The Computational Complexity of Creek Puzzles on Several Grids

Akihiro Uejima1,a) Kentaro Oe1

Received: December 29, 2019, Accepted: September 10, 2020

Abstract: The Creek puzzle is a pencil-and-paper puzzle developed by Japanese puzzle publisher Nikoli, usually played on a square grid. We study the computational complexity of the Creek puzzle, and first it is shown that deciding whether a given instance of this puzzle on a square grid has a solution is \( \text{NP}\)-complete by a reduction from the Circuit-SAT problem. In addition, we prove the \( \text{NP}\)-completeness for the Creek puzzle on triangular grids in the same way.

Keywords: Creek puzzle, pencil-and-paper puzzle, computational complexity, \( \text{NP}\)-complete, \( \text{ASP}\)-complete

1. Introduction and Definitions

A wide variety of puzzles is played all over the world. Pencil-and-paper puzzles, which consist of figures or words on paper, are solved by a person drawing or coloring on the paper with a pencil. Such puzzles are a popular form of recreation. The Creek puzzle is one of many pencil-and-paper puzzles made popular by Nikoli [15], and it is played on a square grid (see Fig. 1). Circles are placed at some lattice points on the grid and these circles are numbered, specifically any of 0, 1, 2, 3, 4. The player’s task is to color some squares blue according to the following two rules [15]: (1) The number inside each circle dictates how many of the squares touched by the circle be colored blue. (2) After coloring to satisfy condition (1), all uncolored squares form a single contiguous region connected by edges. Note that in Ref. [15], squares are colored black instead of blue; clearly this difference is aesthetic and inconsequential.

An instance of a Creek puzzle and its solution is shown in Fig. 1. The example adheres to a 5 × 5 square grid. The instance has the unique solution shown in Fig. 1.

One of the reasons why people enjoy playing games and puzzles is related to how difficult they are relative to the cognitive abilities of the players. From this perspective, many theoretical computer scientists have studied the computational complexity of various games and puzzles by characterizing them as polynomially solvable or \( \text{NP}\)-hard. It is known that many commonly played puzzles are \( \text{NP}\)-complete and/or \( \text{ASP}\)-complete. For example, in 2009, Hearn and Demaine [5] surveyed the computational complexities of combinatorial games and puzzles. Hashiwokakero [1], Number Link [13], Kurodoko [12], Shikaku and Ripple Effect [17], Yajilin and Country Road [6], Yosenabe [7], Shakashaka [4], Fillmat [18], Hebi, Satogaeri, and Suraromu [13], Pipe link [19], Usowan [8], Dosun-Fuwari [9], and Herugolf and Makaro [10] are examples of pencil-and-paper puzzles published by Nikoli. Recent studies have shown that all of these puzzles are \( \text{NP}\)-complete and/or \( \text{ASP}\)-complete. However, the computational complexity of the Creek puzzle has not previously been studied. Thus, in this paper, we study the complexity of the decision version of the Creek puzzle. Specifically, we will explore Creek puzzles on two types of lattice boards: square and triangular grids (see Fig. 1 and Fig. 2, respectively).

We define the decision problem of the Creek puzzle on a square grid as follows.

Creek Decision Problem

Instance: A square grid \( B \) of size \( m \times n \), with circled integers in \( \{0, 1, 2, 3, 4\} \) placed at some lattice points on \( B \).

Question: Is there a solution, i.e., can some squares in \( B \) be colored blue while satisfying constraints (1) and (2) below?

(1) The number inside each circle dictates how many of the squares touched by the circle be colored blue.

(2) All uncolored squares form a single contiguous region connected by edges.

Fig. 1 Instance of a Creek puzzle on a square grid and its solution.
(1), (2), (3), (4), (5), (6) can be placed at some lattice points in a particular game configuration. We define the decision problem of the Creek puzzle on a triangular grid as follows, following the logic of the original Creek puzzle. We denote this as the 3-Creek Decision Problem in contrast to the decision version of the original puzzle (on square grids) which is the 4-Creek Decision Problem.

3-Creek Decision Problem
Instance: A triangular grid \( B \) of size \( m \times n \), with circled integers in \( \{1, 2, 3, 4, 5, 6\} \) placed at some lattice points on \( B \).
Question: Is there a solution, i.e., can some triangles in \( B \) be colored blue while satisfying constraints (1) and (2) below?
(1) The number inside each circle dictates how many of the triangles touched by the circle be colored blue.
(2) All uncolored triangles form a single contiguous region connected by edges.

An instance of a 3-Creek Decision Problem and its solution is shown in Fig. 2. In this figure, the board size is 4 × 6, and note that \( m \times n \) represents the size of board \( B \) in the same way.

We analyze the computational complexity of the above two Creek puzzle decision problems, including proving the following theorems.

**Theorem 1** The 4-Creek Decision Problem is NP-complete.

**Theorem 2** The 3-Creek Decision Problem is NP-complete.

We prove Theorem 1 in Section 2, followed by proving Theorem 2 in Section 3. It is obvious that the problems belong to \( \mathcal{NP} \), since it can be verified whether a solution candidate is correct for each problem in polynomial time. Therefore, we focus on proofs of the \( \mathcal{NP} \)-hardness of these problems in the following sections.

### 2. Proof of Theorem 1

In this section, we will show the validity of Theorem 1 by proving the \( \mathcal{NP} \)-hardness. To prove the \( \mathcal{NP} \)-hardness, we construct a reduction from the circuit-satisfiability problem (Circuit-SAT for short), which is the problem of deciding whether given Boolean circuits \( C(v_1, v_2, \ldots, v_n) \) with \( n \) Boolean variables \( v_1, v_2, \ldots, v_n \) have an assignment of variables that renders the output true. The Circuit-SAT has already been shown to be \( \mathcal{NP} \)-complete in [3].

In what follows, we construct gadgets corresponding to each part of a Boolean circuit by using only the basic logical gates AND and NOT (note that \{AND, NOT\} is universal), wires, and splits. In the reduction, care must be taken with respect to wires crossing in the circuits. To construct Creek puzzle instances from circuits, we consider Boolean circuits restricted to a plane. For this purpose, we can replace crossing wires with McColl’s planar “cross-over” circuit [14] as shown in Fig. 3. This can be constructed by three XOR gates (note that an XOR gate can be expressed as a “planar” circuit consisting of AND and NOT gates).

The left side of any gadget (except those that achieve vertical adjustments) receives a Boolean value from the adjacent gadget, and the output value is transmitted from the right side of the gadget to the next gadget. The signals that propagate the truth-value assignment of variables are expressed in units of 2 squares in width on the Creek board.

#### 2.1 Input, Wire, and Output Gadgets

An input gadget and an output gadget are shown in Fig. 4 (a) and (b), respectively. The input gadget consists of 2 × 3 squares. The squares that cannot be colored blue due to \( \bigcirc \)'s are shaded gray in the figure. To satisfy condition (1) with respect to \( \bigcirc \), only one of the squares marked \( F \) or \( T \) should be colored blue. Whether to color square \( F \) or \( T \) in Fig. 4 (a) can be approached in terms of the truth-value assignments of variables: Coloring square \( T \) corresponds to a true value; in contrast, coloring square \( F \) corresponds to a false value.

We use the gadget in Fig. 4 (b) as an output gadget. The gadget consists of 2 × 3 squares, like input gadgets. Gray-shaded squares in Fig. 4 (b) have the same meaning as the input gadgets. Note that in the following figures, gray-shaded squares have the same meaning. To satisfy the condition of \( \bigcirc \) on the right side, square \( T \) should be colored blue. There is not a solution in the output gadget if we color square \( F \) blue due to the signal from the left side of the output gadget.
A wire gadget is shown in Fig. 5. The basic elements of the wire gadgets are each composed of $2 \times 2$ squares as shown in Fig. 5 (a), and the wire gadget is realized by arranging these basic elements in a horizontal direction as per Fig. 5 (b). To satisfy the condition of each $\oplus$ in the wire gadgets, it is necessary to color only one of $x$ and $x'$ blue. Moreover, according to the arrangement of $\oplus$ side by side, all $\oplus$'s must be matched to color $x$ or $x'$ blue. Since this behavior corresponds to signal propagation, the construction shown in Fig. 5 is a component that performs the function of a wire of Boolean circuits.

When bending wire gadgets in vertical directions, we use the bend gadgets shown in Fig. 6 and Fig. 7. The bend gadget consists of $3 \times 2$ squares in Fig. 6 (a). By using this gadget, we can connect a normal wire gadget and a wire gadget rotated 90 degrees (see Fig. 6 (b). Note that the part indicated by red lines in Fig. 6 (b) is the bend gadget). It is clear that the signal propagates correctly on the bending wire.

Similarly, when changing the course of wire gadgets to the left, it can be realized by using the gadgets shown in Fig. 7.

### 2.2 Split Gadget

We use the gadget shown in Fig. 8 as a split gadget. This gadget branches a signal wire by vertically combining the same structure as a wire gadget. Like bending wires, according to the continuous arrangement of $\oplus$, either all $x$ or all $x'$ squares must be colored blue. Specifically, if the input signal to the split gadget is true, the true signal propagates in vertical directions, while if the input is false, the false signal is transmitted up and down. Note that by arranging a structure like the bend gadget above and below the split gadget, the signal can travel in a horizontal direction.

### 2.3 NOT Gadget

A NOT gadget is shown in Fig. 9. The NOT gadget consists of $4 \times 10$ squares. Because the signal is represented by a pair of two adjacent squares, the two squares representing the signal at the entry and exit of the gadget correspond to the red frames in Fig. 10.

Like the previous gadgets, according to the arrangement of $\oplus$, either all $x$ or all $x'$ squares must be colored blue. If the input of the NOT gadget from the left gadget is true (see Fig. 10 (a)), $x'$ in the red frame on the right side will be colored blue, which means that the output signal is false. If the input of the NOT gadget is false, the coloring in the NOT gadget is determined, and a false output can be obtained, for the same reason as in the true case. Therefore, the NOT gadget acts as a NOT gate in a circuit.
2.4 AND Gadget

Figure 11 shows the AND gadget for an AND gate $z = x \land y$. According to the values of $x, y$, we have four possible cases. First, we consider the case that inputs $(x, y)$ of the AND gadget from the left and upper gadgets are $(true, true)$ as illustrated in Fig. 12. Signals from $x$ and $y$ propagate until the red 1 in the center of Fig. 12 according to the arrangement of 1. Then, squares a, b, and e cannot be colored blue since $x'$ and $y'$ are colored blue. Thus, $z'$ with red 1 must be colored blue to satisfy condition (1) with respect to the red 1. Next, the output to $z$ is determined to be a true signal by the same operation as the wire gadgets. Consequently, the right adjacent gadget receives the true value when both inputs are true.

Next, we consider the case that the inputs $(x, y)$ of the AND gadget from the left and upper gadgets are $(false, false)$ as illustrated in Fig. 13. Due to the input values (false, false) and the 1's consecutively arranged from $x$ and $y$, it can be seen that the upper left square b of the red 1 is colored blue. Therefore, square $z'$ at the lower right of the red 1 is not colored blue. Moreover, square z that is paired with $z'$ is colored blue. These colorings continue until the output of this gadget. Consequently, the right adjacent gadget receives a false value when $(x, y) = (false, false)$. This coloring corresponds to the behavior of an AND gate with (false, false) as both inputs.

We consider the case that inputs $(x, y) = (true, false)$ of the AND gadget as illustrated in Fig. 14. As discussed in the previous two cases, all $x'$ and $y$ in wire-like parts are colored blue. As a result, square e must be colored blue to satisfy the condition of red 1 (Note that the position of the red 1 is different from the previous two figures). From this, the coloring of squares in the gadget is determined as shown in Fig. 14, and this coloring corresponds to the behavior of an AND gate with (true, false) as both inputs.

When inputs $(x, y) = (false, true)$ of the AND gadget, the coloring of squares in the gadget will be determined as shown in Fig. 15. This is equivalent to the behavior of an AND gate with (false, true) as both inputs.

From the foregoing, the AND gadget in a Creek puzzle corresponds to an AND gate in a circuit. Moreover, there is a single correct way to color squares on the AND gadget in each case. Therefore, feasible colorings on the AND gadget have a one-to-one correspondence with the behavior of an AND gate.

2.5 Proof of Reduction Correctness

All the necessary gadgets we listed are constructed in the previous subsections. In the way described in the previous section, we will obtain instance $B$ of the 4-Creek Decision Problem corresponding to the given Boolean circuit $C$ of the Circuit-SAT. In fact, all gadgets propagate signals in units of two adjacent squares. Therefore, by appropriately arranging each gadget (for example, arranging based on even-numbered coordinates) and connecting each gadget using wires or bending wires, the Boolean circuit $C$ can be constructed on the grid $B$. Moreover, by arranging 0 appropriately in the parts outside the gadgets, $B$ can be a square grid. This reduction can be accomplished in polyno-
mial time of the input size of $C$. It should be noted that the combination of coloring of gadgets shown in the previous sections always satisfies constraint (2) due to the structure of gadgets.

It is straightforward that the 4-Creek Decision Problem has a solution if and only if the original circuit has an assignment of variables that renders the output true. The Creek Decision Problem obtained by the reduction correctly simulates a Boolean circuit. Therefore, this problem is NP-complete. Moreover, once we fix an assignment of the variables of $C$, the obtained 4-Creek is uniquely determined. That is, colorings of squares on $B$ of 4-Creek have a one-to-one correspondence with the behavior of the original Boolean circuit $C$. Therefore, the another-solution-problem variant (ASP for short) of 4-Creek is ASP-complete, since the ASP version of the Circuit-SAT is ASP-complete [3] (see Ref. [20] for definitions of ASP and ASP-completeness).

3. Proof of Theorem 2

In this section, the proof of Theorem 2 will be shown briefly in the same way as in the previous section. To prove NP-hardness, input, wire, output, split, NOT, and AND gadgets are described in the following subsections. It should be noted that conditions (1) and (2) are slightly different from those of the normal board (square grid) depending on the difference in board shape (recall the problem definitions in Section 1). In all the following figures, the triangles with a tilde mark cannot be colored blue due to the surrounding conditions, regardless of the propagating signal. Note that all gadgets are designed in units of width 6 except Output gadget.

3.1 Input, Wire, and Output Gadgets on a Triangular Grid

Input, wire, and output gadgets are shown in Fig. 16(a), (b), and (c), respectively. In the input gadget, we can select whether triangle $T$ or $F$ is colored blue. The selection at the input gadget is transmitted by the wire gadget as shown in Fig. 17. When $T$ is selected, to satisfy condition (1) in the green1⃝s, the triangles $y$s are colored blue. Otherwise, the $x$s are colored blue for the red1⃝s.

Since there is a $@$ in the output gadget, it is necessary to color the 6 triangles around it blue. Only when a true signal from the wire gadget reaches the output gadget can the output part be colored as shown in Fig. 18 without failing against condition (1).

The wire gadget extending in the vertical direction is shown in Fig. 19. When bending wire gadgets in the vertical direction, we use the bend gadgets shown in Fig. 20. Note that even if the wire gadget is bent using the bend gadgets, the signal expression in units of 6 triangles wide is maintained (see Fig. 21 and Fig. 22).
3.2 Split Gadget on a Triangular Grid

We use the gadget shown in Fig. 23 as a split gadget on a triangular grid. When a true signal is input to the split gadget, that signal is branched and transmitted rightward and downward as shown in Fig. 24. On the other hand, when a false signal is input, that signal is transmitted in two directions as shown in Fig. 25.

3.3 NOT Gadget on a Triangular Grid

A NOT gadget on a triangular grid is shown in Fig. 26. The NOT gadget consists of $8 \times 18$ triangles with two $\mathbf{b}$s, and it has been determined that the triangles around $\mathbf{b}$s should be colored blue.

We consider that the input of the NOT gadget from the left gadget is true (see Fig. 27(a)). For the red $\mathbf{b}$, it is possible to satisfy condition (1) by coloring the triangle above it blue. However, in that case, for the $\mathbf{b}$ to the right of the red $\mathbf{b}$, the upper right triangle must be colored blue. This results in a situation that contradicts condition (2). As a result, for the true input, all $y$ triangles will be colored blue, which means that the output signal is false.

If the input of the NOT gadget from the left gadget is false (see Fig. 27(b)), a true output is obtained as shown in the figure.

3.4 AND Gadget on a Triangular Grid

Figure 28 shows the AND gadget on a triangular grid for an AND gate $z = x \land y$. As per the square grid, according to the values of $x, y$ we have four possible cases.

When the inputs $(x, y)$ of the AND gadget from the left and upper gadgets are (true, true), the coloring of the triangles in the gadget will be determined as shown in Fig. 29. Consequently, the right adjacent gadget receives the true value when both inputs are true.

Next, we consider the case that the inputs $(x, y)$ of the AND gadget from the left and upper gadgets are (true, false) as illustrated in Fig. 30. According to the inputs, the $y$ and $Y$ triangles are colored blue as shown in the figure. After that, there are two possible ways to color the triangles around the red $\mathbf{a}$ and $\mathbf{c}$: the way to color triangle $\mathbf{a}$ and the way to color triangles $\mathbf{a}$ and $\mathbf{k}$. However, the latter violates condition (2) because triangle $\mathbf{a}$ is isolated. Therefore, in this case, a false signal is output to the
right gadget.

We consider the case that the inputs \((x, y)\) are \((false, true)\) as illustrated in Fig. 31. As in the previous case \((x, y) = (true, false)\), a false signal is output because the way of coloring around the central red \(\Box\) and \(\Box\) is determined.

We consider the case that the inputs \((x, y)\) are \((false, false)\) as illustrated in Fig. 32. According to the inputs, the \(X\) and \(X\) triangles are colored blue as shown in the figure. After that, the way to color around \(\Box\) is uniquely determined, and triangles \(a\) and \(b\)
are colored blue. Therefore, in this case, a false signal is output to the right gadget.

3.5 Proof of Reduction Correctness for 3-Creek

All the necessary gadgets we listed are constructed in the previous subsections. Similar to the method of constructing the grid shown in Section 2.5, each gadget is placed while paying attention to the parity of the arrangement position. Note that by arranging the gadgets in this way, each gadget can be connected by using wires or bending wires. The validity of the proof is omitted because it is the same as in Section 2.5. The proof of Theorem 2 is complete, which also means that the ASP version of 3-Creek is ASP-complete.

4. Conclusions

In this paper, we have studied the computational complexity of Creek puzzles and proved that 4-Creek and 3-Creek are NP-complete and ASP-complete by reducing the Circuit-SAT problem to Creek puzzles.

There are only three shapes that can form regular tilings: the equilateral triangle, square, and regular hexagon. Any one of these three shapes can be duplicated infinitely to fill a plane with no gaps. In this paper, we proved the computational complexity of Creek puzzles on square and triangular grids. We anticipate that the same method can be used to prove NP-completeness and ASP-completeness for the hexagonal grid, which is the remaining planar filling.

Acknowledgments This research was partially supported by the Institute of Informatics, Osaka Electro-Communication University.

References

[1] Andersson, D.: Hashiwokakero is NP-complete, Information Processing Letters, Vol.109, No.19, pp.1145–1146 (2009).
[2] Battista, G.D., Eades, P., Tamassia, R. and Tollis, I.G.: Algorithm for Drawing Graphs: An Annotated Bibliography, Computational Geometry, Vol.4, No.5, pp.235–284 (1994).
[3] Cook, C.S.: The complexity of theorem proving procedures, 3rd ACM Symposium on Theory of Computing, pp.151–158 (1971).
[4] Demaine, E.D., Okamoto, Y., Uehara, R. and Uno, Y.: Computational Complexity and an Integer Programming Model of Shakashaka, IEICE Trans. Fundamentals of Electronics, Communications and Computer Science, Vol.E97-A, No.6, pp.1213–1219 (2014).
[5] Hearne, R.A. and Demaine, E.D.: Game, Puzzles, & Computation, A.K. Peters Ltd., MA, USA (2009).
[6] Ishibashi, A., Sato, Y. and Iwata, S.: NP-completeness of Two Pencil Puzzles: Yajilin and Country Road, Utilitas Mathematica, Vol.88, pp.237–246 (2012).
[7] Iwamoto, C.: Yosebane is NP-complete, Journal of Information Processing, Vol.22, No.1, pp.40–43 (2014).
[8] Iwamoto, C. and Haruiishi, M.: Computational Complexity of Usowan Puzzles, IEICE Trans. Fundamentals of Electronics, Communications and Computer Sciences, Vol.E101-A, No.9, pp.1537–1540 (2018).
[9] Iwamoto, C. and Ibusuki, T.: Dosun-Fuwarin is NP-complete, Journal of Information Processing, Vol.26, pp.358–361 (2018).
[10] Iwamoto, C., Haruiishi, M. and Ibusuki, T.: Computational Complexity of Heragof and Makaro, IEICE Trans. Fundamentals of Electronics, Communications and Computer Sciences, Vol.E102-A, No.9, pp.1118–1125 (2019).
[11] Kanehiro, S. and Takenaga, Y.: Satogai, Hebi and Suraromu are NP-complete, Proc. 3rd International Conference on Applied Computing and Information Technology, pp.47–52 (2015).
[12] Kökler, J.: Kurodoko is NP-complete, Journal of Information Processing, Vol.20, No.3, pp.694–706 (2012).
[13] Kotsuma, K. and Takenaga, Y.: NP-completeness and Enumeration of Number Link Puzzle, IEICE Technical Report, Vol.109, No.465, pp.1–7 (2010) (in Japanese).
[14] McColl, W.: Planar crossovers, IEEE Trans. Computers, Vol.30, No.2, pp.223–225 (1981).
[15] Nikoli: Rules of Creek (online, in Japanese), available from ⟨https://www.nikoli.co.jp/puzzles/creek/⟩ (accessed 2019-09-03).
[16] Seta, T.: The Complexities of Puzzles, CROSS SUM, and Their Another Solution Problems (ASP), Senior Thesis, University of Tokyo (2002).
[17] Takenaga, Y., Aoyagi, S., Iwata, S. and Kasai, T.: Shikaku and Ripple Effect are NP-Complete, Congressus Numerantium, Vol.216, pp.119–127 (2013).
[18] Uejima, A. and Suzuki, H.: Fillimat is NP-Complete and ASP-Complete, Journal of Information Processing, Vol.23, No.3, pp.310–316 (2015).
[19] Uejima, A., Suzuki, H. and Okada, A.: The complexity of generalized pipe link puzzles, Journal of Information Processing, Vol.25, pp.724–729 (2017).
[20] Yato, T. and Seta, T.: Complexity and Completeness of Finding Another Solution and its Application to Puzzles, IEICE Trans. Fundamentals of Electronics, Communications and Computer Sciences, Vol.E86-A, No.5, pp.1052–1060 (2003).

Akihiro Uejima was born in 1975. He received his B.E. and M.E. degrees in information systems engineering from the Department of Information and Computer Sciences, Toyohashi University of Technology in 1998 and 2000, respectively, and a Dr. of Informatics degree from the Department of Communications and Computer Engineering, Graduate School of Informatics at Kyoto University in 2005. He was a lecturer during 2005–2013, and has been an associate professor since 2013 in the Department of Engineering Informatics, Osaka Electro-Communication University. His research interests include graph theory and computational complexity. He is a member of the Institute of Electronics, Information and Communication Engineers, the Information Processing Society of Japan, the Operations Research Society of Japan, the Language and Automaton Symposium.

Kentaro Oe was born in 1994. He received a B.Sc. from the Department of Engineering Informatics, Faculty of Information and Communication Engineering, Osaka Electro-Communication University in 2016.