Puzzles are hard enough just by rotations

Takeshi Yamada* Tom van der Zanden† Ryuhei Uehara*

Abstract

In this paper, we investigate the computational complexity of a series of puzzles. We are given a set of n centers of circles and n unit disks. Each disk is based on a unit circular shape, but a part of it can be bitten by other disks, which is called a lune. We investigate three variants of this puzzle. First, we investigate the classic packing puzzle of lunes. We note that the centers of disks are given as a part of input, and some disks can be lunes. Therefore, essentially, we only can rotate the disks and lunes to pack them. Even under this strong constraint, the packing problem is still NP-complete. Next we turn to the combinatorial reconfiguration variant of this puzzle. That is, we are given two nonoverlapping arrangements of disks and lunes on a given set of centers on a board. Each disk is pinned at the center, and thus we can just rotate it. The problem asks if we can transform one to the other by just rotations of disks without overlapping. We show that this puzzle is PSPACE-complete. Lastly, we focus on the cases in which the puzzle can be solved in polynomial time. The first tractable case is one-dimensional packing puzzle. The second one is the screw-type variant of this puzzle. In this variant, each disk or lune is realized by a thick screw. We are given a nonoverlapping arrangement of them. The operation we can do is that we can screw a disk if its orbit is not blocked by any other screw. We prove that this variant can be solved in polynomial time.

1 Introduction

In the context of research on computational complexity and algorithms, the study of puzzles has played an important role. One of the reasons is that an algorithm is a methodology for how to combine basic operations, while a puzzle is an abstract model of how to combine the basic pieces under some constraints of pieces. For example, in [1], Asano et al. investigate the computational complexity of a puzzle that asks for finding proper order of a deck of cards with/without rotation and/or flipping. On the other hand, in [4], Hearn and Demaine propose a general framework that can capture the computational complexities of many puzzles, including sliding block puzzles. This framework is useful to show the hardness of many problems in combinatorial reconfiguration, especially, token sliding problem on a graph. Recently, in [3], Kanzaki et al. investigate the jumping block puzzles, which is a puzzle counterpart of token jumping problem on a graph investigated in the context of the combinatorial reconfiguration.

In this paper, we investigate representative puzzles that impose strong constraints on both of movements and pieces. This is a framework that has long been popular in the puzzle society, and many commercially available products exist (Figure 1). These puzzles have the following common property: (1) each piece is based on a unit disk of radius 1, (2) some pieces can be missing parts, which is called *lunes*, (3) the centers of disks are essentially fixed (or easy to specify). Let's call these puzzles *rotation puzzles*. That is, the input of the puzzle is a set of n unit disks or lunes and a set of n centers. We call each piece *lune* since they can be missing parts from a unit disk. We assume that the center can be easily determined from the lune. Precisely, when a lune is inscribed in a unit circle, the center point of the lune should be uniquely determined.

We investigate three variants of the rotation puzzles. The first one is the *packing puzzle* that asks us to determine (or to find) if all the lunes can be put on a board so that the centers of the disks are matching to the given centers, and no pair of lunes overlaps (Figure 1(a)). We call such a solution *feasible solution*. That is, the packing puzzle asks us to determine if a given set of n centers and n lunes have a feasible solution. We show that this problem is NP-complete (in Section 2). The next puzzle is the *reconfiguration puzzle* that asks us, for two feasible solutions S and T of the lunes with fixed centers, to determine if S can be transformed to T by a sequence of rotations of lunes. That is, we cannot pick up any piece, and the only operation we can do is rotate a lune at once. We prove that this quite restricted puzzle is PSPACE-complete in general (in Section 3).

We turn to some tractable cases in Section 4. We first show that one-dimensional packing puzzle is linear-time solvable. That is, when the centers are on a line, it can be solved efficiently. This case is inspired by one of classic puzzles shown in Figure 1(c), which requires exponential operations to take out. Another tractable case is the *screw puzzle* (Figure 1(d)). This puzzle is relatively new one comparing to the other classic puz-

*School of Information Science, Japan Advanced Institute of Science and Technology, {tyama,uehara}@jaist.ac.jp

†Department of Data Analytics and Digitalization, Maastricht University, t.vanderzanden@maastrichtuniversity.nl

(a) Circle Crazy, Balloon Boogler[1], Euro-Crisis[2]

(b) Whirligig Plate

(c) Spin Out

(d) Honeycomb[3]

Figure 1: Lune puzzles based on (a) packing and (b) rotation. (c) A classic puzzle "Spin Out" and (d) a lune puzzle based on screwing.

zles. Each piece forms a screw, and we can take it if we can rotate it. In the real puzzle shown in Figure 1(d), it is not difficult to find the first two pieces screwed out, however, the third one to be taken out is not so easy. (We will discuss this point in Section 4.) We will show that this puzzle can be solved in polynomial time.

In this paper, all lunes $\ell_1, \ldots, \ell_n$ are of radius 1. That is, every lune is inscribed in a unit circle, and it cannot be inscribed in a circle of radius $r < 1$. Each of n centers c_i is given by its coordinates (x_i, y_i). (We assume a RAM model as a computational model and each number takes a real value.) A lune is obtained from a unit disk by bitten out some constant number of overlaps with other disks. Therefore, a lune ℓ_i can be represented by the set of unit disks that bite it off.

[1] https://puzzlemist.com/product/balloon-boggler-puzzle/

[2] https://www.puzzlemaster.ca/browse/inventors/jurgenreiche/16681-euro-crisis-deluxe

[3] https://www.puzzlemaster.ca/browse/metalpuzzles/metalpuzzlemaste/18111-honeycomb-metal-puzzle

2 NP-completeness of packing puzzle

In Figure 1(a), we show some commercial products. We can generalized this puzzle as follows:

Packing puzzle of bitten disks

Input: Two sets of n centers c_i and n lunes ℓ_i for $i = 1, \ldots, n$.

Output: Decide if we can arrange n lunes onto n centers without overlapping of lunes.

The main theorem in this section is below:

Theorem 1 *The packing puzzle of lunes is NP-complete in general.*

Proof. (Outline.) It is easy to see that this problem is in NP since we can check if a given arrangement is feasible in polynomial time. Thus we show NP-hardness by reducing the well-known planar 1-in-3 SAT to the packing puzzle. (For NP-hardness of the planar 1-in-3 SAT problem, see, e.g., [5].) Let $F = C_1 \wedge \cdots \wedge C_m$ be an instance of Planar 1-in-3 SAT, where C_i is a clause of three literals in $X = \{x_1, x_2, \ldots, x_{n'}\}$ and $\bar{X} = \{\bar{x_1}, \bar{x_2}, \ldots, \bar{x_{n'}}\}$. We assume that the corresponding graph $G = (V, E)$ for F defined by (1) $V = X \cup \bar{X}$ and (2) $E = E_1 \cup E_2$ where $E_1 = \{\{x_i, \bar{x}_i\}\}$ for each i and $E_2 = \{\{l_i, C_j\}\}$ for each literal l_i in C_j is planar. We first prepare $n' + 1$ different real numbers $\sqrt{2} < r_1 < \cdots < r_{n'} < r_{n'+1} < 2$. We construct *clause gadget*, *wire gadget*, and *variable gadget* as follows.

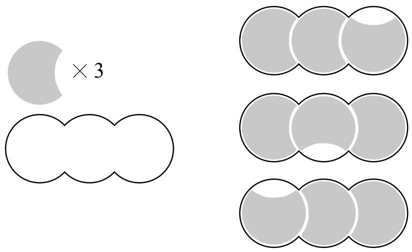

Figure 2: A clause gadget.

A clause gadget consists of three congruent lunes and three centers on a line as shown in the left of Figure reffig:clause. In the figure, we describe boundaries of gadget instead of centers for ease to see. The distance of each pair of consecutive two centers is $r_{n'+1}$, and each lune lacks one lens-shape corresponding to $r_{n'+1}$. (We say that this lens-shape has a *thickness* of $r_{n'+1}$.) When we pack three lunes onto the three centers, as shown in the right of Figure reffig:clause, exactly one of three lunes can have one lens-shape gap. We will use this gap to represent one true literal. We prepare m copies of clause gadgets with reasonable margins.

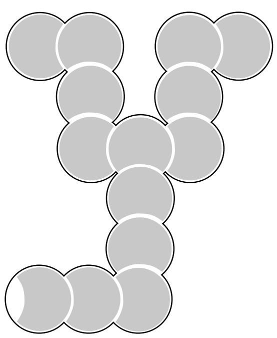

Figure 3: Wire gadgets with turns.

A wire gadget is the same as the clause gadget. It consists of sequence of centers of uniform distance $r_{n'+1}$ from one circle in a clause gadget. We prepare the same lunes that lack one lens-shape of thickness $r_{n'+1}$. The sequence of the wire gadgets extends the clause gadget in three directions in a natural way as shown in Figure 3. It is easy to make a turn.

The key gadget is the variable gadget for each variable x_i. A variable gadget for the variable x_i consists of $2a_i + 2$ centers spaced equally apart on a line with a distance r_i between them, where a_i is the number of appearance of x_i in F. It contains $a_i + 1$ unit disks, one

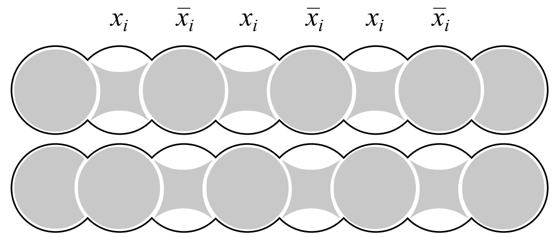

Figure 4: A variable gadget for $a_i = 3$ with two ways of packing.

lune lacking one lens-shape of thickness r_i, and a_i lunes that lack 4 lens-shapes of thickness r_i. It is not difficult to see that we only have four possible ways to pack those lunes and disks onto the centers and we only have two possible boundaries of them as shown in Figure 4. In Figure 4, bottom one describes x_i is true, and the upper one describes x_i is false. Each wire gadget should be connected to the circle of the variable gadget labeled by the corresponding literal x_i or $\bar{x}_i$.

Using the same trick between a clause gadget and a wire gadget, we can connect a variable gadget to a corresponding clause gadget in a natural way. The reduction can be done in a polynomial time. Thus we show that the packing puzzle has a solution if and only if F has an assignment that satisfy 1-in-3 SAT condition.

When F has a feasible assignment, each clause have one literal that is true. In the clause gadget, we make a gap at the corresponding lune. It is transferred by the wire gadget, and the corresponding position of the variable gadget can be convex. On the other hand, if we have a convex shape at the clause gadget since the corresponding literal is false, it is transferred by the wire, and the corresponding position of the vertex gadget should be concave. Therefore, when F has a feasible assignment, we can assign concave lune at each position corresponding to false literal and convex disk at each position corresponding to true literal in the variable gadget. Thus we can pack all the lunes.

Next we assume that the constructed packing puzzle has a solution. We first focus on the variable gadget for x_1. The gadget allow to pack lunes of thickness r_1 since $r_1 < r_i$ for all $i > 1$. Thus we need all lunes of thickness r_1 to pack it. But it has a gab between lunes. If we use some lunes of thickness r_i with $i > 1$ to fill the gap in the gadget for x_1, it is easy observe that we cannot pack the gadget for x_i. Using this argument repeatingly, we can observe that we have to use all lunes of thickness r_i to fill the gadget for the variable x_i. Then, we have to use all lunes of thickness $r_{n'+1}$ to fill the wire gadgets and clause gadgets because we have no gap in these gadgets. Thus, eventually, we use all unit disks to pack the gadgets for x_i. Therefore, when the packing puzzle has a solution, the way of packing gives us the feasible assignment of F, which completes the proof. □

3 PSPACE-completeness of reconfiguration puzzle

In the puzzle shown in Figure 1(b), the lunes are all the same shape and they are pinned at the uniform square grid. This puzzle and its generalization has been investigated in [6] as the *cyclic shift puzzle*. In this paper, we introduce a natural variant of our rotation puzzles in the context of the combinatorial reconfiguration (see, e.g., [2]):

Reconfiguration puzzle of lunes

Input: Two sets of n centers c_i and n lunes ℓ_i for $i = 1, \ldots, n$, and two feasible arrangements S and T of the lunes on the centers.

Output: Decide if we can reconfigure from S and T by a sequence of rotations of lunes.

We note that all lunes are pinned at their centers, and we can just rotate them. We also assume that we rotate one lune at a time. This contrasts to the other reconfiguration problems that allow to shift (or slide), or even jump as discussed in [3].

The main theorem in this section is below:

Theorem 2 *The reconfiguration puzzle of lunes is PSPACE-complete in general.*

Proof. (Outline.) We reduce the nondeterministic constraint logic (NCL) proposed and discussed in [4]. As an NCL instance, we are given a directed graph $G_0 = (V_0, E_0)$ with an edge e in E_0. In each step, we can obtain E_{i+1} from E_i by flipping an edge. The NCL asks if a specific edge e can be flipped or not, and it is PSPACE-complete in general. Further more, in [4], it is still PSPACE-complete even if (1) G_0 is planar, (2) V_0 contains two types of nodes called "AND" and "OR" in [4, Theorem 5.12], and (3) each "OR" has only one "true" assignment [4, Theorem 5.13].

Therefore, essentially, it is sufficient to show that the "AND" node and the "OR" node in the NCL can be realized by our reconfiguration puzzle of lunes like the other puzzles discussed in [4, Chapter 9]. The wire gadget is already described in Figure 3. In the wire gadget, the wire send "a lens-shape gap" as a signal. Therefore, the AND gadget can be realized easily as shown in Figure 5. When it catches two lens-shape gaps at both sides of the central disk, the central lune can be rotated in 180°, and it can transmit the gap to down. We need some device for realizing the OR gadget as shown in Figure 6. In Figure 6, we cannot rotate the central lune since it is locked by two top neighbors. However, once one of the two neighbors is free by receiving a lens-shape gap as shown in Figure 7, it can be rotated until it strikes the other neighbor (in the figure, the central lune receives the lens-shape gap from left, and it rotated in counterclockwise, and left small gap comes to down). Then the down neighbor can be rotated freely, and it can provide a big lens-shape gap at left side to down.

Then, one of two small gaps at the central lune can be down, and its down neighbor can be rotated.

□

4 Polynomial-time solvable puzzles

4.1 One-dimensional packing puzzle

The "Spin Out" shown in Figure 1(c) is a classic puzzle similar to the other famous classic puzzle "Chinese Ring". They are uniformly constructed and both require exponential number of operations like the Hanoi Tower. Inspired by the Spin Out, we can consider the packing puzzle of lunes discussed in Section 2 in one dimension. That is, we consider the following packing puzzle in 1D:

Packing puzzle of lunes in 1D

Input: Two sets of n centers c_i and n lunes ℓ_i for $i = 1, \ldots, n$ such that (1) the centers are assumed to be equally spaced on a line with a distance $r < 1$ between them, and (2) each lune is (a) a unit disk, (b) it lacks one lens-shape of thickness r, or (c) it lacks two lens-shape of thickness r on opposite sides.

Output: Decide if we can arrange n lunes onto n centers without overlapping of lunes.

This case can be solved in linear time:

Theorem 3 *The packing puzzle of lunes in 1D is linear time solvable.*

Proof. (Outline.) Let L_a, L_b, L_c be the sets of lunes in the groups (a), (b), and (c), respectively. When $L_a \neq \emptyset$, we can assume that one element should be put on c_1 without loss of generality. Any pair of lunes in L_a cannot be adjacent. Therefore, the elements of L_a must be placed at least two apart. Between them, we have to use at least one element in L_c. We also observe that we can put each element in L_b in an arbitrary way between the elements in L_a and L_c. Thus we can pack all elements in $L_a \cup L_b \cup L_c$ if and only if $(|L_a| - 1) < |L_c|$, which can be checked in linear time. □

4.2 Screw puzzle

The Honeycomb Metal Puzzle is a new product released in 2023 (Figure 1(d)). We can consider this puzzle is a variant of a lune puzzle based on screwing, which we name as the *disassemble puzzle*. For a given arrangement of lunes, we can solve the disassemble puzzle in polynomial time when we restrict ourselves to just the rotation operation:

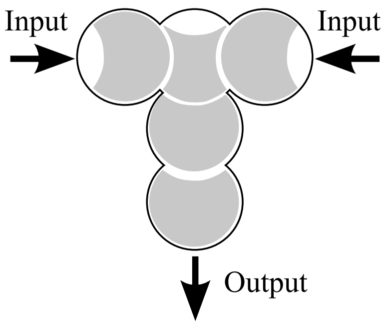

Figure 5: An AND gadget.

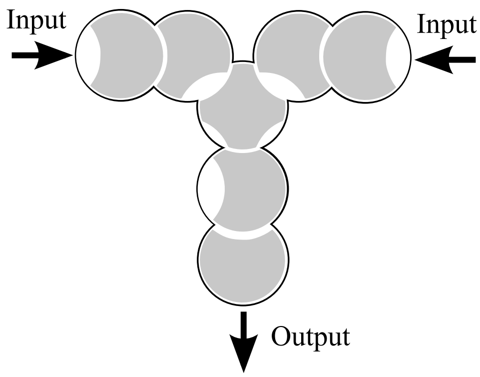

Figure 6: An OR gadget.

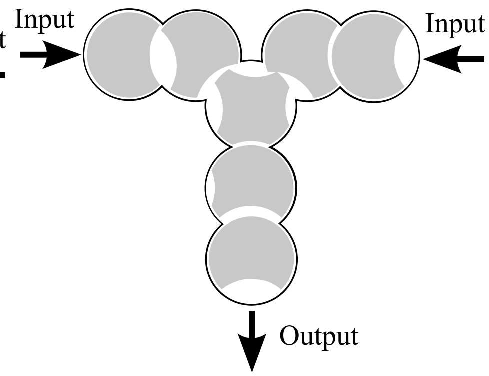

Figure 7: How an OR gadget transmits a signal from left to down.

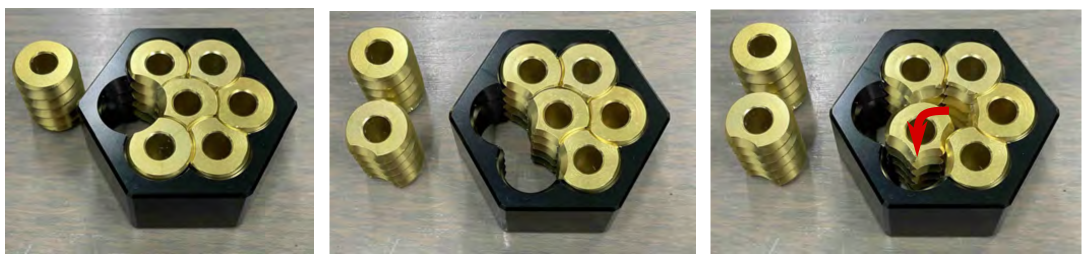

Figure 8: How to disassemble the Honeycomb Metal Puzzle.

Lemma 4 *The disassemble puzzle of lunes can be solved in polynomial time when the possible operations are restricted to rotations at the center of each lune only.*

Proof. For a given arrangement, we can construct a directed graph $G = (V, A)$ as follows. First, we let V be the set of lunes ℓ_i with $i = 1, 2, \ldots, n$. Then the set A contains a directed edge (ℓ_i, ℓ_j) if and only if ℓ_i is an obstacle that prevents ℓ_j from rotating. It is easy to observe that we can screw out ℓ_i from this arrangement when ℓ_i has indegree 0 on G. If we have such a lune, we screw it out and remove it from V. Repeating this process, we can disassemble, and we cannot otherwise. □

As observed in the introduction, in the arrangement in Figure 1(d), we can screw out the first two lunes. However, we cannot screw out the third one in place. That is, this disassemble puzzle has no solution if the possible operations are restricted to rotations at the centers of lunes only.

In fact, we have to rotate the third lune along a *different* center to take out it (Figure 8). It is not difficult to characterize when a lune can be moved locally in this way, as it depends on the conditions of its neighbors. (A similar analysis can be found in the similar puzzle based on hexagonal grid; see, e.g., [7].) For this commercial product, it is not difficult to find the way of disassembling. The more general case remains open.

References

[1] Tetsuo Asano, Erik D. Demaine, Martin L. Demaine, and Ryuhei Uehara. NP-completeness of generalized Kaboozle. *Journal of Information Processing*, Vol. 20, No. 3, pp. 713-718, July, 2012.

[2] Takehiro Ito, Erik D. Demaine, Nicholas J. A. Harvey, Christos H. Papadimitriou, Martha Sideri, Ryuhei Uehara, and Yushi Uno. On the Complexity of Reconfiguration Problems. *Theoretical Computer Science*, Vol. 412, pp. 1054-1065, 2010.

[3] Masaaki Kanzaki, Yota Otachi, Giovanni Viglietta, and Ryuhei Uehara. Computational Complexity of Jumping Block Puzzles, *Theoretical Computer Science*, Vol. 983, 114292(19 pages), February 2024.

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

[5] Alexander Pilz. Planar 3-SAT with a Clause/Variable Cycle, *Discret. Math. Theor. Comput. Sci.*, Vol. 21, No. 3, 20 pages, 2019.

[6] Kwon Kham Sai, Ryuhei Uehara, and Giovanni Viglietta. Cyclic Shift Problems on Graphs. *IEICE Trans.*, Vol. E105-D, No. 3, pp. 532-540, March 2022.

[7] Yanqiu Tan. Research on the Computational Complexity of the Crush Ice Game. *Master thesis*, Japan Advanced Institute of Science and Technology, March 2025.