On The NP-Completeness of The Nurikabe Pencil Puzzle and Variants Thereof

Markus Holzer[1] and Andreas Klein[2] and Martin Kutrib[3]

[1] Institut für Informatik, Technische Universität München,
Boltzmannstraße 3, D-85748 Garching bei München, Germany
email: holzer@informatik.tu-muenchen.de
[2] Fachbereich für Mathematik und Informatik, Universität Kassel,
Heinrich Plett Straße 40, D-34132 Kassel, Germany
email: klein@mathematik.uni-kassel.de
[3] Institut für Informatik, Universität Gießen,
Arndtstraße 2, D-35392 Gießen, Germany
email: kutrib@informatik.uni-giessen.de

Abstract. We show that the popular pencil puzzle Nurikabe is intractable from the computational complexity point of view, i.e., is NP-complete, even when the involved numbers are 1 and 2 only. To this end we show how to simulate Boolean gates by the puzzle under consideration. Moreover, we also study some Nurikabe variants, which remain NP-complete, too.

1 Introduction

Nurikabe (engl. coating wall) is a solitaire puzzle, which was invented by *Nikoli Inc.*, a Japanese publisher—Nikoli is publisher's name of Japan's No. 1 puzzle magazine Nikoli, which contains a wide variety of puzzles like, e.g., visual puzzles, word puzzles, and number puzzles. The Nurikabe puzzle is played on a finite rectangular two-dimensional grid, with some cells labelled with natural numbers. The goal is to fill the cells black or white according to the following rules:

1. Cells containing numbers are filled white.
2. A natural number defines a number of connected white cells—they are linked either horizontally or vertically. Each area of white cells contains only one natural number in it, and they are separated by black cells. Diagonal connections do not count. White cells will be called *island* or *land* in the forthcoming.
3. All black cells are linked to be a connected area. Diagonal connections do not count. Black cells will be called *sea* or *water*.
4. Black cells cannot be linked to form a 2×2 square.

An example of Nurikabe and a solution is shown in Figure 1. How to obtain this solution is shown in the Appendix. The reader is encouraged to verify that the solution to the given puzzle is unique. Implementations of Nurikabe are available on the Internet; e.g., for the PALM Pilot a version can be downloaded from `http://www.palmgamingworld.com/puzzle/nurikabe.shtml`. It is worth mentioning that Nurikabe can be played online at `http://www.puzzle.jp`.

Formally, Nurikabe falls into the category of *pencil* puzzles. Pencil puzzles (or pencil-and-paper puzzles) are those offered as some figure on the paper and solved by drawing on the figure with the pencil. Many pencil puzzles are originated in Japan, are quite popular there, but less known outside of Japan. This is one reason, why only a few complexity results for these type of puzzles is known. Only recently, some researchers have tried to fill this gap in the literature. To our knowledge, the first result on pencil puzzles is due to Ueda and Nagao [7], who showed that Nonogram is NP-complete. Here NP denotes the class of problems solvable in polynomial time on a nondeterministic Turing machine. Since pencil puzzles are hard to solve, but the verification of a solution is easy, most of

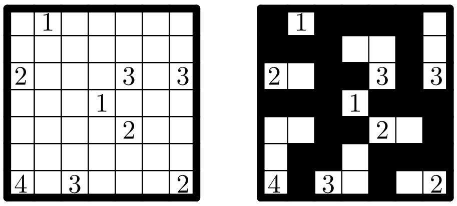


Fig. 1. NURIKABE puzzle and a solution.

them are contained in NP and are even NP-complete. For the present, the following pencil puzzles are known to be NP-complete: Slither Link [9], Cross Sum (jap. *Kakkuro*) [6], Number Place (jap. *Sudoku*) [10], Fillomino [10], Pearl [2], Corral [1], and Spiral Galaxies [3]. In this paper we contribute to this list, namely by showing that NURIKABE is intractable, too, by proving the following theorem:

Theorem 1. *Solving a* NURIKABE *puzzle is* NP*-complete (under deterministic logspace many-one reductions), even when restricting to instances whose cells may contain only numbers* 1 *or* 2*.*

To this end, we show how to simulate Boolean gates *via* NURIKABE puzzles. Observe, that the given result is optimal with respect to the involved numbers, since a NURIKABE puzzle whose cells may contain only the numbers 1 is solvable in deterministic polynomial time. Moreover, we discuss also some variants of NURIKABE, where the (artificial) third and fourth rule of the game description, i.e., black cells form a connected area and black cells cannot be linked to be 2×2 squares, are present and/or absent. In this case, it is shown, that the pencil puzzle game remains intractable. We assume the reader to be familiar with the basics in complexity theory as contained in [5]. Hardness and completeness are always meant with respect to deterministic many-one log-space reducibilities.

2 NURIKABE is Intractable

In this section we show that finding a solution to a given NURIKABE instance is NP-complete, thus proving Theorem 1. The containment within NP is immediate, since it is obvious that a Turing machine firstly can guess a black and white pattern, and secondly verifies within polynomial time that it is a valid solution. Thus it remains to prove hardness. To this end, we show how to reduce planar 3SAT to NURIKABE. Planar 3SAT was shown to be NP-complete in [4] and is defined as follows:

Instance: A set of Boolean variables $X = \{x_1, \ldots, x_n\}$ and a set of clauses $C = \{c_1, \ldots, c_m\}$. Furthermore, the bipartite graph $G = (X \cup C, E)$ is planar, where $E = \{\, (x_i, c_j) \mid x_i \in c_j \text{ or } \bar{x}_i \in c_j \,\}$.

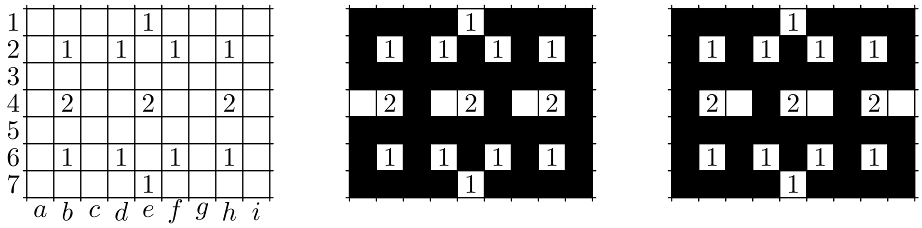
Question: Is there an assignment for the variables such that all clauses are satisfied?

Obviously, the associated graph to a formula can be easily transformed into a Boolean circuit with multiple output gates or nodes. This is done, by replacing a "variable" vertex by an input gate, a "clause" vertex by an OR-gate—to be more precise, by a cascade of OR-gates, and finally inserting a NOT-gate between an input gate and an OR-gate if necessary. The output for all OR-gates is true if and only if planar 3SAT is satisfiable. Observe, that if the graph is planar, then also the Boolean circuit is planar. Since three XOR-gates can simulate the crossover of two wires, the restriction to planar circuits is not essential.

Now we are ready to present our reduction. A Boolean circuit with AND-, OR-, and NOT-gates can be constructed from the following components: Wires carrying Boolean values, input and output gates, signal splitters, and OR- and NOT-gates. AND-gates are not necessary, since they can be simulated by DeMorgan's Law with OR- and NOT-gates. Moreover, since the circuits under consideration are planar, crossover of wires does not need to be considered.

For each component we construct a Nurikabe sub-puzzle, but before doing this, we explain how to treat cells which are *not* involved in the presentation of the circuit. First of all, we present our Nurikabe sub-puzzles such that the black cells form a single connected component—this significantly simplifies presentation, since under this constraint, the black cells always form a connected area. Moreover, we use a chess like coordinate system to identify cells. Blocks that are not treated in the circuit presentation have to filled black and white appropriately. This can be done simply by placing a large enough number, which depends on the free space to be filled, such that the whole region will be filled white. Another solution would be to appropriately place 1's and 2's. Here the argumentation that the whole black area remains connected is more involved. Nevertheless, at the end of this section we discuss this problem in more detail, in order to obtain an optimal result.

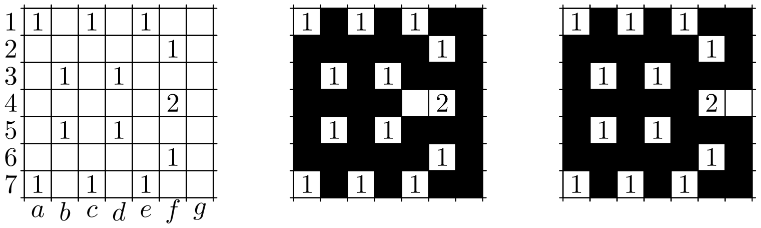
Now we concentrate on how to build Nurikabe sub-puzzles for the basics. A straight wire with information flow from left to right is shown in Figure 2. Observe, that the white space around the 2's must be horizontal. This is easily seen, since a vertical white space is either blocked by a 1 or would disconnect the black space, e.g., $e2$ or $e6$. Now the horizontal orientation forces that either $a4$, $d4$, and $g4$ or $c4$, $f4$, and $i4$ are white. Thus, this two local solutions of the Nurikabe sub-puzzle correspond to "true" and "false." We say, that a wire carries "true" whenever the white space follows the 2's (in direction of information flow) in a trailer like fashion, and "false" otherwise. Extending a wire to the left or right or connecting some other to be described sub-puzzle is always done by overlapping the vertical 1-2-1 pattern on the left or right.



(a) Sub-puzzle wire (b) Solution "true" (c) Solution "false"

Fig. 2. A straight wire with information flow from left to right and its solutions.

Adapting the straight wire one obtains an input-gate or an output-gate, see Figures 3 and 4.



(a) Input-gate

(b) Solution “true”

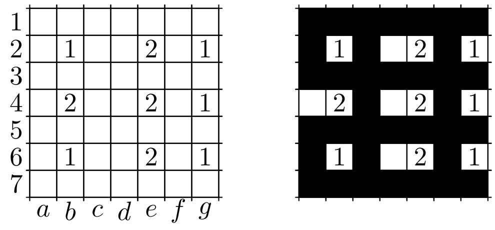
(c) Solution “false”

Fig. 3. Input-gate with information flow from left to right.

The former Nurikabe sub-puzzle has two local solutions and thus can either produce “true” or “false” as input, while the latter one has only *one* solution, which corresponds to verifying whether the wire carries “true.”

Next, we show how to split a wire—repeated use allows a wire to be splitted several times. The Nurikabe sub-puzzle realizing this split is depicted in Figure 5. The input is on the left, while the output is on top and bottom. By similar reasons as in the wire case, the white cells around the 2’s must be oriented in flow of information direction. We distinguish two cases: (1) The cell $a7$ and thus $d7$ is white. Now consider the 2×2 areas $f7, f6, g7, g6$ and $f8, f7, g8, g7$. By the 2×2 rule on black cells in both areas at least one cell must be white. This forces both $f6$ and $f8$ to become white space. Then the cells $f3$ and $f11$ are triggered to be white, too. (2) The cell $c7$ and thus $f7$ is white. Then the only way to arrange the white spaces around the 2’s at $f5$ and $f9$ is to fill $f4$ and $f10$ white, which, in turn, triggers $f1$ and $f13$. This completes the description of the splitter device. Observe, that the splitter can also be used to bend a wire. In this case, the unused wire must be terminated, which can be done by reverting (vertical mirror) the input-gate sub-puzzle, which was already shown in Figure 3.

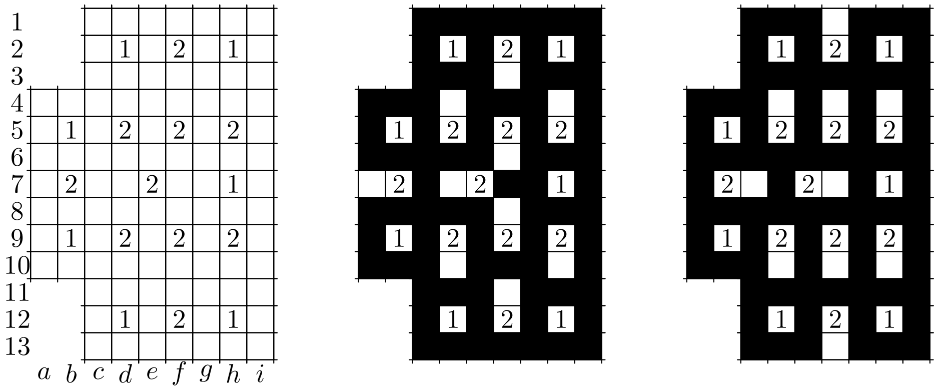
Finally, we describe the Boolean gates. The Nurikabe sub-puzzle for the NOT-gate is drawn in Figure 6. For the analysis of the sub-puzzle we again consider two cases: (1) The cell $a4$ is white. By the 2×2 rule we deduce that the area $c4, c5, d4, d5$ must contain at least one white cell. This can be only achieved by the 2 at $d3$, which implies that $d4$ is white. Again by the 2×2 rule considering $e3, e4, f3, f4$ the white space around the 2 at $f5$ is forced to be vertical. Therefore, cell $i4$ must be



(a) Output-gate

(b) Solution “true”

Fig. 4. Output-gate with information flow from left to right.

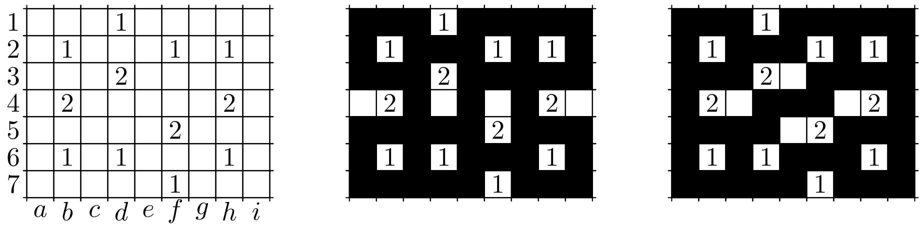


(a) Sub-puzzle splitter

(b) Solution "true"

(c) Solution "false"

Fig. 5. A signal splitter—input from left, and outputs at the top and bottom.



(a) Sub-puzzle NOT-gate

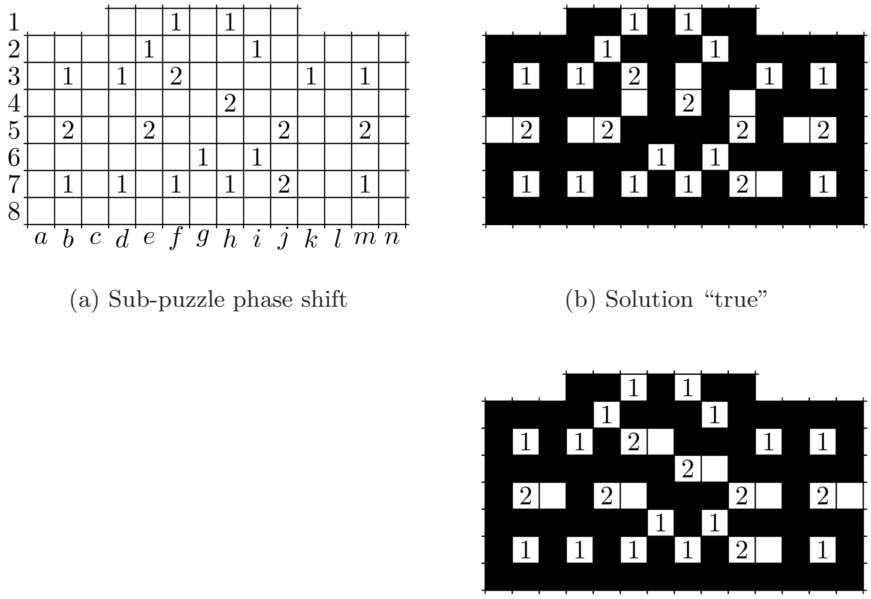
(b) Solution "true"

(c) Solution "false"

Fig. 6. NOT-gate with information flow from left to right.

white. This shows that a “true” input is transformed into a “false” output. (2) The cell at $c4$ is white. Then the only way for the white space triggered by the 2 at $d3$ is to fill $e3$. The 2×2 rule applied to $d4, d5, e4, d5$ gives that $e5$ is white, and in turn $g4$ is white. This shows that a “false” input is transformed into a “true” output.

Before we can describe the OR-gate Nurikabe sub-puzzle, we have to overcome some problem. The information in the wire is carried by the 2’s which appear periodically, and imposes that both wires that are connected to the OR-gate sub-puzzle must have the same periodicity. This may be not the case, since bending wires destroys the original synchronicity. To resolve this difficulty we use a phase-shift-gate, which rearranges the periodicity of a wire and is shown in Figure 7. We show the solution drawing, but leave it as an exercise for the reader to verify its correctness, since the arguments are quite similar as in the NOT-gate case.



	a	b	c	d	e	f	g	h	i	j	k	l	m	n
1						1		1						
2					1				1					
3		1		1		2					1		1	
4								2						
5		2			2					2			2	
6							1		1					
7		1		1		1		1		2			1	
8														

(a) Sub-puzzle phase shift

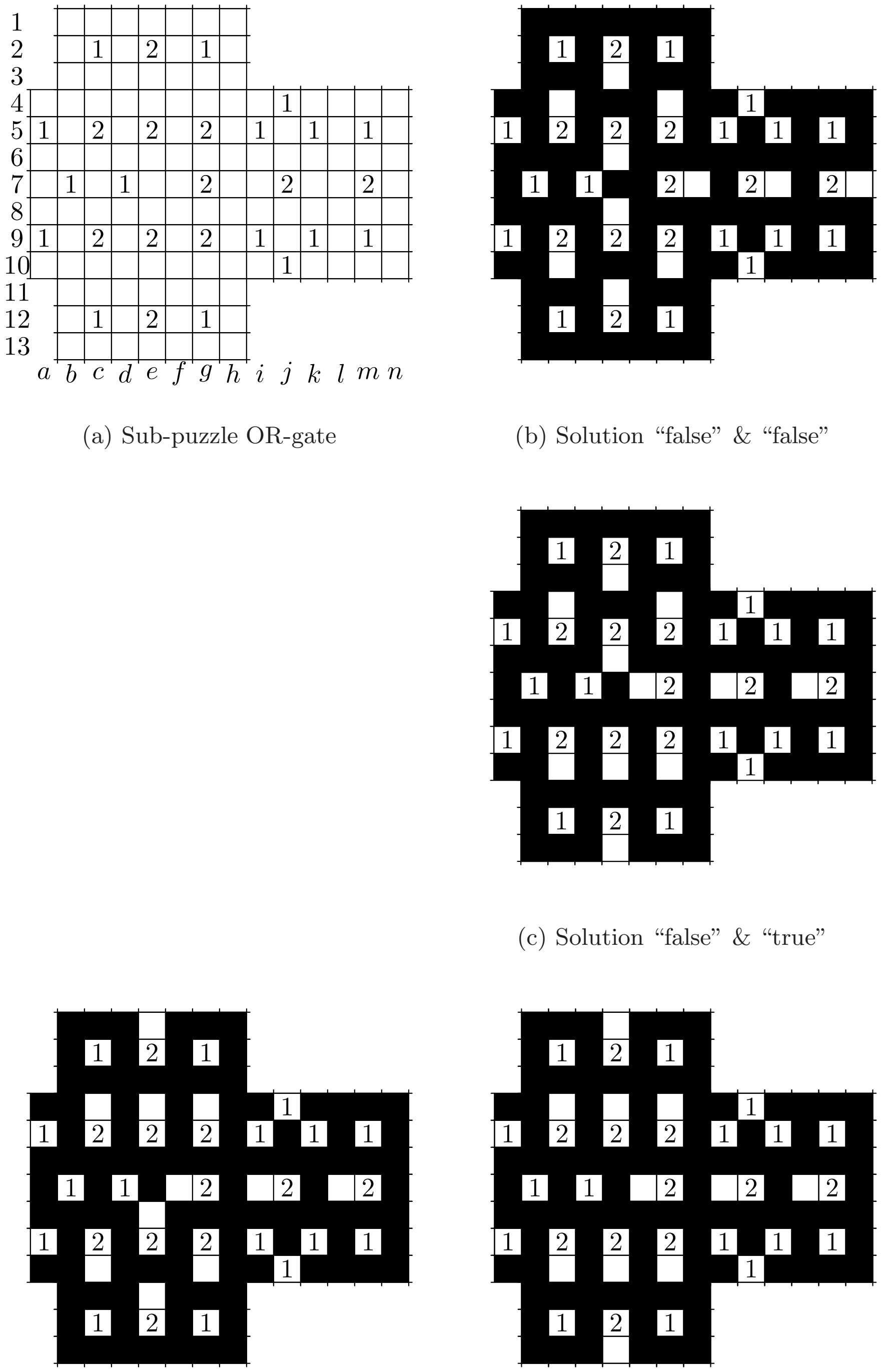
(b) Solution “true”

(c) Solution “false”

Fig. 7. Phase shifter with information flow from left to right.

Now we are ready for the OR-gate Nurikabe sub-puzzle, which is depicted in Figure 8. One observes, that the information carrying 2’s can only be arranged in information flow direction. Then consider $e7$. Whenever the white space induced by $e5$ reaches $e6$, and the white space from $l9$ goes to $l8$, the cell $e7$ is not allowed to be isolated. Thus, this triggers $h7$, $k7$, and $n7$ to be white. Therefore, when both inputs are “false,” the output is “false,” too. If at least one input wire carries “true”, then consider either the 2×2 square $e6, e7, f6, f7$ or $e7, e8, f7, f8$. Since at least one cell must be white, the white space around $g7$ must continue to the left. This triggers $i7$, and $l7$ to be white. Therefore, when at least one input is “true,” then the output is also “true.” This completes the description of the OR-gate, and shows that finding a solution to a Nurikabe puzzle is NP-complete.

In order to obtain an optimal result, we have to reconsider how to treat cells which are not involved in the presentation of the circuit. Instead of placing a large enough number, we have to show how to appropriately place 1’s and 2’s, such that the space is filled and the black cells remain connected. This is always possible, when the to be filled space is large enough. To keep the



	a	b	c	d	e	f	g	h	i	j	k	l	m	n
1														
2			1		2		1							
3														
4										1				
5	1		2		2		2		1		1		1	
6														
7		1		1			2			2			2	
8														
9	1		2		2		2		1		1		1	
10										1				
11														
12			1		2		1							
13														

(a) Sub-puzzle OR-gate

(b) Solution "false" & "false"

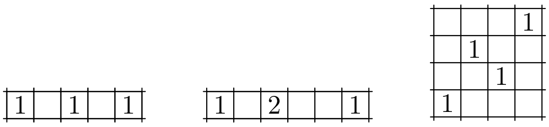
(c) Solution "false" & "true"

(d) Solution "true" & "false"

(e) Solution "true" & "true"

Fig. 8. OR-gate—input from top and bottom, and output goes right.

presentation simple, we may assume that all gates of the circuit are surrounded by cells containing 1's and a layer of black cells, since it is an easy task to extend the presented drawing accordingly. Then the idea is to decompose the to be filled region into rectangular boxes, which are filled line by line from top and bottom simultaneously leaving blank lines in between. By adjusting the length of the wires we can always achieve that the size of all rectangular boxes are at least 5 × 5 (not counting the surrounding black cells). First we show how to fill a single line in a recursive manner: If the line length is one or two, respectively, then label the first cell with 1 or 2, respectively, and terminate. Otherwise label the first cell of the line with 1, cut off two cells from the front, and if the remaining length is one or two we recursively proceed with the remaining part. Otherwise, we label the last cell with 1, cut off two cells from the end, and recursively proceed with the remaining part. Thus, in principle the following two patterns shown in Figures 9(a) and 9(b)—odd and even length, respectively—appear.

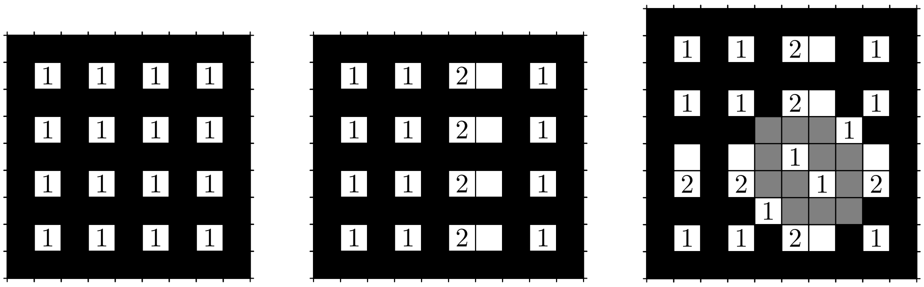


(a) Odd line

(b) Even line

(c) 4 × 4 region

Fig. 9. Nurikabe patterns.



(a) Odd times odd

(b) Odd times even

(c) Even times even

Fig. 10. How to fill a rectangular box—the surrounding black cells don't count.

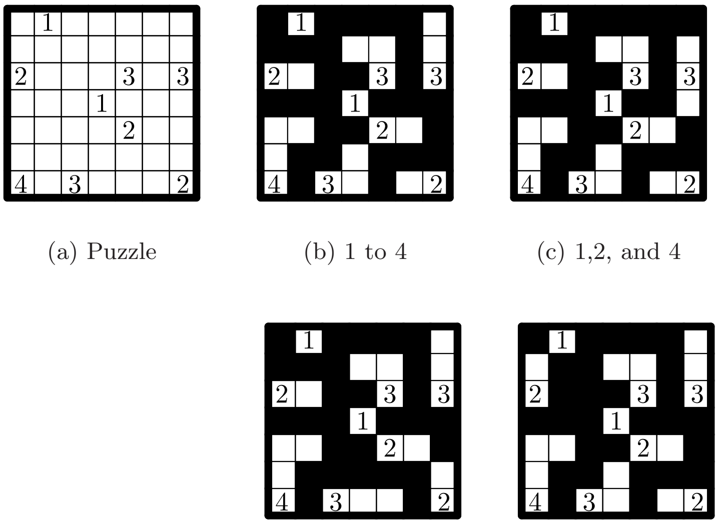
A rectangular box is filled as follows: Compute the patterns according to the above given recursive procedure for the first line and column. Draw the line pattern into the first line and the column pattern into the first column. Then place the patterns repeatedly into the considered rectangular region such that they fit (in a rectangular fashion) to the already drawn patterns. Except for the case where the region was of even times even length, the algorithm terminates. In the remaining case a 4 × 4 region remains (in the intersection of the 2's), which is filled with the pattern depicted in Figure 9(c). Some examples on how to fill a rectangular box are shown in Figure 10—the missing case "even times odd" is similar to the present case "odd times even."

Then one can show that the constructed puzzle for the rectangular region has a unique solution. Thus, the number of solutions of the constructed NURIKABE puzzle that simulates a Boolean circuit is not affected. Since there is a one-to-one correspondence between satisfying assignments of the Boolean circuit and the constrcucted NURIKABE puzzle, we have proven Theorem 1 and moreover the following stronger result. Here #P refers to the corresponding counting version of NP, that is, the class of functions f for which there is a nondeterministic polynomial time bounded Turing machine M, such that $f(x)$ equals the number of accepting computations of M on input x—for further details we refer to [8].

Theorem 2. *Computing the number of solutions to a* NURIKABE *puzzle is #P-complete.* □

3 NURIKABE Variants and Their Complexity

We consider variants of the NURIKABE puzzle, where the Rules 3 and 4 are absent or present—this induces four NURIKABE variants. Observe, that the intractability proof from the previous section was heavily based on these two rules. For instance, the OR-gate construction without the connected black cell rule as shown in Figure 8 may produce "true" as output although the input is "false" and "false." How the rule sets affect the solutions is shown in Figure 11 for the example puzzle depicted in Figure 1. Note that Figure 11(b) shows the solution of the original NURIKABE version.


(a) Puzzle

(b) 1 to 4

(c) 1,2, and 4

(d) 1,2, and 3

(e) 1 and 2

Fig. 11. NURIKABE puzzle solutions with respect to different rule sets—the numbers indicate the used rules.

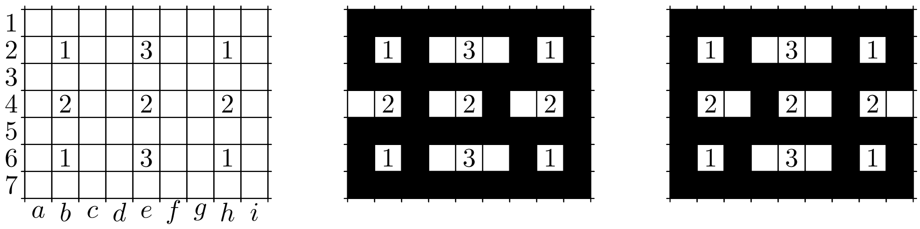
Surprisingly it turns out that all variants are NP-complete, which reads as follows:

Theorem 3. *Solving a* NURIKABE *puzzle with or without the rules*

1. *black cells have to be linked to form a connected area, and*
2. *black cells cannot be linked to form* 2×2 *squares,*

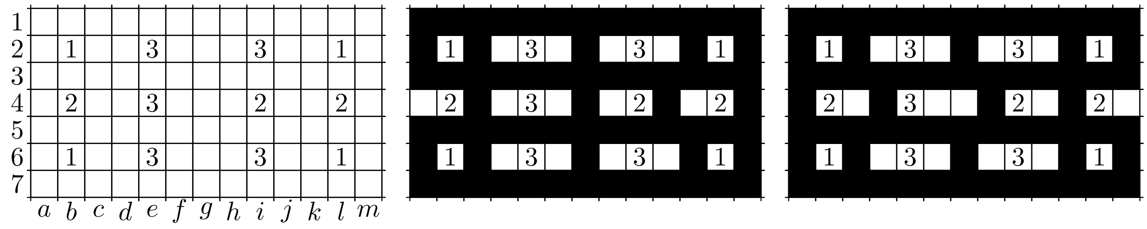
is NP-complete.

It remains to investigate the non-original versions. First we consider the case when playing Nurikabe with only the Rules 1,2, and 4, i.e., compared to the original puzzle version the black area need not to be connected anymore—compare with Figure 11(c). We proceed as in the proof for the original Nurikabe puzzle showing NP-completeness, too. A careful analysis of the gates from the previous section reveals, that only the following sub-puzzles have made use of Rule 3: The wire, the phase shift, and the OR-gate. In order to overcome this situation, we give new implementations of the aforementioned puzzles—instead of the OR-gate we will describe an AND-gate. The wire and phase shift are depicted in Figure 12 and 13, respectively.



(a) Sub-puzzle wire (b) Solution “true” (c) Solution “false”

Fig. 12. A straight wire with information flow from left to right and its solutions in (the absence of Rule 3 and) the presence of Rules 1,2, and 4.

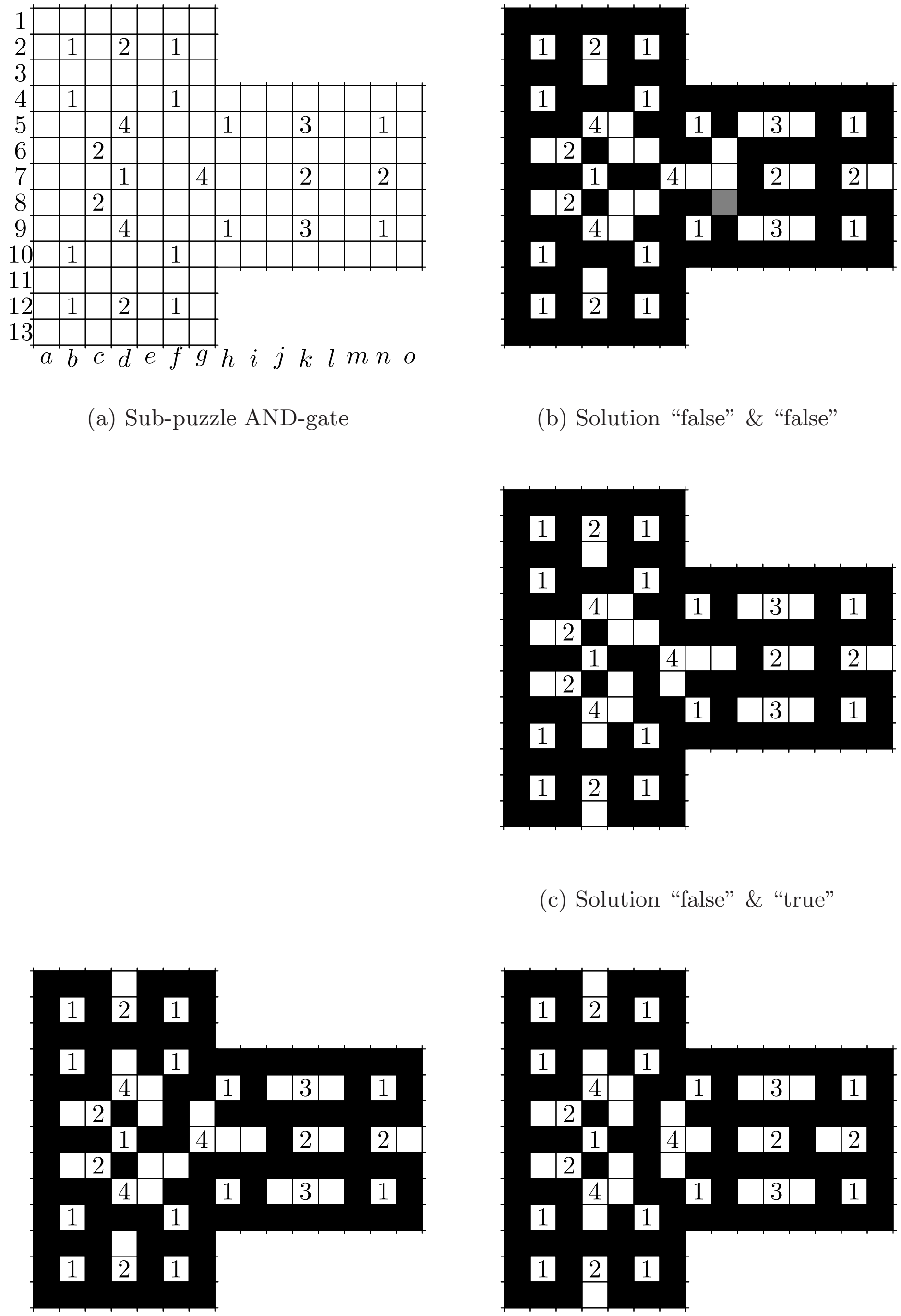


(a) Sub-puzzle phase shift (b) Solution “true” (c) Solution “false”

Fig. 13. A phase shifter with information flow from left to right and its solutions in (the absence of Rule 3 and) the presence of Rules 1,2, and 4.

The analysis of the correctness of the sample solutions is left to the reader. Thus, it remains to give the implementation of an AND-gate. It is depicted in Figure 14.

Note that the information carrying 2's can only be arranged in information flow direction, and in any case both 2's at $c6$ and $c8$ have to be drawn as depicted. Then consider the 2×2 square $f5$, $f6$, $g5$, and $g6$. Since at least one cell must be white either the 4 at $d5$ or $g7$ will contribute a white cell. For the 2×2 square $f8$, $f9$, $g8$, $g9$ it will be either the 4 at $d9$ or $g7$ will contribute a white cell. Then we distinguish two cases: (1) The 4's at $d5$ and $d9$ contribute at least *one* white cell to the above considered 2×2 squares. This means that at least one input wire carries “false.” Thus, this triggers that the 4 at $g7$ must and can contribute at most *one* white cell to the aforementioned 2×2 squares. Therefore, at least two cells (and at most three) to the right of $g7$ have to be left white.



	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
1							
2		1		2		1	
3							
4		1				1									
5				4				1			3			1	
6			2												
7				1			4				2			2	
8			2												
9				4				1			3			1	
10		1				1									
11							
12		1		2		1	
13							

(a) Sub-puzzle AND-gate

(b) Solution “false” & “false”

(c) Solution “false” & “true”

(d) Solution “true” & “false”

(e) Solution “true” & “true”

Fig. 14. AND-gate—input from top and bottom, and output goes right—and its solutions in the absence of Rule 3 and the presence of Rules 1,2, and 4.

In both cases cell $i7$ is white. This forces the 2 at $k7$ to flip to the right, i.e., cell $l7$ must be white. Therefore, when at least one input is "false,", then the output is "false." (2) None of the 4's at $d5$ and $d9$ contribute a white cell to the 2×2 squares $f5$, $f6$, $g5$, $g6$ and $f8$, $f9$, $g8$, $g9$. This situation appears when both input wires carry "true." The only way to construct a valid puzzle is that the 4 at $g7$ constributes a white cell to each of the above mentioned 2×2 squares. This implies that only one cell to the right of $g7$ will be white, which triggers the white space around the 2 at $k7$ to be at $j7$. Thus, if both inputs are "true," then the output is "true," too. This completes the description of the AND-gate. Thus, we have shown the following result.

Lemma 4. *Solving a* NURIKABE *puzzle, when played with the Rules 1,2, and 4, but not with Rule 3, is NP-complete.* □

What about the remaining two cases, when the 2×2 rule is missing? Surprisingly it turns out that these variants are NP-complete, too. Observe, that here the situation is much more involved compared to the absence of Rule 3. This is due to the fact that without the 2×2 rule water can drain of spontaneously. For instance even the wire construction as shown in Figure 2 (or Figure 12) is useless to transport binary information, since the two blocks transporting the "true" or "false" value may flip arbitrarly from "true" to "false" and *vice versa*. Nevertheless, we are able to succeed but due to the lack of space we have to omit formal proofs, which are quite involved, and state only the result.

Lemma 5. *Solving a* NURIKABE *puzzle, when played with (1) the Rules 1,2, and 3, but not with Rule 4 or (2) the Rules 1 and 2, but not with both Rules 3 and 4, is NP-complete.* □

This proves Theorem 3, the main result of this section. As the reader may have noticed, the given constructions are not optimal compared to the puzzles from the previous section, where the NP-completeness of the original NURIKABE puzzle was shown.

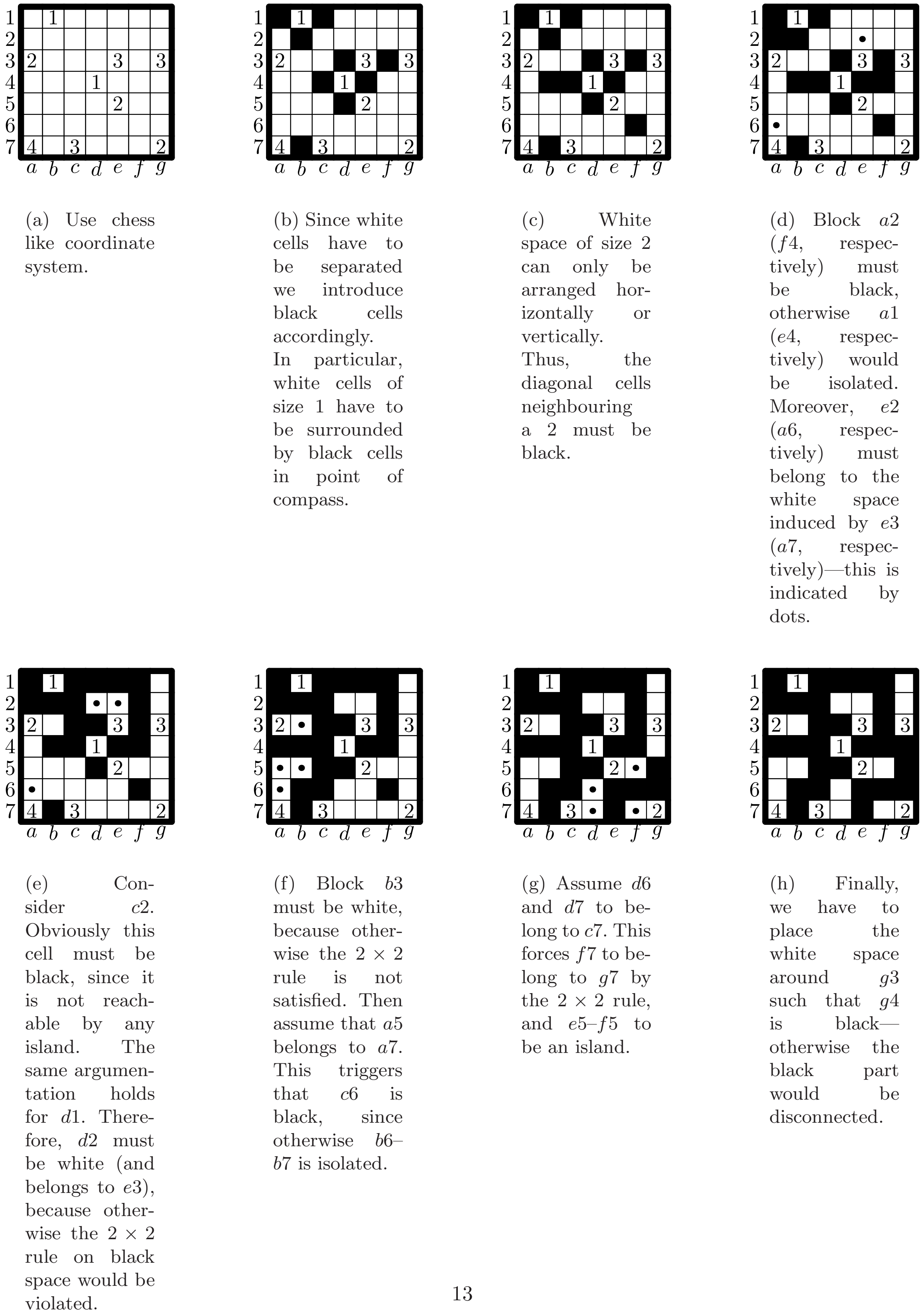
4 Conclusions

We have investigated the computational complexity of solving NURIKABE pencil puzzles and variants thereof. It turned out that all four considered variants induced by the black cell rules are intractable, i.e., are NP-complete. It is worth mentioning that the NP-completeness result for original NURIKABE pencil puzzle is optimal with respect to the involved numbers within the puzzle cells. It remains open whether a similar statement is also valid for the other presented NP-completeness results.

References

1. E. Friedman. Corral puzzles are NP-complete. Technical report, Stetson University, DeLand, FL 32723, 2002. www.stetson.edu/ efriedman/papers/corral.ps.
2. E. Friedman. Pearl puzzles are NP-complete. Technical report, Stetson University, DeLand, FL 32723, 2002. www.stetson.edu/ efriedman/papers/pearl.ps.
3. E. Friedman. Spiral galaxies puzzle are NP-complete. Technical report, Stetson University, DeLand, FL 32723, 2002. www.stetson.edu/ efriedman/papers/spiral.ps.
4. D. Lichtenstein. Planar formulae and their uses. *SIAM Journal on Computing*, 11:329–343, 1982.
5. C. H. Papadimitriou. *Computational Complexity*. Addison-Wesley, 1994.
6. T. Seta. The complexities of CROSS SUM. *IPSJ SIG Notes*, AL-84:51–58, 2002. (in Japanese).
7. N. Ueda and T. Nagao. NP-completeness results for NONOGRAM via parsimonious reductions. Technical Report TR96-0008, Department of Computer Science, Tokyo Institute of Technology, 1996.
8. L. G. Valiant. The complexity of computing the permanent. *Theoretical Computer Science*, 8(2):189–201, 1979.
9. T. Yato. On the NP-completness of the slither link puzzle. *IPSJ SIG Notes*, AL-74:25–32, 2000. (in Japanese).
10. T. Yato. Complexity and completeness of finiding another solution and its application to puzzles. Master thesis, Graduate School of Science, University of Tokyo, January 2003.

Appendix



(a) Use chess like coordinate system.

(b) Since white cells have to be separated we introduce black cells accordingly.
In particular, white cells of size 1 have to be surrounded by black cells in point of compass.

(c) White space of size 2 can only be arranged horizontally or vertically.
Thus, the diagonal cells neighbouring a 2 must be black.

(d) Block $a2$ ($f4$, respectively) must be black, otherwise $a1$ ($e4$, respectively) would be isolated. Moreover, $e2$ ($a6$, respectively) must belong to the white space induced by $e3$ ($a7$, respectively)—this is indicated by dots.

(e) Consider $c2$. Obviously this cell must be black, since it is not reachable by any island. The same argumentation holds for $d1$. Therefore, $d2$ must be white (and belongs to $e3$), because otherwise the 2×2 rule on black space would be violated.

(f) Block $b3$ must be white, because otherwise the 2×2 rule is not satisfied. Then assume that $a5$ belongs to $a7$. This triggers that $c6$ is black, since otherwise $b6$–$b7$ is isolated.

(g) Assume $d6$ and $d7$ to belong to $c7$. This forces $f7$ to belong to $g7$ by the 2×2 rule, and $e5$–$f5$ to be an island.

(h) Finally, we have to place the white space around $g3$ such that $g4$ is black—otherwise the black part would be disconnected.

Fig. 15. Step by step sample solution to the NURIKABE puzzle.