice that each of L-TILING, I-TILING and LI-TILING takes all possible rotated copies of trominoes as the set S of polyominoes; the same holds for 2BAR-TILING with $S = \{1 \times k,\ k \times 1\}$ for any fixed integer $k \geq 3$. Therefore, we can see the correspondence between the tiling and the packing problems for such cases, and hence the packing problem and its variants are NP-complete, #P-complete and ASP-complete for such cases.

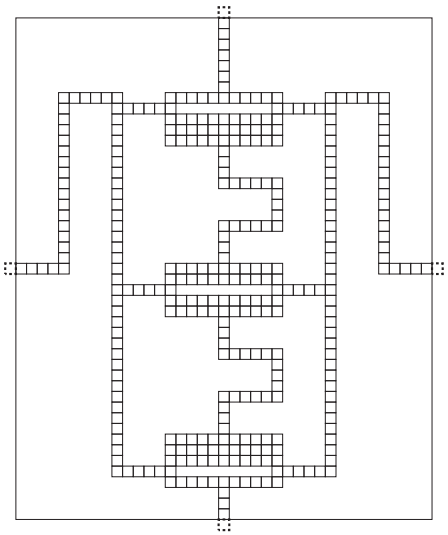

Fig. 19 A cross gadget for 2BAR-TILING with $S = \{1 \times 5,\ 4 \times 1\}$.

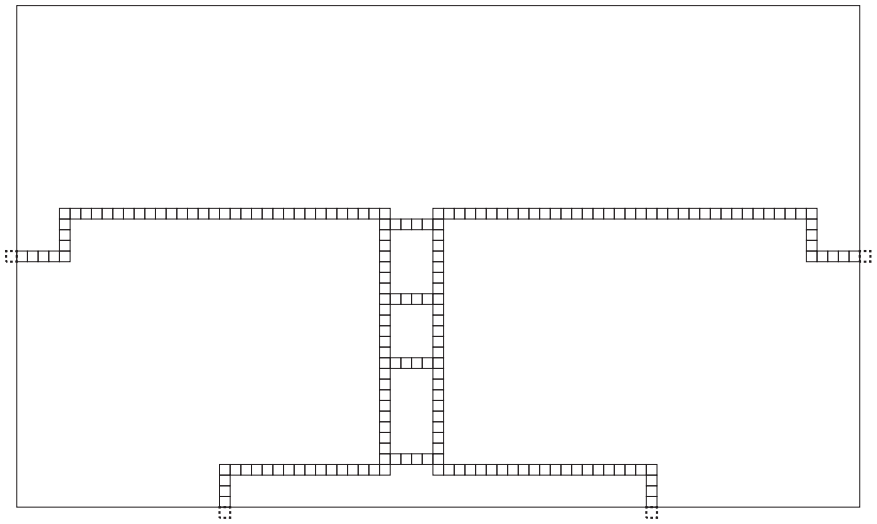

Fig. 20 A duplicator gadget for 2BAR-TILING with $S = \{1 \times 5,\ 4 \times 1\}$.

By converting the gadgets in this paper to the positions of points (and regarding the set of points as the set P), we can also see the correspondence between the tiling and the (unique) covering problems. Therefore, the (unique) covering problem and its variants are NP-complete, #P-complete and ASP-complete.

Acknowledgements We are grateful to Yoshio Okamoto for his fruitful suggestions, and thank anonymous referees of the preliminary version and of this journal version for their helpful suggestions. This work is partially supported by MEXT/JSPS KAKENHI Grant Numbers JP15K00008 and JP24106007 (T. Horiyama), JP15H00849 and JP16K00004 (T. Ito), JP26730001 (A. Suzuki), and JP26330009 and JP24106004 (R. Uehara).

References

1. D. Beauquier, M. Nivat, E. Rémila and M. Robson. Tiling figures of the plane with two bars. *Computational Geometry* 5, pp. 1–25 (1995)
2. N. Creignou and M. Hermann. Complexity of generalized satisfiability counting problems. *Information and Computation* 125, pp. 1–12 (1996)
3. D. El-Khechen, M. Dulieu, J. Iacono and N. van Omme. Packing 2×2 unit squares into grid polygons is NP-complete. *Proc. of the 21st Canadian Conference on Computational Geometry*, pp. 33–36 (2009)
4. R.J. Fowler, M. Paterson and S.L. Tanimoto. Optimal packing and covering in the plane are NP-complete. *Information Processing Letters* 12, pp. 133–137 (1981)
5. S. Golomb. *Polyominoes* (2nd edition). Princeton University Press (1994)
6. R.A. Hearn and E.D. Demaine. *Games, Puzzles, and Computation.* A K Peters Ltd. (2009)
7. T. Horiyama, T. Ito, K. Nakatsuka, A. Suzuki and R. Uehara. Packing trominoes is NP-complete, #P-complete and ASP-complete. *Proc. of the 24th Canadian Conference on Computational Geometry*, pp. 219–224 (2012)
8. C. Kenyon and R.W. Kenyon. Tiling a polygon with rectangles. *Proc. of the 33rd Annual Symposium on Foundations of Computer Science*, pp. 610–619 (1992)
9. C. Moore and J.M. Robson. Hard tiling problems with simple tiles. *Discrete & Computational Geometry* 26, pp. 573–590 (2001)
10. I. Pak and J. Yang. Tiling simply connected regions with rectangles. *Journal of Combinatorial Theory*, Series A 120, pp. 1804–1816 (2013)
11. E. Rémila. Tiling a simply connected figure with bars of length 2 or 3. *Discrete Mathematics* 160, pp. 189–198 (1996)
12. E. Rémila. Tiling with bars and satisfaction of Boolean formulas. *European Journal of Combinatorics* 17, pp. 485–491 (1996)
13. E. Rémila. Tiling a polygon with two kinds of rectangles. *Discrete & Computational Geometry* 34, pp. 313–330 (2005)
14. T.J. Schaefer. The complexity of satisfiability problems. *Proc. of the 10th Annual ACM Symposium on Theory of Computing*, pp. 216–226 (1978)
15. T. Yato and T. Seta. Complexity and completeness of finding another solution and its application to puzzles. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences* E86-A, pp. 1052–1060 (2003)