Utrecht University

Bachelor thesis

On the NP-completeness of the logic puzzle *irasuto*

Author
Shi Yi Butter

Supervisor
Dr. Benjamin Rin

Second reader
Colin Caret

7,5 ECTS
Bachelor Kunstmatige Intelligentie
Faculty of Humanities
Utrecht University
2 July 2021

Abstract

This thesis investigates the computational complexity of the Japanese pen-and-paper puzzle *irasuto*. We show that the problem is NP-complete by presenting a polynomial time reduction from Planar Circuit SAT containing only `NOR` gates to *irasuto*.

Contents

Introduction

In computer science, problems are distinguished by the class of polynomial time computable problems (P), the class of nondeterministic polynomial time computable problems (NP), and the class of problems that are beyond NP. Multiplication or sorting a list in order can be done in a reasonable amount of time and these are in P. The well known puzzles *sudoku* and *minesweeper* are known to be NP-complete [1][2]. An $n \times n$ *sudoku* grid can take a considerable amount of time to solve, however verifying if a solution is correct is done in polynomial time by a deterministic Turing machine. Additionally, the NP-completeness of the pen-and-paper puzzles *latin squares*, *number link*, *pipe link*, and *zig-zag numberlink* has been proven [3][4][5][6]. It is known that $P \subseteq NP$, but the question whether P is equivalent to NP, known as the P versus NP problem introduced by Stephen Cook, still remains [7].

Artificial intelligence (AI) is developing rapidly. Therefore, it is essential to study the complexities of algorithms when implementing them, in the interest of their efficiency. The importance of logic in AI is not to be underestimated. Many sub fields in AI, for instance, knowledge representation, planning, and problem solving, all require logic. The aim of the thesis is to show that the logic puzzle *irasuto* [8] is NP-complete. The proof of *irasuto* will hopefully give more insight into and provide a better understanding of NP-completeness since the P versus NP problem is still not solved.

The following definitions are given by Michael Sipser in *Introduction to the Theory of Computation* [9].

Definition 0.0.1. A language B is NP-complete iff

1. B is in NP, and
2. every A in NP is polynomial time reducible to B.

Definition 0.0.2. A language A is polynomial time time reducible to a language B, written $A \leq_{\mathrm{P}} B$, if a polynomial time computable function $f : \Sigma^* \longrightarrow \Sigma^*$ exists, where for every w, we have $w \in A \Longleftrightarrow f(w) \in B$.

Thus, a reduction is desired for the second step of the proof. There are various problems that are known to be NP-complete. In case of *irasuto*, we show a reduction from the planar circuit satisfiability problem (Planar Circuit SAT).

Chapter 1

Irasuto

Irasuto is a pen-and-paper puzzle named after the Japanese word for ‘illustration’. Many instances of *iratuto* of varying difficulty can be found on the Janko website (see [8]).

1.1 Rules

The goal of *irasuto* is to colour each cell [1] in the grid black or white and the rules are as follows:

- A number in a white cell or a black cell indicates how many cells of that colour can be seen from the numbered cell.
- A cell can look only in the four directions north, east, south, and west up to a cell with a different colour, a numbered cell or the edge of the grid.

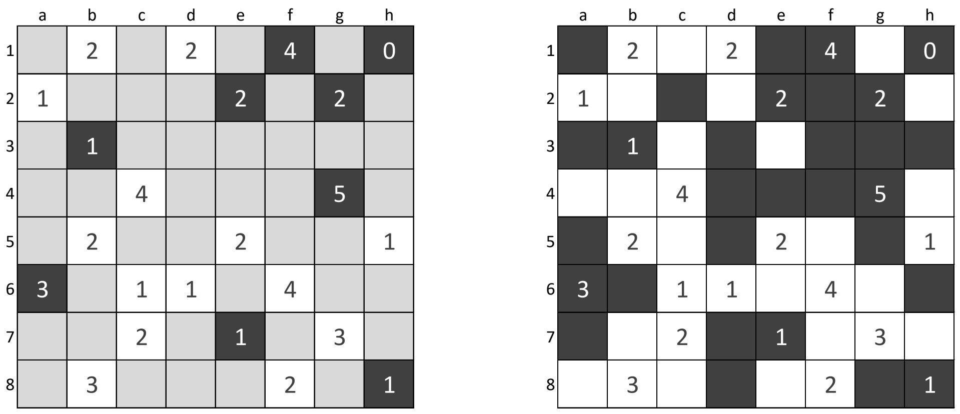

Figure 1.1: Example of an instance of *irasuto* on the left and a solution on the right

Here are some examples to demonstrate the rules. The black 0 in $h1$ in figure 1.1 has to see zero black cells, thus $g1$ and $h2$ are forced to be white. Therefore, the black 2 in $g2$ sees two white cells $g1$ and $h2$. The only cells left are $f2$ and $g3$ for the reason that a numbered cell cannot see other numbered cells, in this case $e2$ and $g5$, and a cell can only look into the cardinal directions. Thus, to satisfy the black 2 in $g2$, the cells $f2$ and

[1]The terms ‘cell’ and ‘square’ are used interchangeably but they have the same meaning. They both refer to a position on the grid e.g. $e4$.

$g3$ must be black. As a last example, $c2$ is black. Suppose $c2$ is white, then the white 1 in $a2$ sees two white cells, which is against the rules.

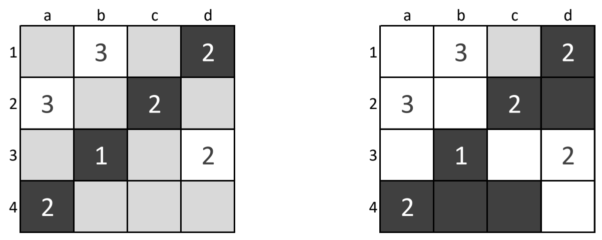

Figure 1.2: Example of an unsolvable instance

Figure 1.2 shows an unsolvable instance. The white 3 in $b1$ can only see the three cells $a1$, $b2$ and $c1$. To satisfy white 3, all these cells must be white. However, the black 2 in $d1$ forces $c1$ and $d1$ to be black. Cell $c1$ cannot be white and black at the same time. Hence, this instance is unsolvable.

1.2 Planar Circuit SAT

Planar Circuit SAT is the decision problem that tests whether there exists an assignment to variables of an formula such that a Boolean circuit C is satisfiable, i.e. an interpretation of the variables that makes the circuit output TRUE [9]. Let

$$\text{Planar Circuit SAT} = \{\langle C\rangle \mid C \text{ is a satisfiable planar Boolean circuit}\}$$

The boolean operators are AND, OR, and NOT, and they are represented as $\wedge, \vee$, and $\neg$ respectively.

1.3 Proof NP-completeness

Proof idea. To show that *irasuto* is NP-complete, we must show that it is in NP and that all NP-problems are polynomial time reducible to it. To show that every NP-problem is polynomial time reducible to *irasuto*, we show that Planar Circuit SAT is polynomial time reducible to *irasuto.* We do this by taking various features of the circuit and we convert them into corresponding features of *irasuto.*

Proof. First, we show that *irasuto* $\in$ NP. Thus, there exists a polynomial time verifier that verifies if a solution of the puzzles is correct. Let V be verifier for *irasuto.* Given a certificate c where c is a solution of the puzzle, test if the algorithm visits each numbered cell and check if the number of coloured cells of the adjacent cells of the numbered cell corresponds to the number of the numbered cell. The algorithm has to look for each cell if the cells in its row and its column obey the rules. If the test passes, *accept*; otherwise, *reject.* The number of cells is $n \times n$. Checking the adjacent row and column cells of a cell takes $2n - 1$ computations. Thus, the number of calculations is $n^2 \cdot (2n - 1) = 2n^3 - n^2$. The big O notation disregards the coefficient 2 and the term n^2 because n^2 is dominated by n^3. Therefore, the worst case running time of V is $\mathcal{O}(n^3)$, which is cubic polynomial time. Hence, *irasuto* $\in$ NP.

Now we show that Planar Circuit SAT $\leq_{\text{P}}$ *irasuto.* We do this by building gadgets. A gadget is part of a problem B instance that corresponds to a part of a problem A

instance. The following gadgets are constructed to give a reduction from Planar Circuit SAT : wire gadget, split gadget, turn gadget, crossover gadget, gadget simulating the gates of the circuit, and TRUE terminator gadget. The gadget simulating the gates of the circuit is the `NOR` gadget. The `NOR` operator is, similar to the `NAND` operator, functional complete, i.e., combining `NOR` operators can generate all Boolean operators. The `NOR` is represented as $\downarrow$ and its truth table is shown in table 1.1. Additionally with respect to the puzzle, the white cells represent TRUE and the black cells represent FALSE.

Table 1.1: The truth table of the `NOR` gate

Input		Output
A	B	$A \downarrow B$
1	1	0
1	0	0
0	1	0
0	0	1

Wire gadget

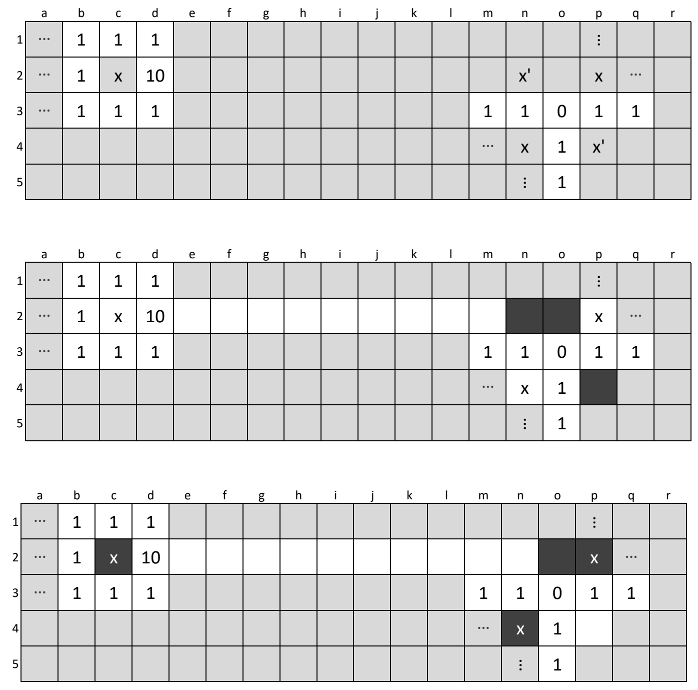

Figure 1.3: Wire gadgets

Figure 1.3 shows three wire gadgets. Wire gadgets can also be vertical. The purpose of a wire gadget is to simulate the wires of a circuit. Circuit wires copy a Boolean value and transport it from one location in the circuit to another. The wire gadgets are all straight in one direction, but in combination with turn gadgets (see below), values can be copied and redirected to any direction we need. The cells with the three dots could be anything as long as they are consistent with the gadget. We read this wire from left to right. The first wire is labeled, but there is no value assigned to the variable. The x is to label the cell and to show that all x's have the same value. The x' stands for a value that is not equal to the value of x. Since the squares have a Boolean value, the x' have the opposite value of x. To make the wires easy to read, the x' is only used in the first wire gadget. In practice, no cells are labeled when solving the puzzle. They are only used in order to explain the gadgets. Let's now look at the second wire gadget. Suppose we have an input x in $c2$ on the left hand side (LHS) and x is TRUE (thus the cell is white). The white 1s diagonal of x block off the possibility that the adjacent cells of x see two white cells. The white 10 in $c1$ sees that x is TRUE. Based on the rules, the white 10 has to see nine other white cells. Since the white 10 already sees x and the north and west cells are numbered cells, the white 10 can only look to the east. The adjacent cells $e2 - m2$ are therefore white. At this point, the white 10 is satisfied and therefore cell $n2$ is forced to be FALSE (thus $n2$ is black), otherwise the white 10 sees eleven white cells. It follows that $n4$ is TRUE because the white 1 in $n3$ sees $n2$ with value FALSE. The white 0 in $o3$ forces $o2$ to be always FALSE. Next to $n4$ is a white 1 in $o4$ which forces $p4$ to be FALSE. Then again, $p3$ sees a $p4$ with value FALSE and therefore $p2$ is TRUE. Note that $n4$ and $p2$ have the same value as input x as desired. Both these two cells can be used as output which is shown later. In this example, the x in $p2$ is assigned as the output.

Suppose input x is FALSE (this is illustrated in the third wire), then the ten cells, $e2 - n2$, next to the white 10 are TRUE. In this case, the white 1 in $n3$ sees a white cell in $n2$ which forces $n4$ to be FALSE. The white 1 in $o4$ then forces $p4$ to be TRUE which subsequently forces $p2$ to be FALSE as desired [2].

The cells $e2 - m2$ are always TRUE. The output is depending on the value of $n2$. Note that the wire can be extended or shortened by increasing or decreasing the number of the white 10.

Split gadget

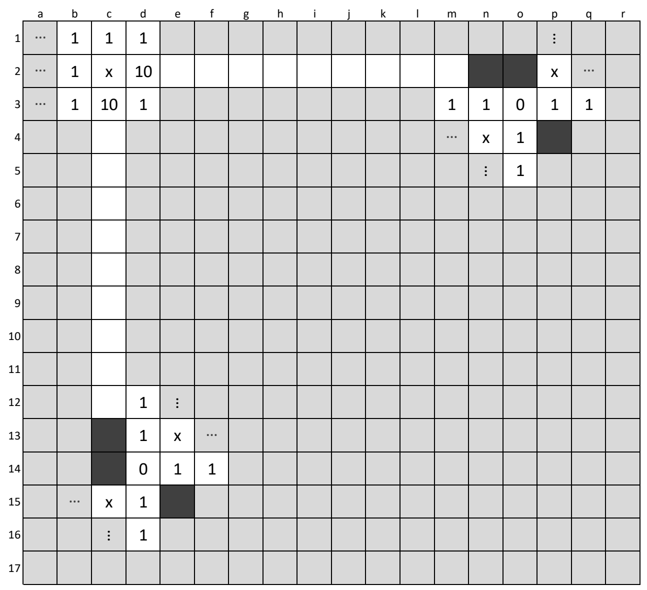

Figure 1.4: Split gadget with TRUE input

Figure 1.4 shows a split gadget. This gadget splits input x into two wires. The output of the horizontal wire is in $p2$ and the output of the vertical wire is in $c15$. The two

[2]At first sight, this gadget seems cumbersome because of the rotation of the alternating TRUE and FALSE cells in $n2$, $n4$, $p3$ and $p4$. However, we first attempted to create a simpler wire gadget but in the end the gadget did not obey the rules of the puzzle. See appendix 1.3 for further explanation.

output x's have the same value as the input x. They can be used as input for a new split gadget. A single split gadget can split an input into four output wires at most by increasing the number of $b2$ and $c1$ in this case. Observe that the parts at the end of the wires can be rotated as well as mirrored. This is illustrated in 1.5

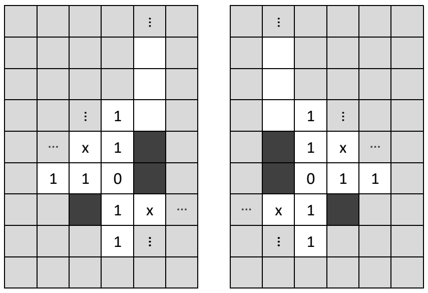

Figure 1.5: The value x that is carried can rotate to the left or to the right.

Turn gadget

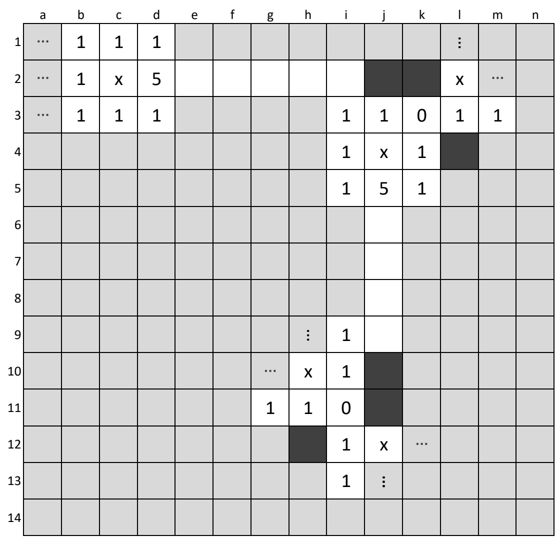

Figure 1.6: Turn gadget

Figure 1.6 shows a gadget where input x in $c2$ is TRUE and comes from the LHS. It is a horizontal wire and the x in $j4$ at the end of the horizontal wire is the input of the vertical wire. The x in $j12$ [3] at the bottom is the output of the vertical wire and has the same value as the input x of the horizontal wire.

[3]Again, $h10$ can be designated as output since $h10$ and $j12$ always have the same value as the input.

Crossover gadget

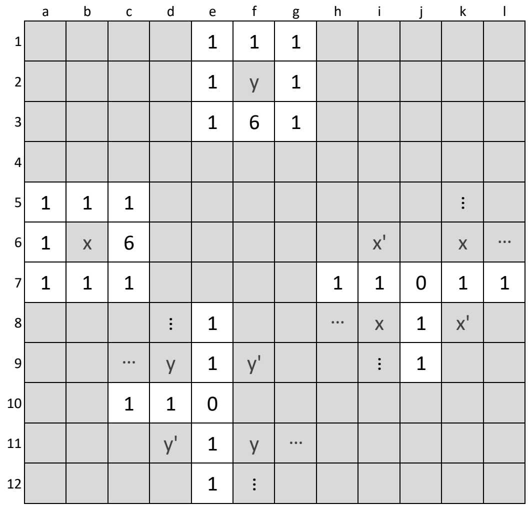

Figure 1.7: Crossover gadget

Figure 1.7 shows the crossover gadget where the cells that have the same value as the inputs and the cells that have the opposite value as the inputs are labeled. The purpose of the crossover gadget is that two wires can cross each other without interacting. In this case, the x-wire crosses the y-wire, without interacting, i.e. the value of x or y do not change. Although, the reduction is from Planar Circuit SAT, it is still useful to have available (see figure 1.21).

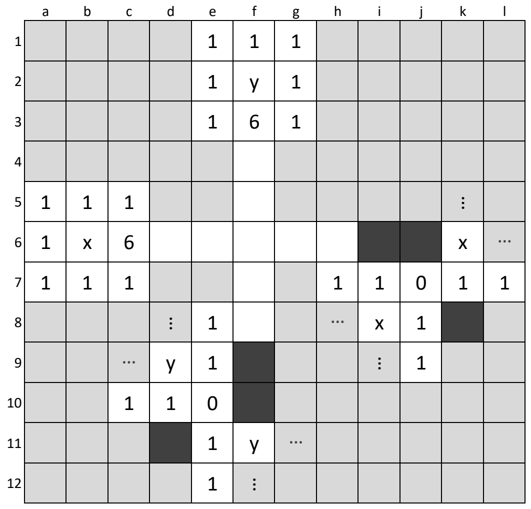

Figure 1.8: Crossover gadget in which input x and input y are both TRUE

Figure 1.8 shows a gadget where input x in $b6$ and y in $f2$ are TRUE. The x in $b6$ is the input for the horizontal wire gadget. The y in $f2$ is the input for the vertical wire gadget. The two wires cross each other and the white cross in the middle, generated by the white 6s, prevents the wires from interacting. Consequently, the output x in $k6$ and output y in $f11$ have the same value as their respective inputs.

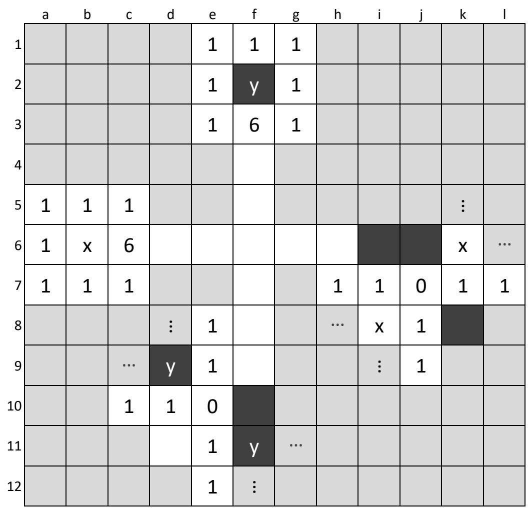

Figure 1.9: Crossover gadget in which input x is TRUE and input y is FALSE

Figure 1.9 shows the TRUE − FALSE case. A horizontal wire with input x is TRUE crosses a vertical wire with input y is FALSE. The cells of the cross in the middle are all TRUE, the same way as the cross in figure 1.8. As a result, the two wires do not interact with each other.

The FALSE − TRUE case as shown in figure 1.10, is similar to theFALSE − TRUE case.

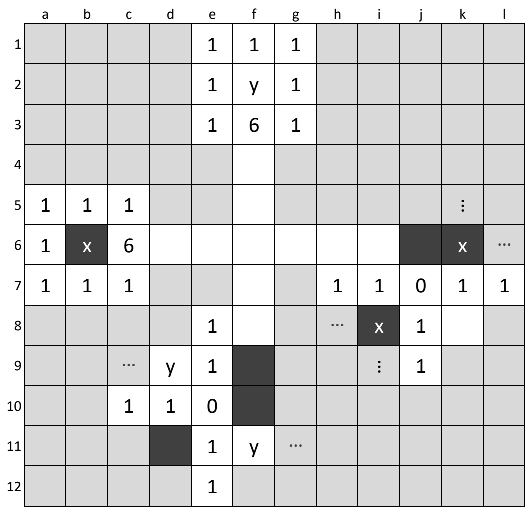

Figure 1.10: Crossover gadget in which input x is FALSE and input y is TRUE

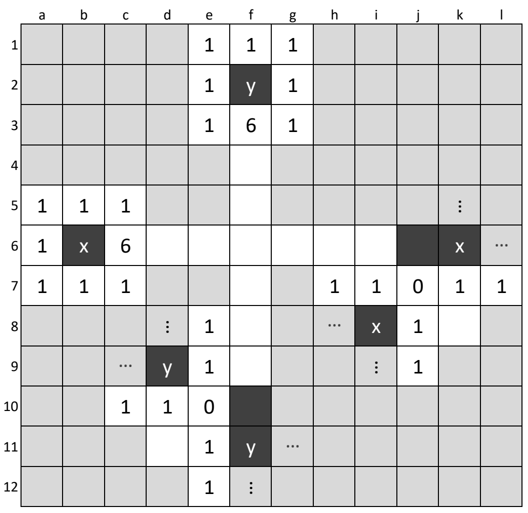

Figure 1.11: Crossover gadget in which input x and input y are both FALSE

Figure 1.11 shows the crossover gadget for the FALSE – FALSE case. A wire in a crossover gadget in which an input is FALSE is explained above in the TRUE – FALSE case. Having another wire whose input is FALSE does not affect the cross in the middle. Regardless of the input values, the cross remains white and therefore the wires can crossover safely without interfering.

NOR gadget

Squares $b2$ and $n2$ respectively represent the inputs x and y of the NOR gate, and $h6$ represents the output, see 1.12 for a NOR gate without a value assigned to the inputs. Note that the gadget can be enlarged as needed, much like the wire gadget (see figure 1.21).

We now argue that the input and output values obey the truth table for the NOR function—that is, for each of the four possible truth assignments V for the input squares, the puzzle can be solved only if the output has value $V(x)$ NOR $V(y)$.

Below, the figures 1.12, 1.14, 1.15, and 1.17 show an unsolved NOR gadget, and three out of the four input cases.

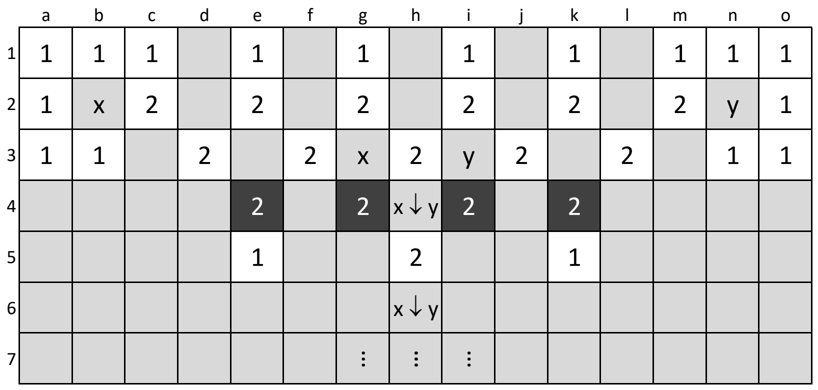

Figure 1.12: NOR gadget simulating the gates of the circuit

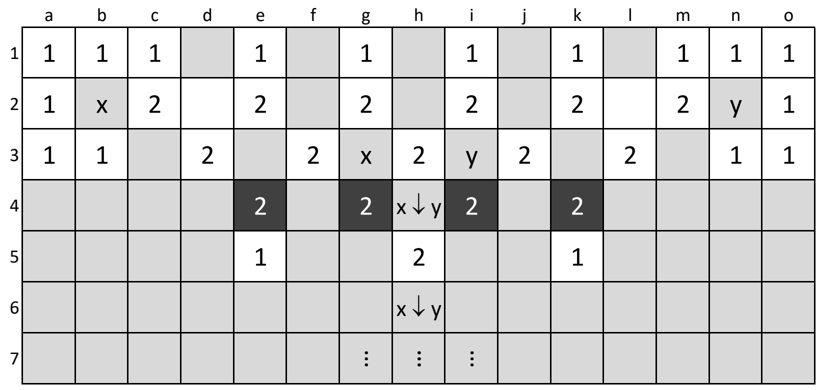

Figure 1.13: The squares $d2$ and $l2$ are in all the four cases TRUE

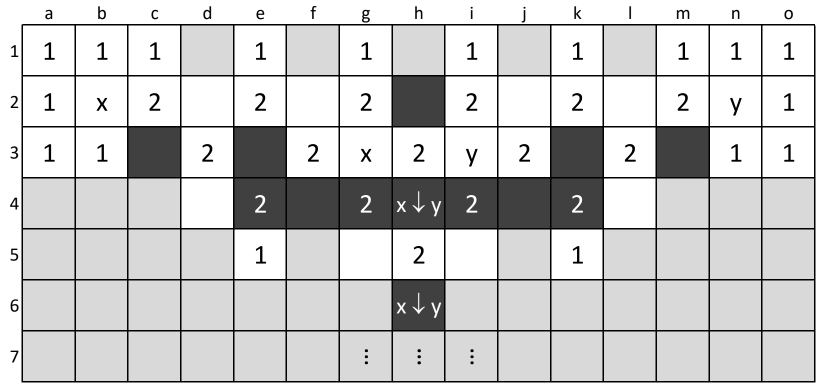

Figure 1.14: NOR in which input x and input y are both TRUE

For the case in which both inputs are true, suppose that $b2$ and $n2$ are both white squares and assume that the instance is solvable. We argue that the puzzle solution must be as depicted in figure 1.14. Let's begin from the LHS. Since x is TRUE in $b2$ and therefore the white 1 in $b3$ is satisfied, $c3$ must be FALSE. This means that $d2$ is TRUE in order that the white 2 in $c2$ sees two white cells. Observe that the white 2 in $d3$ requires that precisely one of $d4$ and $e3$ is TRUE, since $d2$ is TRUE and $c3$ is FALSE. We now argue that, in particular, $d4$ must be TRUE and $e3$ must be FALSE.

Suppose for contradiction $d4$ is FALSE and $e3$ is TRUE. We will show that this implies that $n2$ must be FALSE, contradicting our earlier assumption that $n2$ is TRUE. In what follows, figure 1.15 will be a useful reference. Observe that the white 2 in $e2$ forces $f2$ to be FALSE, and the black 2 in $e4$ forces $f4$ to be FALSE. Therefore, the white 2 in $g2$ requires that $g3$ and $h2$ are TRUE, and therefore $h4$ and $i3$ are FALSE to satisfy the white 2 in $h3$. Now the black 2s in $g4$ and $i4$ force $g5$ and $i5$ to be TRUE, making $h6$ FALSE because of $h5$. The white 2 in $i2$ now makes $j2$ TRUE. The black 2 in $i4$ already sees two black cells, thus $j4$ is TRUE. Thus $k4$ makes $k3$ and $l4$ FALSE. Now $l2$ and $m3$ must be white to satisfy the white 2 in $l3$. Hence, $m2$ makes $n2$ FALSE, as required. Thus $d4$ is TRUE and $e3$ is FALSE, as claimed.

We now define some shorthand. Let $\alpha \to \beta$ denote that the number in square α, together with known truth values of some adjacent squares, forces the value in square β to be as in figure 1.15. Thus the reasoning in the above paragraph can be abbreviated $e2 \to f2,\ e4 \to f4,\ g2 \to g3,\ g2 \to h2,\ \dots,\ m2 \to n2$.

Having shown that $d4$ is TRUE and $e3$ is FALSE, we now continue to argue that figure 1.14 depicts the unique solution for the case in which x and y are both TRUE. (In what follows, the $\rightarrow$ notation now refers to figure 1.14 rather than 1.15.)

$$\begin{array}{ccc} e2 & \rightarrow & f2 \\ e4 & \rightarrow & f4 \\ f3 & \rightarrow & g3 \\ g2 & \rightarrow & h2 \\ i2 & \rightarrow & i3 \\ i2 & \rightarrow & j2 \\ h3 & \rightarrow & h4 \\ g4 & \rightarrow & g5 \\ j3 & \rightarrow & j4 \\ j3 & \rightarrow & k3 \\ i4 & \rightarrow & i5 \\ h5 & \rightarrow & h6 \\ k2 & \rightarrow & l2 \\ k4 & \rightarrow & l4 \\ l3 & \rightarrow & m3 \end{array} \tag{1.1}$$

This establishes the claim for the TRUE – TRUE case. For the TRUE – FALSE case, suppose that $b2$ is TRUE and $n2$ is FALSE and assume the instance is solvable. We argue that the solution must be as in figure 1.15 on the next page (here the $\rightarrow$ notation refers again to figure 1.15).

$$\begin{array}{ccc} e2 & \rightarrow & f2 \\ e4 & \rightarrow & f4 \\ f3 & \rightarrow & g3 \\ g2 & \rightarrow & h2 \\ h3 & \rightarrow & h4 \\ h3 & \rightarrow & i3 \\ g4 & \rightarrow & g5 \\ i4 & \rightarrow & i5 \\ h5 & \rightarrow & h6 \\ i2 & \rightarrow & j2 \\ i4 & \rightarrow & j4 \\ j3 & \rightarrow & k3 \\ k2 & \rightarrow & l2 \\ k4 & \rightarrow & l4 \\ l3 & \rightarrow & m3 \end{array} \tag{1.2}$$

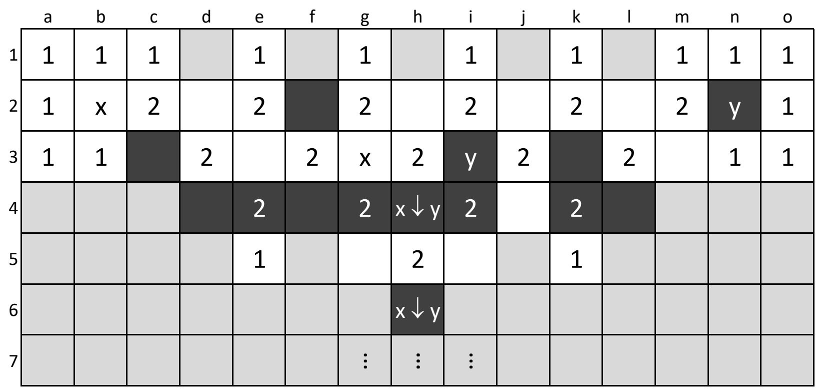

Figure 1.15: NOR gadget in which input x is TRUE and input y is FALSE

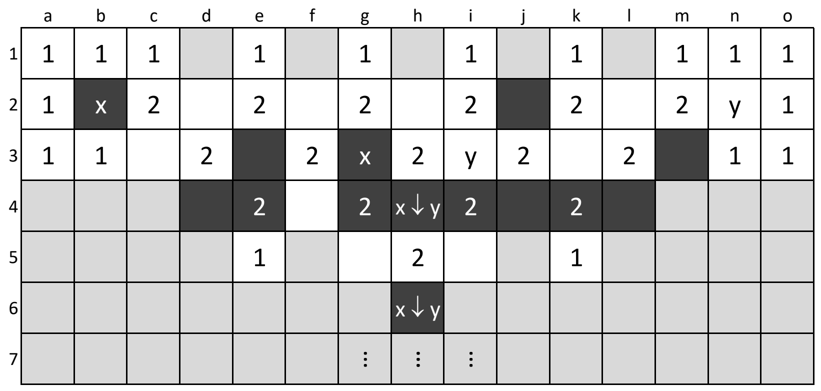

Figure 1.16: NOR gadget in which input x is FALSE and input y is TRUE

Figure 1.16 shows a NOR gadget where input x is FALSE and y is TRUE. This instance is the mirrored version of figure 1.15 and the argument is essentially the same.

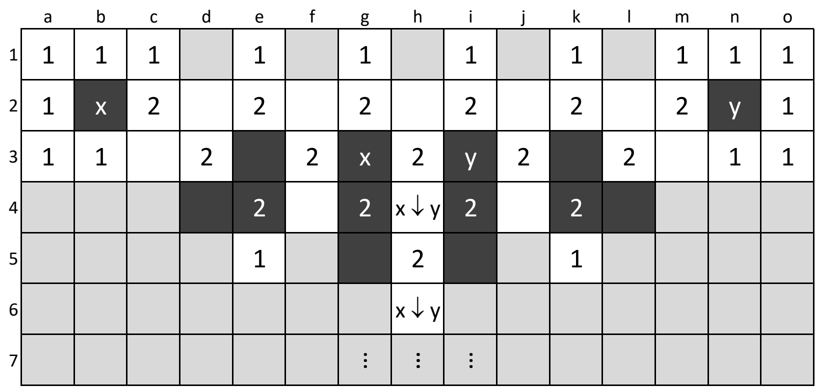

Figure 1.17: NOR gadget in which input x and input y are both FALSE

Figure 1.17 shows a NOR gadget where input x is FALSE and y is FALSE. It is easy to see that $c3$ and $d2$ are white and therefore $d4$ and $e3$ are black. This makes $f2$ and $f4$

white. Consequently, $g3$ is black because the white 2 in $f3$ already sees two white cells, and $h2$ is white to satisfy the white 2 in $g2$. Verify that the RHS is symmetrical to the LHS.

Now, $g3$ and $i3$ are black and they have the same value as the input, and we know from above that $h2$ is white. Therefore $h4$ is white because of the white 2 in $h3$. The black 2s in $g4$ and $i4$ see two white cells and one black cell, thus $g5$ and $i5$ are black. Hence the output $x \downarrow y$ in $h6$ is TRUE.

TRUE **terminator gadget**

Before the figure for the terminator gadget: The purpose of the TRUE terminator gadget is to force the simulated circuit to have TRUE final output. We need this because a correct proof requires that a circuit be satisfiable if and only if the corresponding puzzle is solvable. With the TRUE terminator gadget, the puzzle becomes unsolvable unless the final output of the simulated circuit (which is wired to the input of the TRUE terminator gadget) is TRUE.

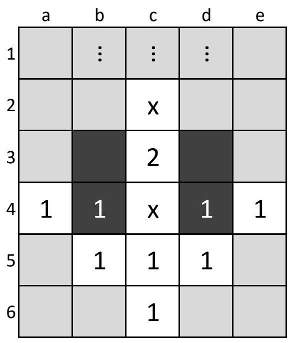

Figure 1.18: TRUE terminator gadget

Figure 1.18 shows a gadget where input x is forced to be always TRUE.

The white 1 in $c5$ forces $c4$ to be TRUE. The black 1s in $b4$ and $d4$ respectively force $b3$ and $d3$ to be FALSE. Thus $c3$ forces the input in $c2$ to be TRUE.

NOR **formula**

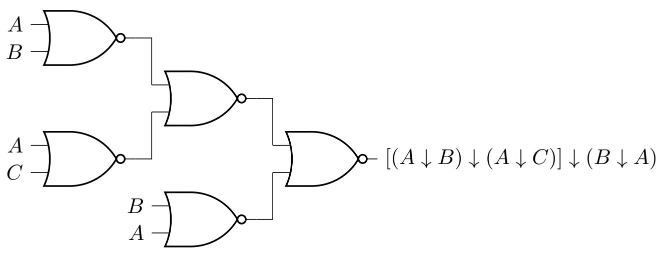
A
B
A
C
B
A
$[(A \downarrow B) \downarrow (A \downarrow C)] \downarrow (B \downarrow A)$

Figure 1.19: Planar circuit consisting entirely of NOR gates. This output is TRUE only when A is FALSE, B is TRUE and C is FALSE.

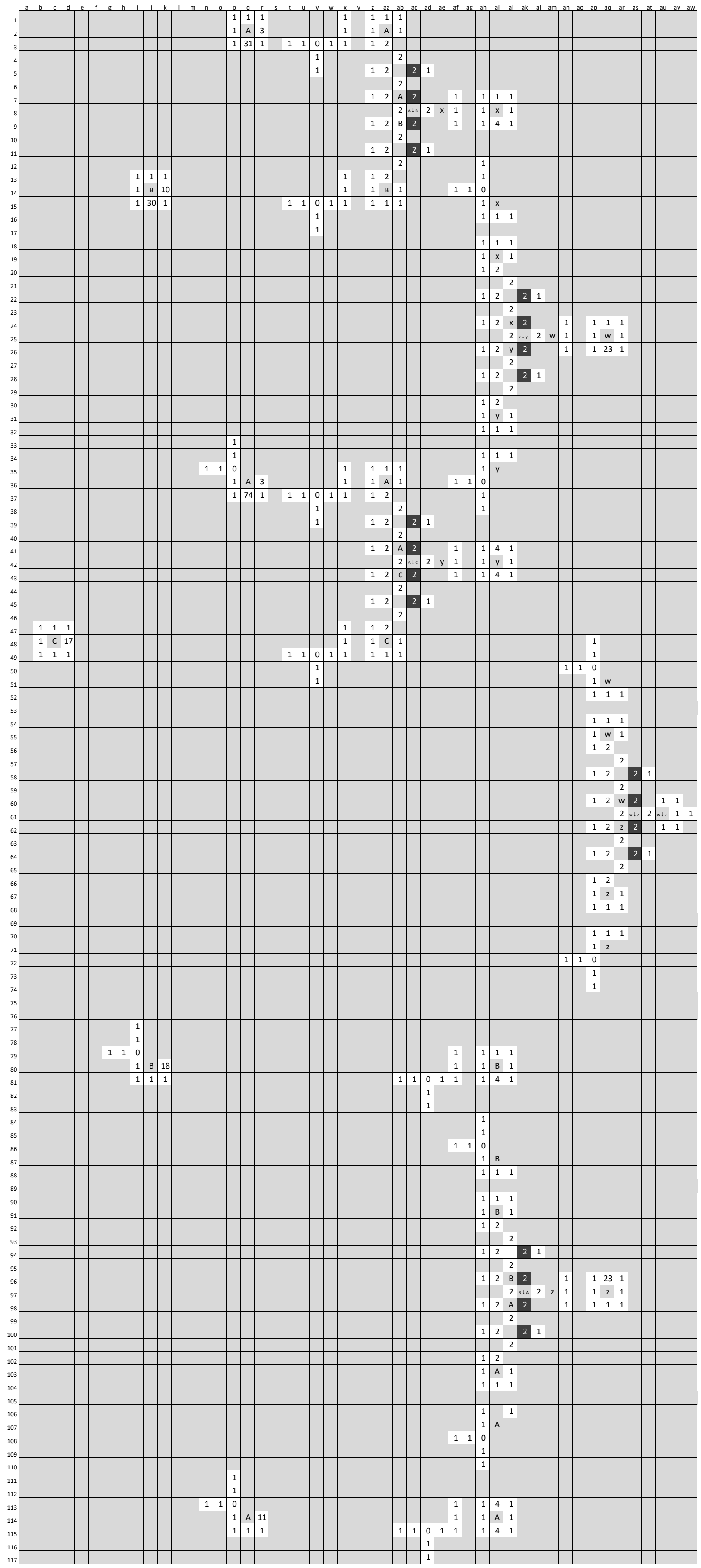

Figure 1.20: Illustration of an unsolved grid consisting of gadgets that express together $[(A \downarrow B) \downarrow (A \downarrow C)] \downarrow (B \downarrow A)$

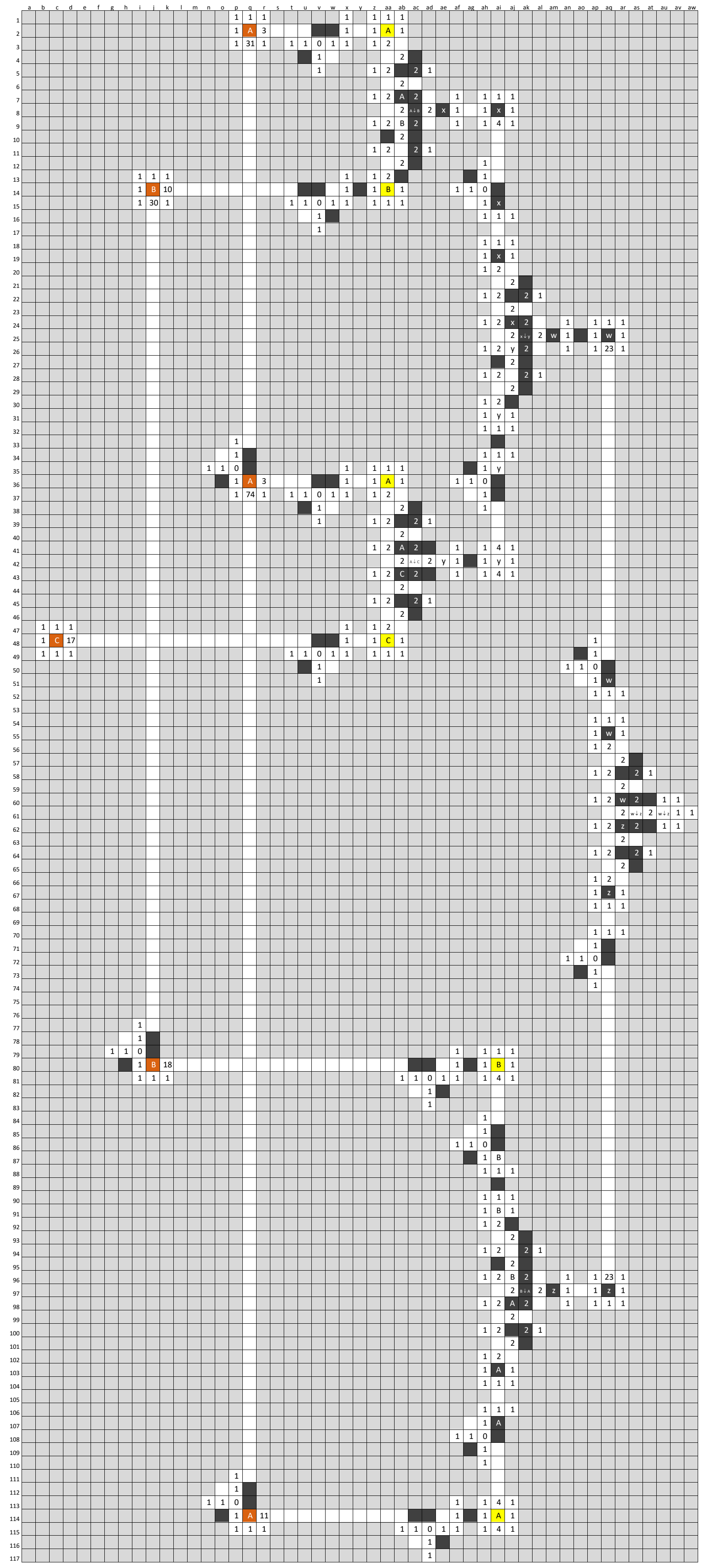

Figure 1.21: Solvable solution where A is FALSE, B is TRUE, and C are FALSE.

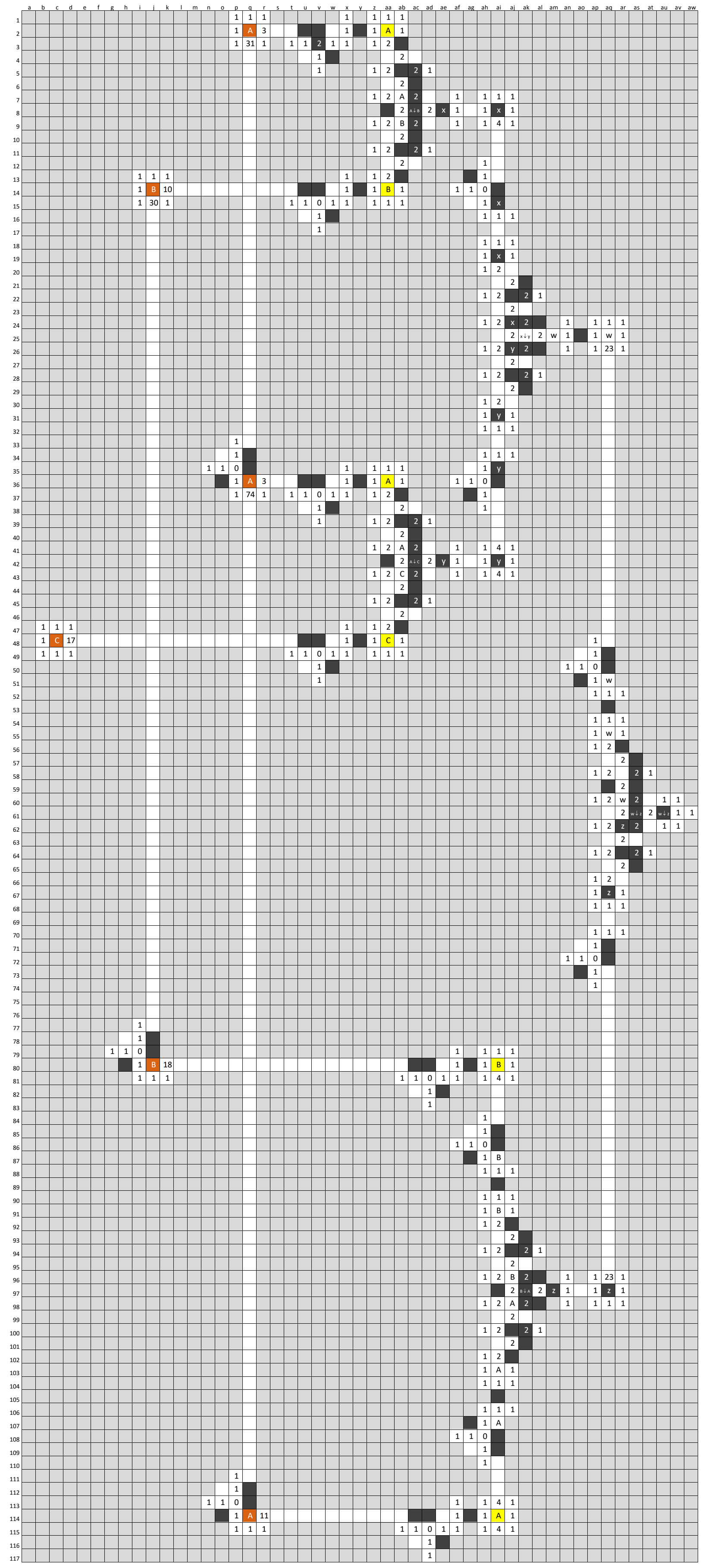

Figure 1.22: Failed solution where A, B, and C are TRUE.

Figure 1.19 is an example of a planar circuit consisting of only `NOR` gates with as input A, B and C, and as output the formula $[(A \downarrow B) \downarrow (A \downarrow C)] \downarrow (B \downarrow A)$. Figure 1.20 shows how to convert a planar circuit to a *irasuto* configuration using the gadgets as described earlier. There are no Boolean values assigned to the variables in this grid. In figure 1.21, the orange input squares A,B and C are TRUE. The output is in $au61$. The `NOR` gadgets are 90 degrees rotated. This does not affect the functioning of the gadgets. Note the similarity of the formation of the `NOR` gates of the two figures 1.19 and 1.21. This grid is to show how the gadgets are connected to correspond to the planar circuit. Figure 1.21 can be seen as a binary tree but turned a quarter clockwise. The inputs of the formula are in $q2$, $j14$ and $c48$ and they are at the same time used as input for the split gadget. They are derived from the first appearance of that variable in the formula, read from left to right. For clarity, the inputs are marked in bold: $[(\boldsymbol{A} \downarrow \boldsymbol{B}) \downarrow (A \downarrow \boldsymbol{C})] \downarrow (B \downarrow A)$. The inputs are carried to the input cells of the `NOR` gadgets through the wire and the turn gadgets, and eventually become the inputs of the `NOR` gadgets. The inputs of the `NOR` gadgets are marked yellow in $aa2$, $aa14$, $aa36$, $aa48$, $ai80$ and $aa114$, and these are the leaves of the tree. The outcome of the `NOR` gadget with inputs A and B, thus $A \downarrow B$ in $ae8$, is assigned to the variable x. The variables x, y, w and z are used for the purpose of readability. Eventually, the root is in $au61$. The root is at the same time the input for the TRUE terminator gadget which tests if this *irasuto* configuration is solvable. We can see that the final output in $au61$ is white, thus TRUE. Figure 1.22 shows that the particular assignment, A, B and C are TRUE, does not produce a solution to the puzzle.

Now, the question arises if there exists an algorithm computed by a polynomial Turing machine that converts a planar circuit to an *irasuto* configuration that is consistent if and only if the planar circuit is satisfiable. First, we place the inputs. These are coloured orange in this example. The distance (the number of cells) between an input and the `NOR` gadget can be expressed as $7k$, where k is the counter of unique appearances in the formula. Input A is the first appearance of the formula, thus the distance of A is $7 \cdot 1 = 7$. Input B is the second unique appearance in the formula, the distance of B is $7 \cdot 2 = 14$. Lastly, the distance of C is $7 \cdot 3 = 21$. Note that it is possible to make the distance larger by increasing the 7 in $7k$.

Additionally, the length of the `NOR` gadget is at least 15 cells, see the two `NOR` gadgets with inputs A and B, and A and C. These two gadgets are vertically separated from each other by nineteen cells. This does not necessarily have to be nineteen cells. The number of cells can be smaller or bigger. The separation results in extended `NOR` gadgets in the next layer to ensure there is enough space to place (other) gadgets. The `NOR` gadgets closer to the root are potentially larger. The length of a `NOR` gadget increases polynomially, when more layers are added.

By constructing multiple gadgets, we gave a reduction from Planar Circuit SAT to *irasuto*. We have shown that a planar circuit is satisfiable if and only if the *irasuto* instance is solvable. Thus, we have proved that *irasuto* is in NP and it is NP-hard. It follows that *irasuto* is NP-complete.

□

Conclusion and discussion

In this thesis, we presented a proof of the NP-completeness of *irasuto* by providing a reduction from Planar Circuit SAT . For this reduction, we built *irasuto* gadgets in a way that these correspond to parts of a planar circuit. We have shown that the gadgets can be connected such that the final output of the *irasuto* configuration corresponds to whether a Boolean circuit consisting of `NOR` operators is satisfiable or not. The proof we have provided, shows that *irasuto* is at least as hard as Planar Circuit SAT. In addition, we can establish that *irasuto* is at least as hard as any NP-complete problem.

There are two things we would have done differently in this paper if we had had more time. First, we could provide an in-depth analysis of the growth in the size of the grid when more layers are added but the growth is clearly polynomial in the size of the circuit. Second, the wire gadget is perhaps not the most efficient or elegant gadget. It is interesting to investigate if such a gadget can be constructed. If that is the case, then the wires in the split gadget, turn gadget and crossover gadget should be adjusted to conform to the new wire gadget.

Now it is known that *irasuto* is NP-complete, this problem can be used to prove as a basis for future reductions. Furthermore, other puzzles on the Janko website can be proven to be NP-complete for future research [10]. For example, *yakuso* is a pen-and-paper puzzle involving a partially completed grid as well however the puzzle has a different set of rules [11]. Therefore, instead of a reduction from a variant of SAT, it might be easier to consider other NP-complete problems. With regard to *yakuso*, we consider Subset Sum Problem to be a promising known NP-complete problem to reduce from.

Finally, reduction from variations of Planar Circuit SAT are likely possible for many pen-and-paper puzzles on the Janko website, however others may be more intuitive and/or more conveniently solved by other kinds of reductions (e.g., Hamiltonian grid graph path or Vertex-Cover) depending on the characteristics of the puzzles.

Bibliography

[1] Takayuki Yato and Takahiro Seta. “Complexity and completeness of finding another solution and its application to puzzles”. In: *IEICE transactions on fundamentals of electronics, communications and computer sciences* 86.5 (2003), pp. 1052–1060.

[2] Richard Kaye. “Minesweeper is NP-complete”. In: *The Mathematical Intelligencer* 22.2 (2000), pp. 9–15.

[3] Charles J Colbourn. “The complexity of completing partial latin squares”. In: *Discrete Applied Mathematics* 8.1 (1984), pp. 25–30.

[4] Kouichi Kotsuma and Yasuhiko Takenaga. “NP-completeness and enumeration of number link puzzle”. In: *IEICE Technical Report COMP2009–49, IEICE* (2010).

[5] Akihiro Uejima, Hiroaki Suzuki, and Atsuki Okada. “The complexity of generalized pipe link puzzles”. In: *Journal of Information Processing* 25 (2017), pp. 724–729.

[6] Aaron Adcock et al. “Zig-zag numberlink is NP-complete”. In: *Journal of Information Processing* 23.3 (2015), pp. 239–245.

[7] Stephen Cook. “The P versus NP problem”. In: *The millennium prize problems* (2006), pp. 87–104.

[8] Otto Janko. *Irasuto*. URL: `https://www.janko.at/Raetsel/Irasuto/index.htm`. (accessed: 08-05-2021).

[9] Michael Sipser. “Introduction to the Theory of Computation”. In: (2012).

[10] Otto Janko. *Rätsel, Puzzles und anderer Denksport*. URL: `https://www.janko.at/Raetsel/index.htm`. (accessed: 08-05-2021).

[11] Otto Janko. *Irasuto*. URL: `https://www.janko.at/Raetsel/Yakuso/index.htm`. (accessed: 02-07-2021).

Appendix

As stated earlier, we attempted to build a simpler gadget which is shown in figure 1.23. This gadget is functioning when the input is `TRUE`. However, when the input x is `FALSE`, the white 1 left from output x sees ten white cells. Clearly, this is against the rules. The solution for this is to pivot around the white 0 as we did in the final wire gadget 1.3.

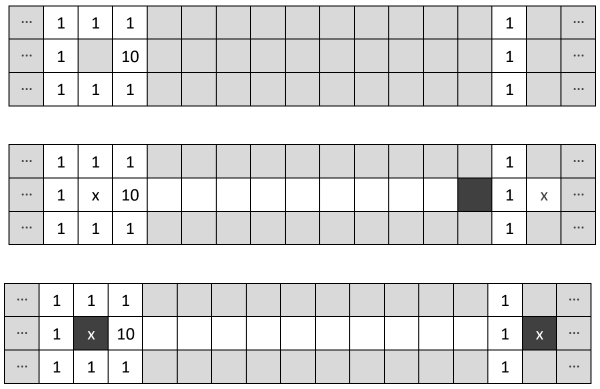

Figure 1.23: Incorrect wire gadgets