# The NP-completeness of pen and paper puzzles

Author: Mitchell Donkers (5720664)

Supervisor: Benjamin Rin

Second evaluator: Dominik Klein

Bachelor Artificial Intelligence

Utrecht University

## Abstract

Pen and paper puzzles are often NP-complete. When a problem is NP-complete, it is commonly understood that (under the assumption that P is not equal to NP) the problem is too complex for computers to compute a solution in reasonable time. In this paper we use the Hamiltonian grid graph problem and Planar NOR CircuitSAT to prove that respectively Arukone[3] and Bariasensa are NP-complete.

## Acknowledgement

I would like to express a special word of gratitude to my supervisor Benjamin Rin, whose many hours of guidance and persistent help made this thesis possible. I could not have imagined having a better advisor and mentor for my thesis.

# Contents

## 1. Introduction

Time complexity is an important subject of research in AI. This expresses the number of steps it takes to run an algorithm relative to the size of the input. One of the most interesting situations is when we look at the worst case scenario. The worst case scenario is when the given problem instance is at its most difficult to process. When the time complexity of an algorithm gets bigger then it is impractical for it to work with reasonably large data sets. Cobham's thesis states that an algorithm can be feasibly computed if it can be computed in polynomial time.

**Definition 1.** Polynomial time (P) solvable is when the time it takes for a computer to solve the problem is a polynomial function of the size of the input.

This means that the execution time in the worst case scenario is at most a polynomial function of the input. That brings up one of the more interesting time complexity classes: non deterministic polynomial time complete (NP-complete).

**Definition 2.** We say a problem is in the class Non deterministic Polynomial time (NP) when the time it takes for a computer to verify a solution to a problem instance is in polynomial time.

**Definition 3.** We say a problem is NP-hard when every problem in the class of NP can be reduced in polynomial time to said problem.

**Definition 4.** We say a problem is NP-complete when the problem is NP-hard and is in NP.

NP-completeness is interesting and important for two reasons:

- There has been no polynomial time algorithm discovered (yet) for any NP-complete problem. At the same time it has also not been proven for any NP-complete problem that it must have a super-polynomial lower bound.
- If there is a polynomial time algorithm discovered for any NP-complete problem, then all NP-complete problems are solvable in polynomial time.

Most people believe that there are no polynomial time algorithms for NP-complete problems and thus that they are not feasible to be computed, but it has not been proven. This problem is so important that the Clay Mathematics Institute (CMI) has established it as one of the seven Millennium Prize Problems [1]. The person that can solve the problem will earn 1 million dollars!

A lot of research has been done about puzzles in NP. Puzzles are something that are closely related with NP-completeness. An NP-complete problem is hard to solve and easy to check. The same holds for puzzles. The difficult part is finding a solution to a puzzle, but when have a solution we can easily see if it right or wrong. This connection is also seen in the research done. Almost all puzzles are NP-complete. Some of the more well-known NP-complete puzzles are: Sudoku, Pipe Link, Latin Square, Minesweeper and Nonograms [2][3][4][5][6]
Most researches try to prove NP-completeness via reductions.

**Definition 5**. A reduction is an algorithm to transform one problem into another problem.

The NP-complete puzzles chosen for the reduction are as numerous as the number of puzzles researched. For example recently research has been done about Rikudo puzzles [7]. This is a hexagonally shaped puzzle where cells need to be filled with consecutive numbers. The people who worked on the Rikudo research discovered a reduction from the Hamiltonian circuit problem in a hexagonal grid graph. This is a related problem to the Hamiltonian path one we will use for my proof. Here instead of a path there exists a cycle in the graph where each node is reached exactly once. Another example is the research about Light-up [8]. Light-up is a puzzle where light bulbs are placed to illuminate all the white squares. Here the researchers found that the flexibility of wire in a Boolean circuit could help prove the Light-up puzzle. Below we will explain more about Boolean circuits as we will also use them for my proof. Both researches succeeded in proving that their puzzle is NP-complete.

## 1.1 Thesis

For this paper we will look at two pen and paper puzzles, namely Arukone[3] and Bariasensa, and will show that they are NP-complete. In this paper for each of these puzzles we will answer the question is the solvability of this puzzle an NP-complete problem. We will show that it is by finding for each of the two puzzles a reduction from a known NP-complete problem.

## 1.2 Research method

In my paper we want to try to prove the NP-completeness of two logic puzzles. We will do this in two steps. First we will try to show that the puzzle is in NP. We will show that given a solution we can check every step in at most polynomial time. Next we will try to show that the puzzle is NP-hard. NP-hard means that every problem in NP can be reduced to the problem (in this case the puzzle) in polynomial time. We will primarily use the Hamiltonian path on a grid graph problem and the Boolean circuit problem for already proven NP-complete problems. From these problems we will make a reduction to the logical puzzles we are trying to prove. This will be done with proof by construction. We will build some gadgets that represent a translation from a piece of the known NP-complete problems to a part of the problem puzzle. This must be done for all possible pieces that can exist in the problem so that all possible instances of the problem are covered. All gadgets combined can give a full translation from the known NP-complete problem to the problem puzzle. We will show that any given instance of the known NP-complete problem has a solution if and only if its translation has a solution.

## 2. Proven NP-complete problems

### 2.1 Hamiltonian path on a grid graph

The Hamiltonian path on a grid graph problem is a graph problem. A graph is a collection of vertices (or nodes) that may be connected by edges (or lines). The edges can either be directed (one-way) or undirected (both ways). Because it is on a grid the vertices are on the corners of the grid and the edges are following the lines of the grid.

**Definition 6**. A two dimensional grid graph is an m x n graph which drawing forms a regular tiling.

The Hamiltonian path problem is defined as follows: a path in an undirected or directed graph (on a grid) that visits each vertex exactly once. For the Hamiltonian path on a grid graph problem the question whether there exists a Hamiltonian path has been proven to be NP-complete.

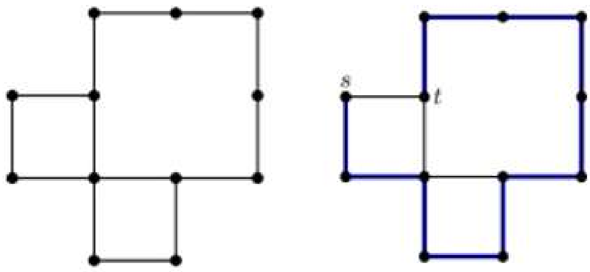


*Figure 1 A graph and the hamiltonian path problem [12]*

### 2.2 Planar NOR CircuitSAT

The Planar NOR CircuitSAT problem is subtype of Planar CircuitSAT. The Planar CircuitSAT problem is the problem if a Planar Boolean circuit, a mathematical model with digital logic circuits that represent logical formulas, can be satisfied. Here planar means that the edges of the model only intersect at the endpoints. A SAT formula (or CircuitSAT) can be *satisfied* if there is a way to make the formula true (by given logical values to the variables). Because we are talking about Planar NOR CircuitSAT all the operator-gates in the circuit are NOR-gates. So Planar NOR CircuitSAT is the problem of if a non-intersecting Boolean circuit consisting of only NOR-gates can be made true.

## 3. Arukone$^3$

Arukone$^3$ is an n x m pen and paper puzzle on a grid. It is the successor of Arukone and Arukone$^2$. Both of those are already proven to be NP-complete. Arukone$^3$ is a variant of Arukone containing just one additional rule (the 2x2 rule—see below), but this rule has large effects on the nature of the puzzle. The goal is to connect every pair of identical numbers with a line. The line can run either horizontally or vertically through the white squares, must never cover a 2x2 area and can't cross itself. The puzzle is complete when every number is connected to another via a line and all the squares of the grid are filled.

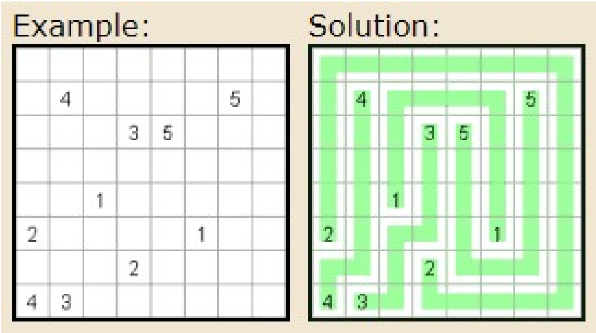


*Figure 2 Arukone$^3$ example puzzle with solution [9]*

More formally, we have the following definitions.

**Definition 7**. An **Arukone$^3$ puzzle grid** is a rectangular grid of $m \times n$ squares in $m$ rows and $n$ columns. We use coordinates to identify squares as in matrices—e.g., $(1,2)$ is the square to the right of the top-left square. If a square $A$ contains a number $n$, we write $N(A) = n$.

**Definition 8**. Square $(w, x)$ is **adjacent** to another square $(y, z)$ if and only if either $w = y$ and $|x - z| = 1$, or $|w - y| = 1$ and $x = z$.

**Definition 9**. A **unique pair** is a pair of squares $A$ and $B$ for which $N(A) = N(B) \neq N(C)$ for all other squares $C$.

**Definition 10**. A **line** is an ordered tuple $(A_1, \ldots, A_n)$ of squares such that square $A_1$ is adjacent to square $A_2$, square $A_{n-1}$ is adjacent to square $A_n$, and each square $A_i$ for $1 < i < n$ is adjacent to $A_{i-1}$ and $A_{i+1}$.

**Definition 11.** A **solution** is a set of lines such that: (1) each square in the grid is in exactly one line, (2) for all lines $(A_1, \ldots, A_n)$, we have $N(A_1) = N(A_n)$, (3) there is no set $\{(a,b), (a+1,b), (a,b+1), (a+1,b+1)\}$ of squares that are all in the same line.

**Definition 12.** Given a solution $S$ and a line $(A_1, \ldots, A_n) \in S$, two squares $X$ and $Y$ are **connected by** $(A_1, \ldots, A_n)$ when $X = A_1$ and $Y = A_n$.

## 3.1 Proof idea

First we show that Arukone$^3$ is in NP. I will show that given a solution we can check every step in at most (worst-case O) polynomial time . Next I will show that Arukone$^3$ is NP-hard. From Hamiltonian path problem I will make a reduction to Arukone$^3$. I will build some gadgets that represent a translation from a piece of the Hamiltonian path on a grid problem to a part of Arukone$^3$. All gadgets combined can give a full reduction from the Hamiltonian path on a grid problem to Arukone$^3$. Finally we put a unique number combination on the square representing the start node and the square representing the end node. If the Hamiltonian path on a grid problem has a solution then Arukone$^3$

has one and if there isn't a solution to the Hamiltonian path on a grid problem then Arukone[3] also doesn't have a solution.

### 3.2 Proof

**Theorem 1**. *Arukone[3] is np-complete for k pairs in an m x n grid*

First we show that Arukone[3] is in NP. We do this by showing that a solution of Arukone[3] is verifiable in polynomial time. To check if a solution is correct we have to check (1) if every line is between the same numbers, (2) there are no 2x2 regions covered by a single line and (3) every square is filled. The worst case scenario to check is when all squares are filled with numbers. For an m x m puzzle there are $m^2$ number of squares. Let $n = m^2$. To check condition 1 we have to compare the start and end point of every line. Because every line has at least a starting point and an end point that cannot be the same square there are at most $0.5m^2 < n$ number of lines to check. The check of each line takes no worse than linear time since the longest possible line has length n. So condition 1 can be checked in polynomial time. For condition 2 we need to check every 2x2 area individually if they fulfill the condition. There are $(m-1)^2 < n$ many 2x2 areas in the grid and each check takes constant time to do. So to check the 2x2 condition takes polynomial time. Lastly to check condition 3 we have to verify all $m^2 = n$ squares. The each check takes constant time so the third condition can also be done in polynomial time. Once all the conditions are checked, the solution is verified. Because all conditions can be checked in polynomial time, the entire solution can be verified in polynomial time. So Arukone[3] is in NP.

Next we show that Arukone[3] is NP-hard. We do this by making a reduction from the Hamiltonian path in a grid graph problem to Arukone[3]. For the reduction we take every possible combination of nodes and edges and create a gadget for each combination. The gadgets have lines corresponding to the directions of the edges.

The lines are shaped by what we call *Pillars*.

**Definition 12**. An ***Inner pillars*** *is a 2x2 areas in the puzzle closest to the central square.*

**Definition 13**. An ***Outer pillars*** *is a 2x2 areas in the puzzle furthest from the central square.*

For the reader these areas are colored red and yellow in the gadget. We also use the color blue to show the central square and (possible) the Hamiltonian path solution. The purple parts need to be coordinated between gadgets and the black parts can only be filled a gadget is connected to another gadget (this will be made more clear in the gadget connection part). Because lines consists of pillars the direction of the lines is tilted 45 degrees from the direction of the edge.

**Definition 14**. *An* ***upward edge*** *in an Arukone[3] is the path around the north-east inner and outer pillar*

**Definition 15**. *An* ***rightward edge*** *in an Arukone[3] is the path around the south-east inner and outer pillar*

**Definition 16**. *An* ***downward edge*** *in an Arukone[3] is the path around the south-west inner and outer pillar*

**Definition 17**. *An* ***upward edge*** *in an Arukone[3] is the path around the north-east inner and outer pillar*

The way the reduction works is by fulfilling the rules of a Hamiltonian path for each gadget.

To enforce that every node can only be visited once we created central squares. A central square can only be filled when it is the start/end of a Hamiltonian path or when it gets passed by a line.

Another rule that needs to be enforce is that all squares need to be filled. When filling the lines as intended then the left over spaces, the ones that are not in the Hamiltonian path solution, will also need to be filled somehow. This is what the pillars are for.

**Definition 18**. *A pair is* ***directly*** *connected when it is connected with a line of length 2.*

**Definition 19**. *A pair is* ***indirectly*** *connected when it is connected with a line of length 3 or higher.*

The yellow pair can be connected directly when the Hamiltonian path passes around it (Figure 3 left), but can be connected around the red pair with a line of length 10 when the Hamiltonian path does not go through that direction (figure 3 right).

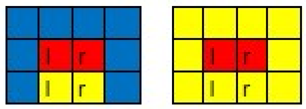


*Figure 3 A path around an directly connected pillar (left) and an indirectly connected pillar without surounding path (right)*

The possible line directions for the solution are created by putting up walls. These walls are made with lines of 1 x 2 (two of the same number next to each other) in the directions where the node does not have edges connected. This makes sure that the line of the solution can pass through in the same way the path goes through the nodes in the Hamiltonian path on a grid graph problem.

Below we will show, for each of nine gadgets, that they can only be filled as intended. Because of the way the gadgets are constructed if we make a line in any other way then intended we will reach a dead end when trying to fill all the squares.

For the sake of simplicity we will make all numbers unique. Rather than displaying these numbers explicitly, we represent this by replacing each number with either an ‘l’ (left), ‘r’ (right), ‘u’ (up) or ‘d’ (down). Every ‘l’ needs be connected with an ‘r’ directly to the right of it and every ‘u’ with the ‘d’ directly below it. We will also only show the parts that can be fulfilled and do not show part of the edges that do not reach a node. Those parts of the gadgets can only be filled in if they are connected to another gadget in that direction, so we leave that outside the example (colored black). This will be explained in more detail in the part gadget connection

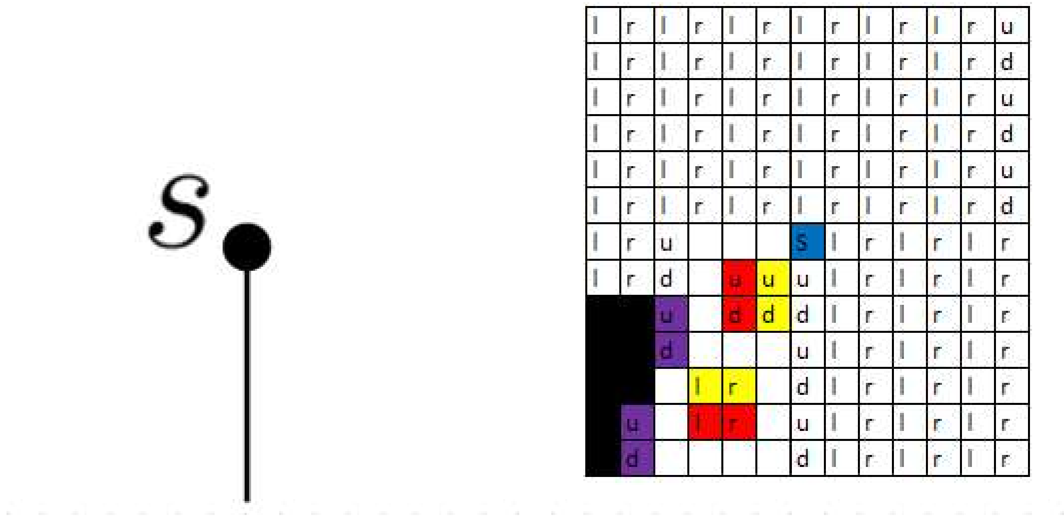


*Figure 4a A single downward edge (as the start of a Hamiltonian path) with corresponding gadget [12]*

First of all note that when a graph has a note with a single edge, then that node needs to be either the beginning or the end of a Hamiltonian path (if it exists). So there does not exist a single edge gadget that is not the start or end of the Hamiltonian path.

Because the gadget consists of single path without the possibility to diverge and the fact that the central square is either the start or the end of a line we know that all the white squares must be part of the line. In figure 4b the path is colored it blue and filled in with *a's* to show that it is a single line.

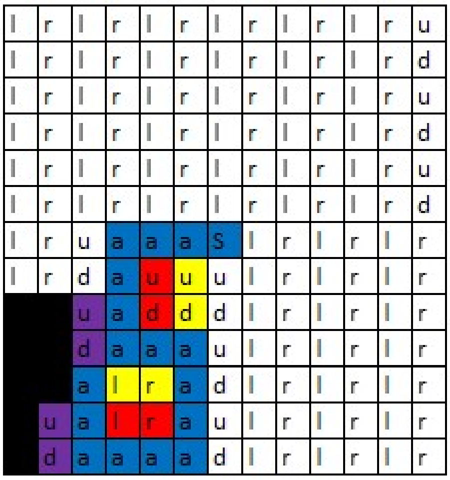

*Figure 4b A filled in single edge gadget*

When there is no hamiltonian path in a graph a single edge gadget can never be fulfilled. For a solvable puzzle all squares need to have the possibility to be filled. The central square can never be filled (because it either needs to be the start or the end of a path). So there is no situation where this can be fully filled when there is no hamiltonian path.

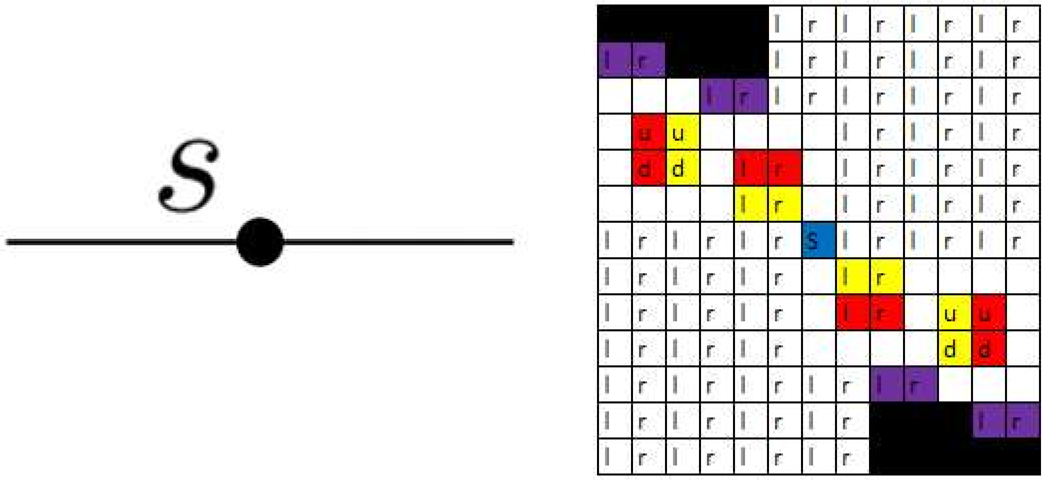


*Figure 5a A double horizontal edge (as start the of a Hamiltonian path) with corresponding gadget [12]*

When the central square is the starting (or end) point of the Hamiltonian path we follow one edge (e.g. leftward edge in figure 5b) till the end of the gadget because there is no way to deviate. The square directly adjacent to the central square in the opposite direction (e.g. square below the central square in figure 5b) can only be filled by the adjacent inner pillar. The only way to do that is by connecting the yellow part indirectly around the red part. Now the white square adjacent to the line of the inner pillar and the side of the outer pillar can only be filled by indirectly connecting the yellow part of the outer pillar. So every white square is filled as intended. The other directions are symmetric and we leave that for the readers see for themselves. In figure 5b the path is colored it blue and filled in with *a's* to show that it is a single line while the line of the inner pillar and outer pillar are filled with *b's* and *c's* respectively and are colored yellow.

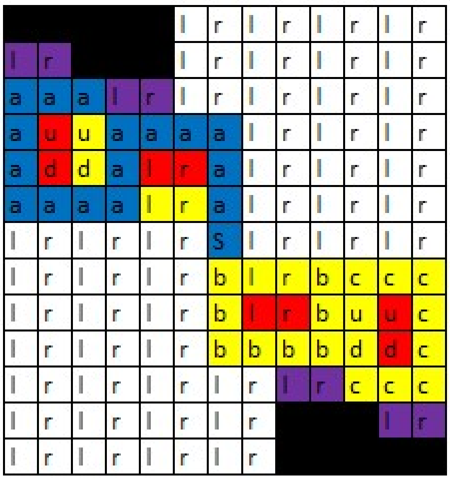

*Figure 5b A filled in double horizontal edge (as start the of a Hamiltonian path) gadget*

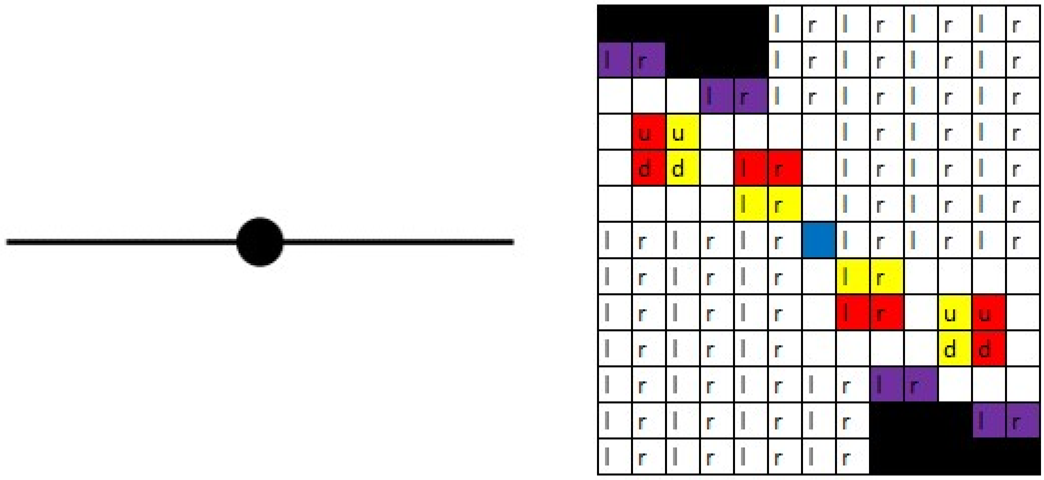

*Figure 6a A double horizontal edge with corresponding gadget [12]*

When the central square is part of a line we know that from the central square both directions need to be filled with the same path. This is because the central square must be filled and it cannot be the start or end of a line. We follow both edges till the end of the gadget and, because there is no place to deviate, we know that, that the start and end of the line are not in the gadget. So the entire gadget is filled with the Hamiltonian path line. In figure 6b the path is colored it blue and filled in with *a's* to show that it is a single line

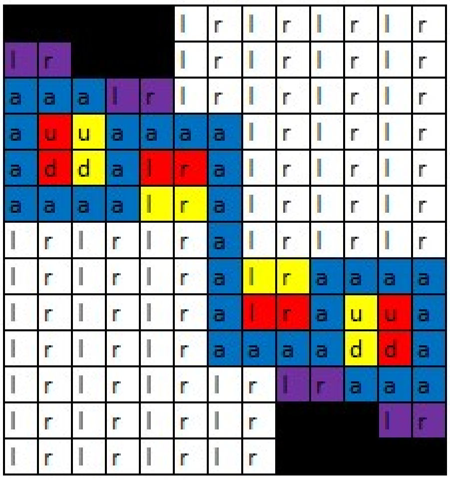

*Figure 6b A filled in double horizontal edge gadget*

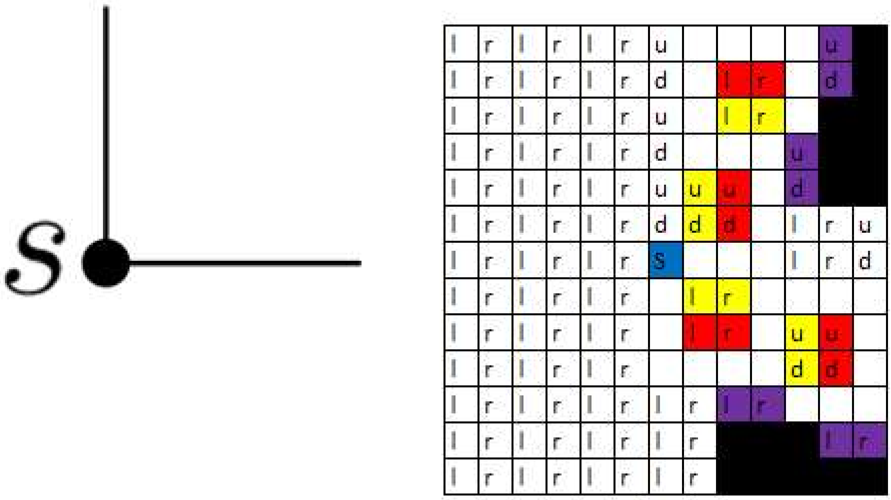


*Figure 7a A double upward and rightward corner edge (as start the of a Hamiltonian path) with corresponding gadget [12]*

When the central square is the starting (or end) point of the Hamiltonian path. If this is the case we follow one edge (e.g. upward edge in figure 7b) till the point where the path splits up (figure 7b left image). Next we see that the white square directly adjacent to the central square at a 90 degree angle (e.g. square below the central square in figure 7b) can only be filled by the adjacent inner pillar. The only way to do that is by connecting the yellow part indirectly around the red part. Now the choice that we saw for the Hamiltonian path line no longer exists (figure 7b middle image). We fill again the rest of the direction with the line  because there is no further choice. Finally the white square adjacent to the yellow line of the inner pillar can only be filled by indirectly connecting the yellow part of the outer pillar. So every white square is filled as intended. In figure 7b the path is colored it blue and filled in with *a's* to show that it is a single line while the line of the inner pillar and outer pillar are filled with *b's* and *c's* respectively and are colored yellow.

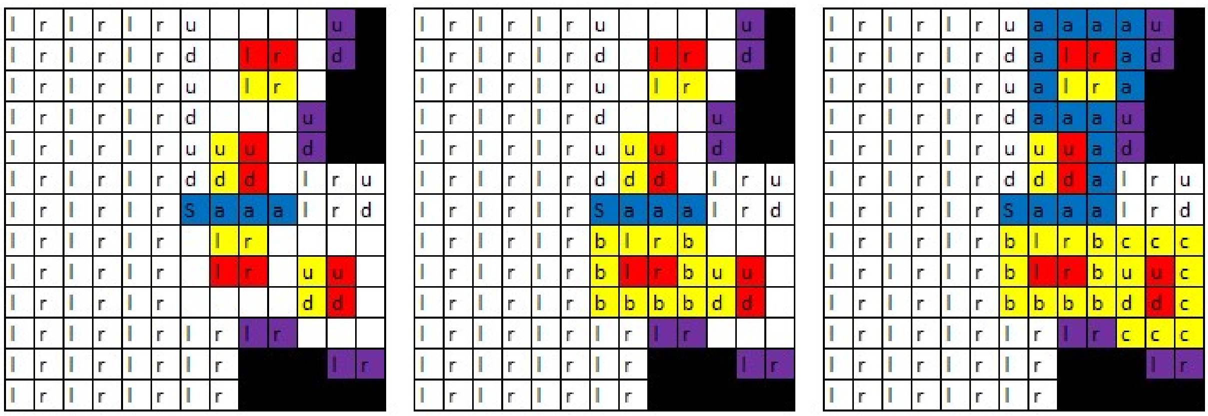

*Figure 7b The steps for a filled in double upward and rightward corner edge (as start the of a Hamiltonian path)*

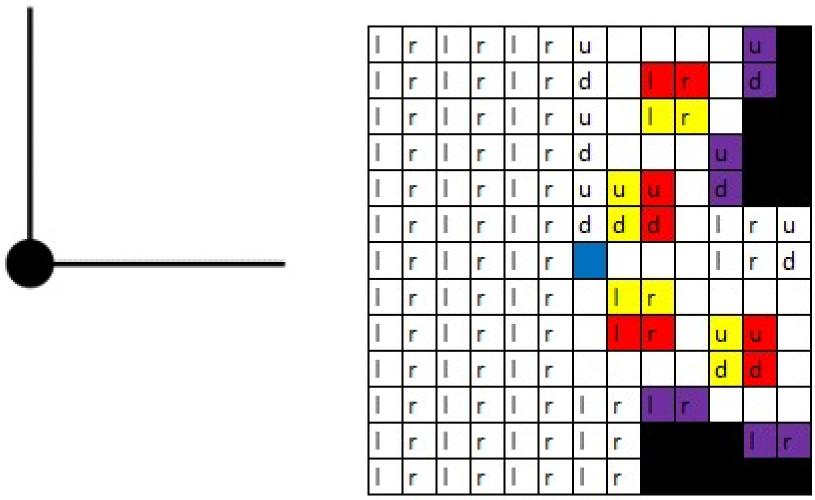

*Figure 8a A double upward and rightward corner edge with corresponding gadget [12]*

When the central square is part of a line we know that from the central square both directions need to be filled with the same path. We follow both edges till the point where the path splits up. Note that both line directions are now next to each other and block off one of the line direction choices for each other (see figure 8b left). Because both those squares do not have any numbers in them the path cannot end there. We continue to fill in both edges till the end of the gadget and, because there is no place to deviate, we know that, that the start and end of the line are not in the gadget. So the entire gadget is filled with the Hamiltonian path line. In figure 8b the path is colored it blue and filled in with *a's* to show that it is a single line

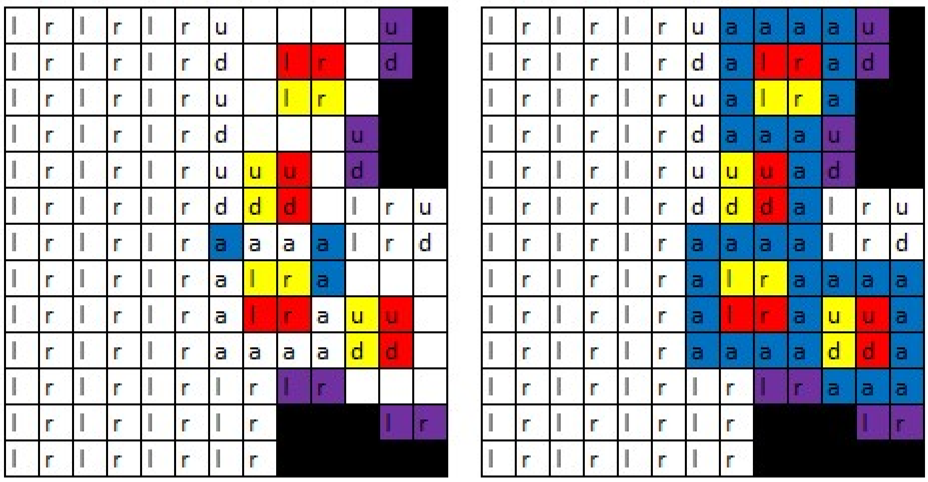

*Figure 8b The steps for a filled in double upward and rightward corner edge*

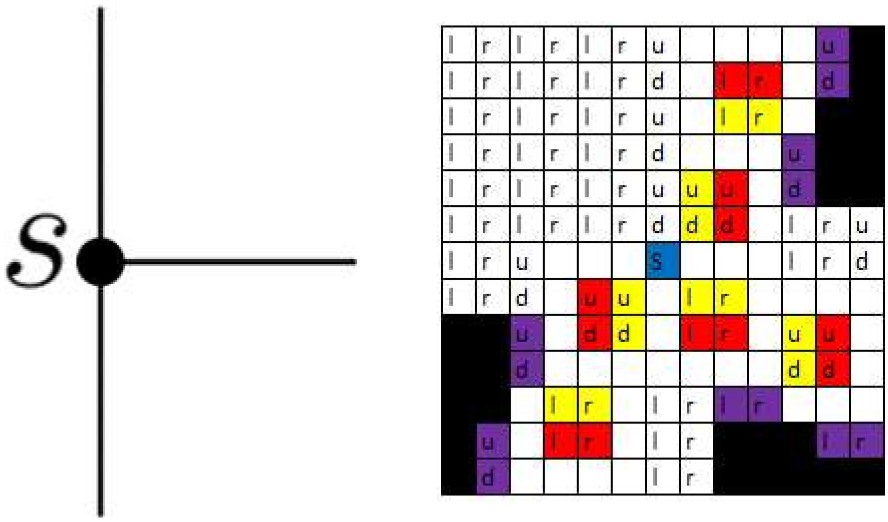


*Figure 9a A triple upward, rightward and downward edge with corresponding gadget (as start the of a Hamiltonian path) [12]*

For this gadget we need to differentiate between two different, namely when the leftover (not in the Hamiltonian path) edges form a corner (e.g. if the upward edge is in a Hamiltonian path in figure 9a) and when the edges form a straight line (e.g. if the rightward edge is in a Hamiltonian path in figure 9a).

In the first case we go to the right from the central square (to start filling in the upward edge for a Hamiltonian path) till we reach a point where we have to make a choice. Next we see that the white squares directly adjacent to the central square can only be filled by their adjacent inner pillars. The only way to do that is by connecting the yellow part indirectly around the red part. Now the choice that we saw for the Hamiltonian path line no longer exists (figure 9b middle image). We fill again the rest of the direction with the line because there is no further choice. The rest of the white squares can only be filled by indirectly connecting the outer pillars. Now the entire gadget is filled in as intended. In figure 9b the path is colored it blue and filled in with *a's* to show that it is a single line. Further we use the letters '*b*', '*c*', '*e*' and '*f*' (Note that the letter '*d*' is already used) to represent the lines for the different pillars that are colored yellow.

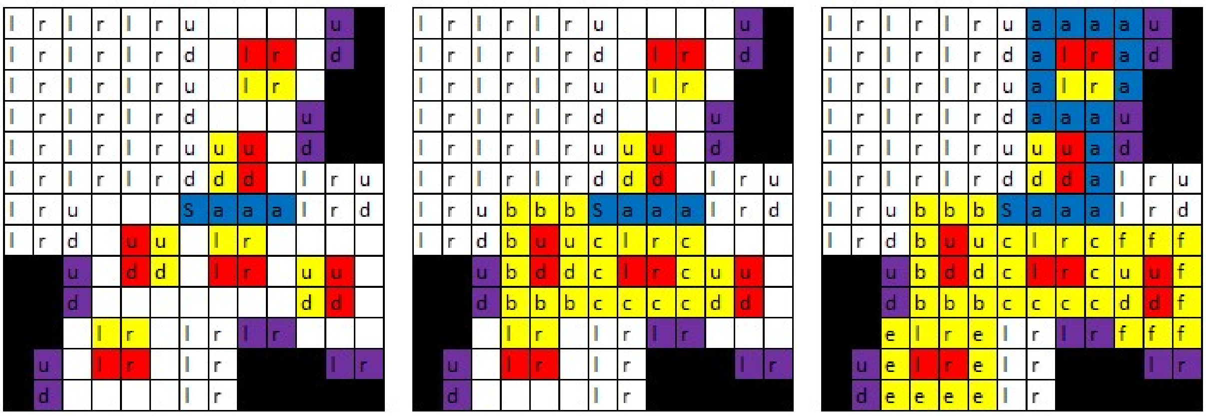

*Figure 9b The steps for case 1 and a filled in triple upward, rightward and downward edge (as start the of a Hamiltonian path)*

In the second case we go to the bottom from the central square (to start filling in the rightward edge for a Hamiltonian path) till we reach a point where we have to make a choice. This is really similar to the first case and we will only show an image with the steps (see figure 9c) so that reader can see it for themselves.

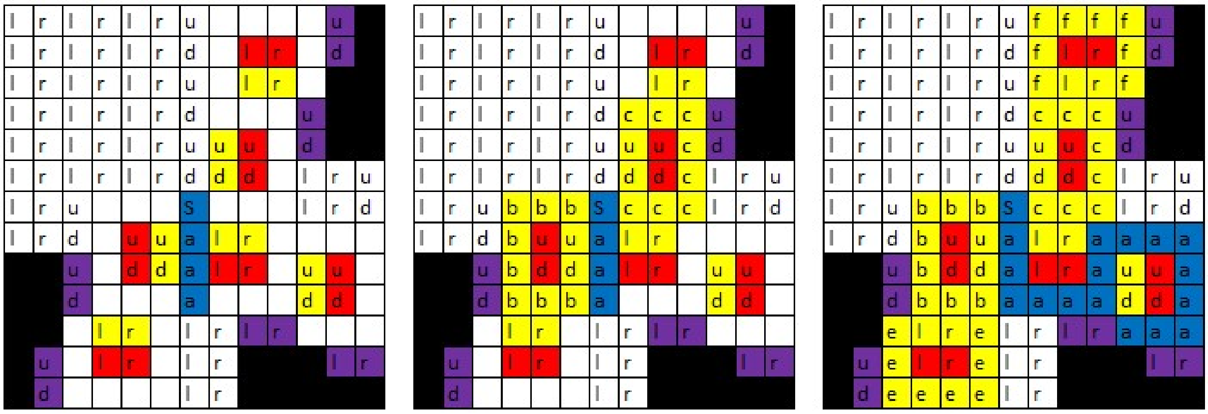

*Figure 9c The steps for case 2 and a filled in triple upward, rightward and downward edge (as start the of a Hamiltonian path)*

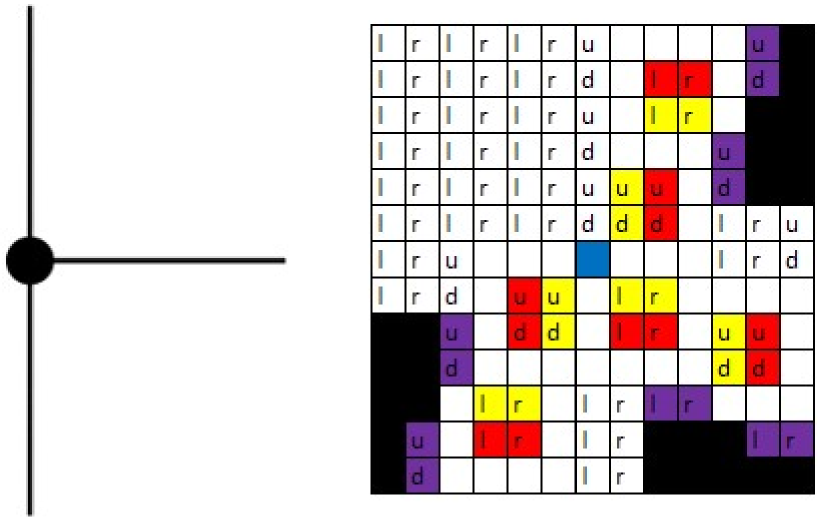

*Figure 10a A triple upward, rightward and downward edge with corresponding gadget [12]*

For this gadget we need to differentiate between two different, namely when line edges form a corner (e.g. if the upward edge and the rightward edge are in a Hamiltonian path in figure 10a) and when the line edges form a straight line (e.g. if the upward and downward edge are in a Hamiltonian path in figure 10a). When the central square is part of a line we know that from the central square exactly two directions need to be filled with the same line.

In the first case we fill in the squares going right (to fill the upward edge) and down (to fill the rightward edge) for both edges till the point where the path splits up (see figure 10b left). Next we see that the only white square directly adjacent to the central square can only be filled by its adjacent inner pillar when it is connected indirectly. This eliminates the choice for the rightward edge. We continue to fill that edge in till point where there is again a choice. Note that both line directions are now next to each other and block off one of the line direction choices for each other (see figure 10b middle), so there is no longer a choice for direction. Because both those squares do not have any numbers in them the path cannot end there. We continue to fill in both edges till the end of the gadget and, because there is no place to deviate, we know that, that the start and end of the line are not in the gadget. Finally we see the leftover white squares can only be filled by indirectly connecting the adjacent outer pillar. So the entire gadget is filled as intended. In figure 10b the path is colored it blue and filled in with *a's* to show that it is a single line while the line of the inner pillar and outer pillar are filled with *b's* and *c's* respectively and are colored yellow.

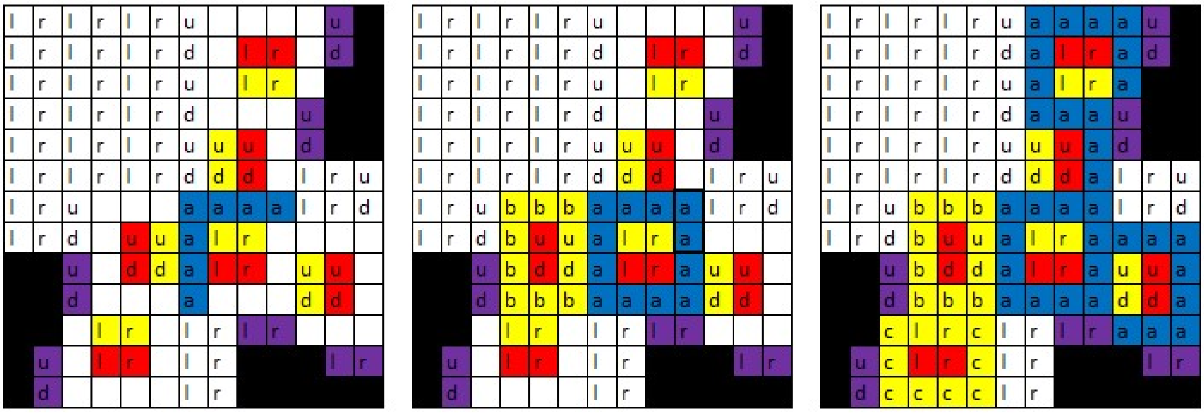

*Figure 11b The steps for case 1 and a filled in triple upward, rightward and downward edge*

In the second case we fill in the squares going right (to fill the upward edge) and left (to fill the downward edge) for both edges till the point where the path splits up (see figure 10c left). Next we see that the only white square directly adjacent to the central square can only be filled by its adjacent inner pillar when it is connected indirectly (see figure 10c middle). This eliminates the choice for both edges. We continue to fill in both directions till the end of the gadget. Finally we see the leftover white squares can only be filled by indirectly connecting the adjacent outer pillar. So the entire gadget is filled with the Hamiltonian path line. In figure 10c the path is colored it blue and filled in with *a's* to show that it is a single line while the line of the inner pillar and outer pillar are filled with *b's* and *c's* respectively and are colored yellow.

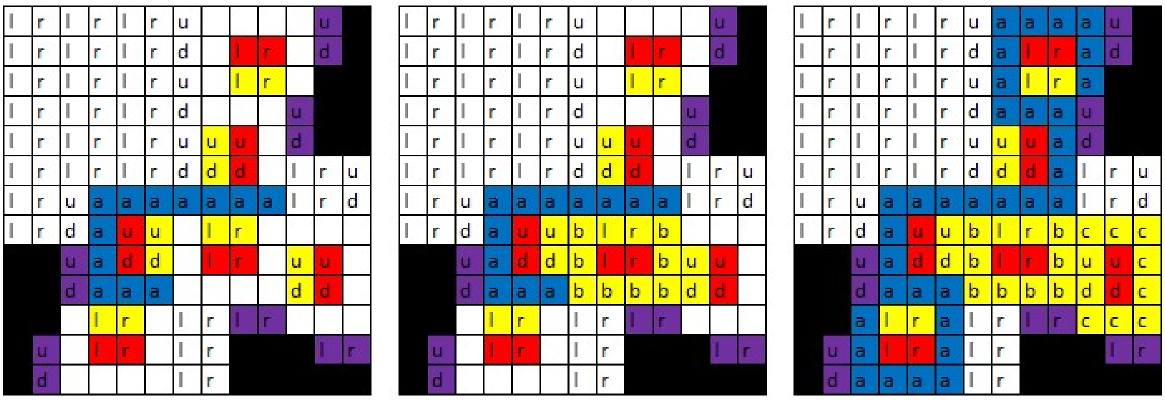

*Figure 12 The steps for case 2 and a filled in triple upward, rightward and downward edge*

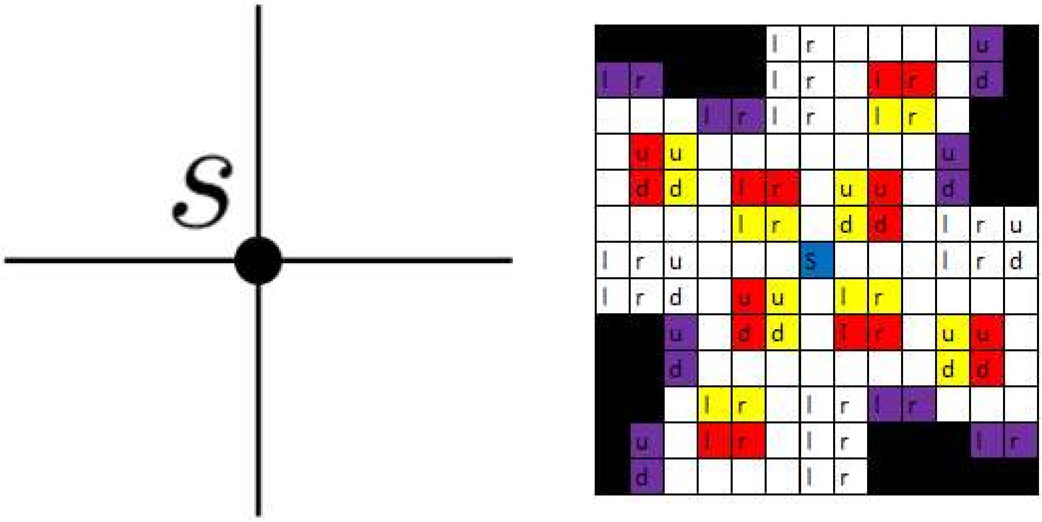


*Figure 13a A quadruple cross edge with corresponding gadget (as start the of a Hamiltonian path) [12]*

When the central square is the starting (or end) point of the Hamiltonian path we follow one edge (e.g. upward edge in figure 11b) till we get a choice (figure 11b left). Note that the squares directly adjacent to the central square can only be filled by the adjacent inner pillars. The only way to do that is by connecting the pillars indirectly (figure 11b middle). Now there is no longer any choice left for the Hamiltonian path line and we can fill it till the end of the gadget. The white squares directly connected to the line of the  indirectly connected inner pillars can only be filled by indirectly connecting the outer pillars. When we do that all white squares are filled. Now the entire gadget is filled in as intended. In figure 11b the path is colored it blue and filled in with *a's* to show that it is a single line. Further we use the letters '*b*', '*c*', '*e*', '*f*', 'g' and 'h' (Note that the letter '*d*' is already used) to represent the lines for the different pillars that are colored yellow.

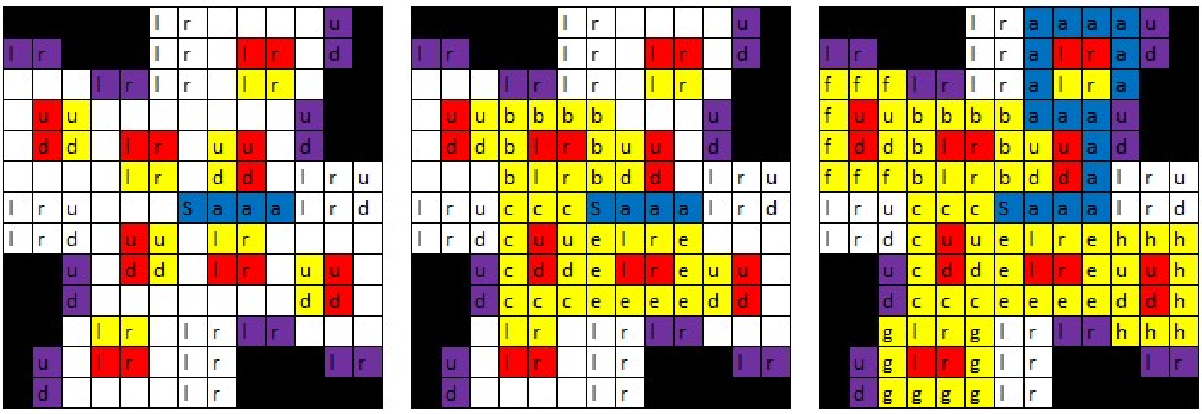

*Figure 14 The steps for a filled in quadruple cross edge (as start the of a Hamiltonian path)*

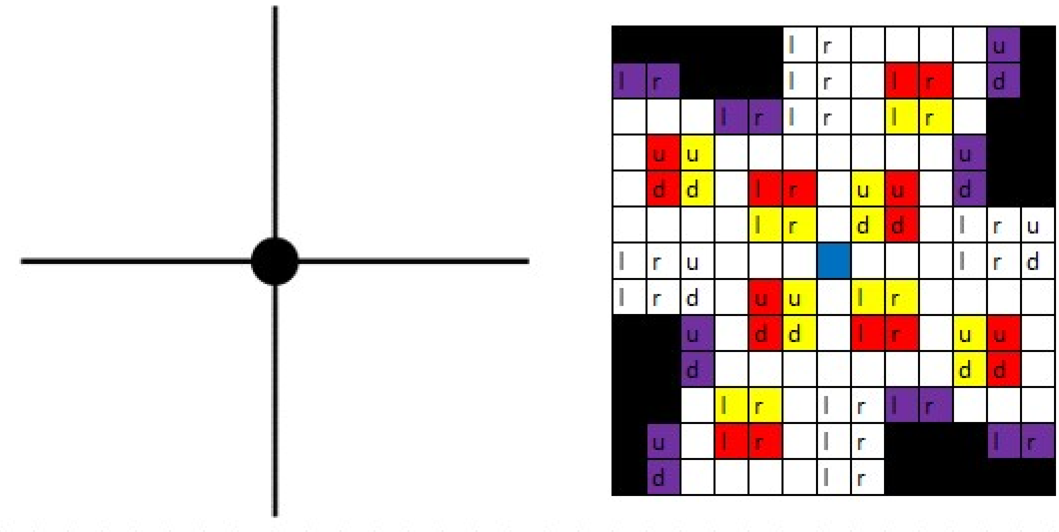

*Figure 12a A quadruple cross edge with corresponding gadget [12]*

For this gadget we need to differentiate between two different, namely when line edges form a corner (e.g. if the upward edge and the rightward edge are in a Hamiltonian path in figure 12a) and when the line edges form a straight line (e.g. if the upward and downward edge are in a Hamiltonian path in figure 12a). When the central square is part of a line we know that from the central square exactly two directions need to be filled with the same line.

In the first case we fill in the squares going right (to fill the upward edge) and down (to fill the rightward edge) for both edges till the point where the path splits up (see figure 12b left). Next we see that the only white square directly adjacent to the central square can only be filled by its adjacent inner pillar when it is connected indirectly. This eliminates the choice for the rightward edge. We continue to fill that edge in till point where there is again a choice. Note that both line directions are now next to each other and block off one of the line direction choices for each other (see figure 12b middle), so there is no longer a choice for direction. Because both those squares do not have any numbers in them the path cannot end there. We continue to fill in both edges till the end of the gadget and, because there is no place to deviate, we know that, that the start and end of the line are not in the gadget. Finally we see the leftover white squares can only be filled by indirectly connecting the adjacent outer pillars indirectly. So the entire gadget is filled as intended. In figure 12b the path is colored it blue and filled in with *a's* to show that it is a single line. Further we use the letters '*b*', '*c*', '*e*' and '*f*' (Note that the letter '*d*' is already used) to represent the lines for the different pillars that are colored yellow.

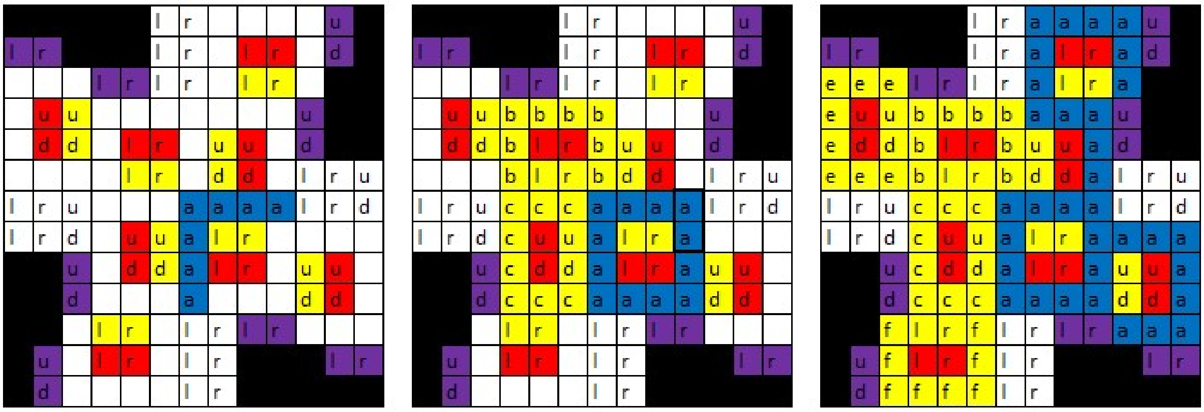

*Figure 15 The steps for case 1 for a filled in quadruple cross edge*

In the second case we fill in the squares going right (to fill the upward edge) and left (to fill the downward edge) for both edges till the point where the path splits up (see figure 12c left). Next we see that the white squares directly adjacent to the central square can only be filled by its adjacent inner pillars when it is connected indirectly (see figure 12c middle). This eliminates the choice for both edges. We continue to fill in both directions till the end of the gadget. Finally we see the leftover white squares can only be filled by indirectly connecting the adjacent outer pillar. So the entire gadget is filled with the Hamiltonian path line. In figure 12c the path is colored it blue and filled in with *a's* to show that it is a single line while the line of the inner pillar and outer pillar are filled with *b's* and *c's* respectively and are colored yellow.

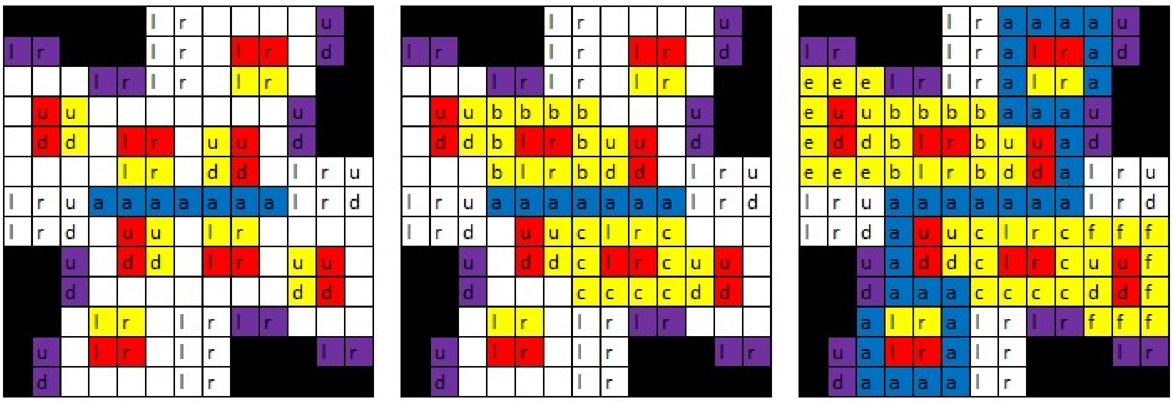

*Figure 16 The steps for case 2 for a filled in quadruple cross edge*

When there is a Hamiltonian path it is clear to see that there will always be a solution for the puzzle. We have every graph can be reduced to a puzzle, they connect in the shape of the graph and there is at least a way to fulfill the puzzle.

When there is a solution to the puzzle there is a Hamiltonian path. We have shown that to fulfill the gadgets that they either need to be the start (or end) of Hamiltonian path (which means there is a path) or that there is a line that passes through the central square. Because of the way the gadgets are setup such a line can only start from a central square, which means that there is a Hamiltonian path.

So Arukone$^3$ is solvable if and only if there is a Hamiltonian path.

The time it takes to reduce a Hamiltonian grid graph with m nodes to an arukone[3] puzzle is O(m) steps. So arukone[3] is NP-hard. □

### 3.3 Gadget connection

The gadgets will be connected in the same way as the nodes in the grid graph, except that the Gadgets overlap exactly 4×5 (or depending on the direction 5x4) squares (see figure 13). This is because the created lines follow a curve to fulfill the no 2x2 area rule, which would otherwise create a weirdly shaped structure. If the gadgets are connected this way then all the possible lines for the arukone[3] solution are in the same direction as the paths that are possible in the Hamiltonian path grid graph, tilted by 45 degrees because of the overlap of the gadgets. Here we also see the meaning of the purple parts. These need to coordinate between different gadgets because of the overlap.

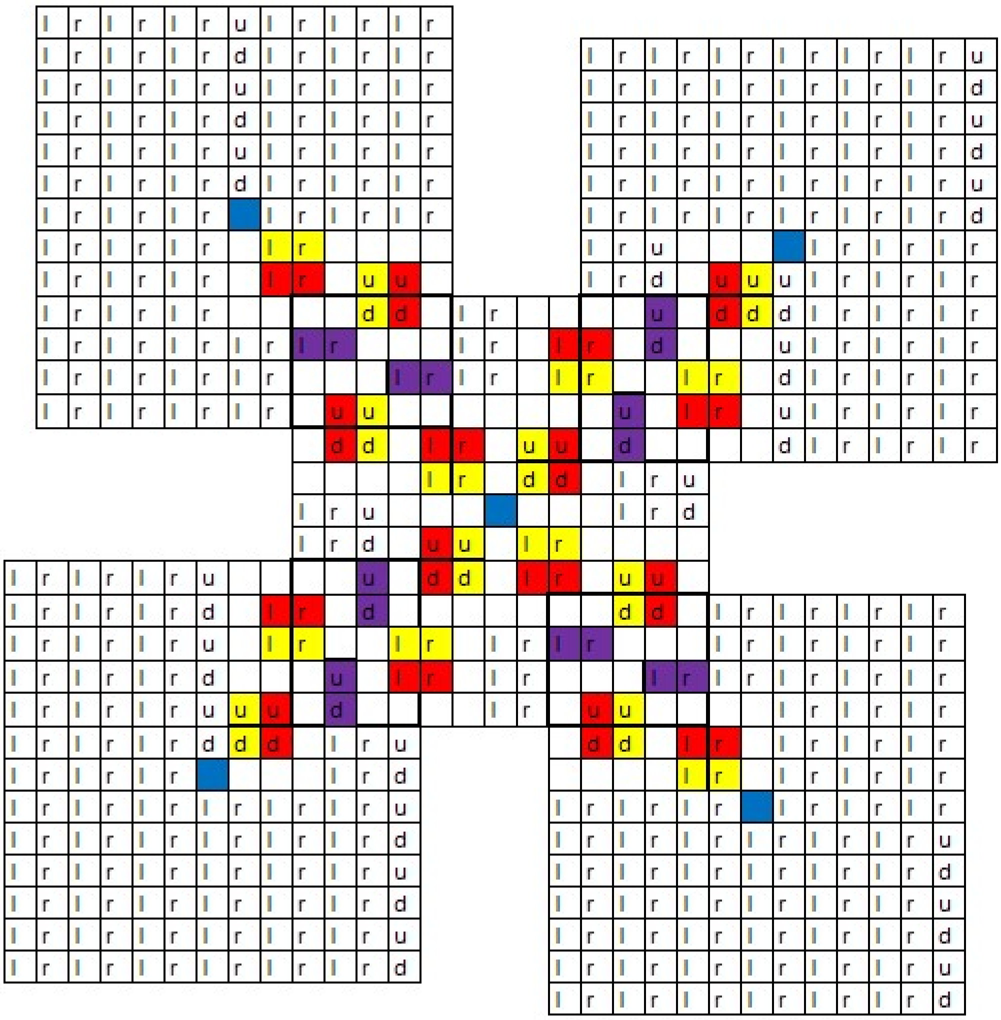

*Figure 13 Gadget with all possible connections shown*

### 3.4 Completing the reduction

Finally the only thing left to do is to show a full reduction.

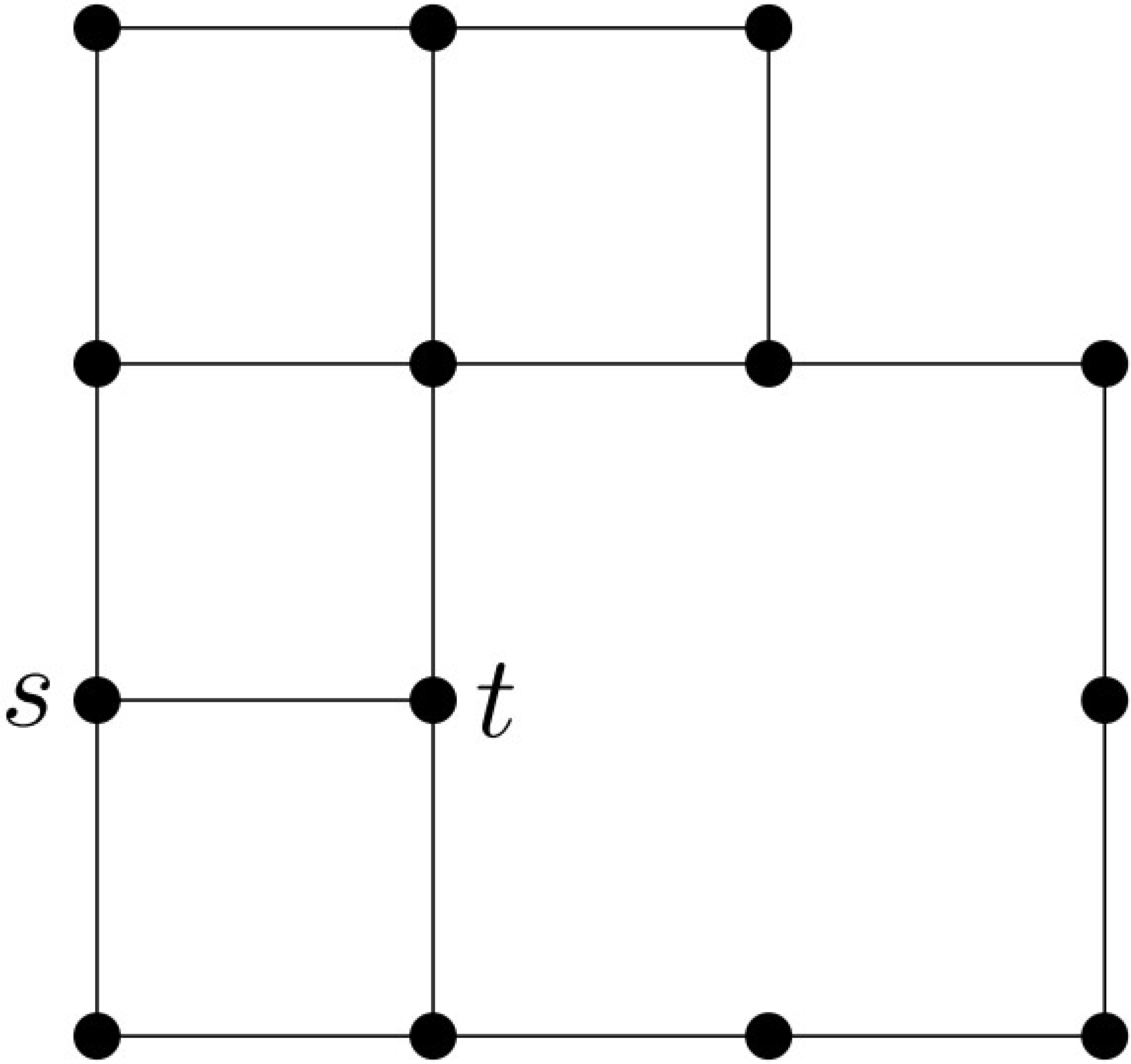


*Figure 17a A grid graph with a Hamiltonian path [12]*

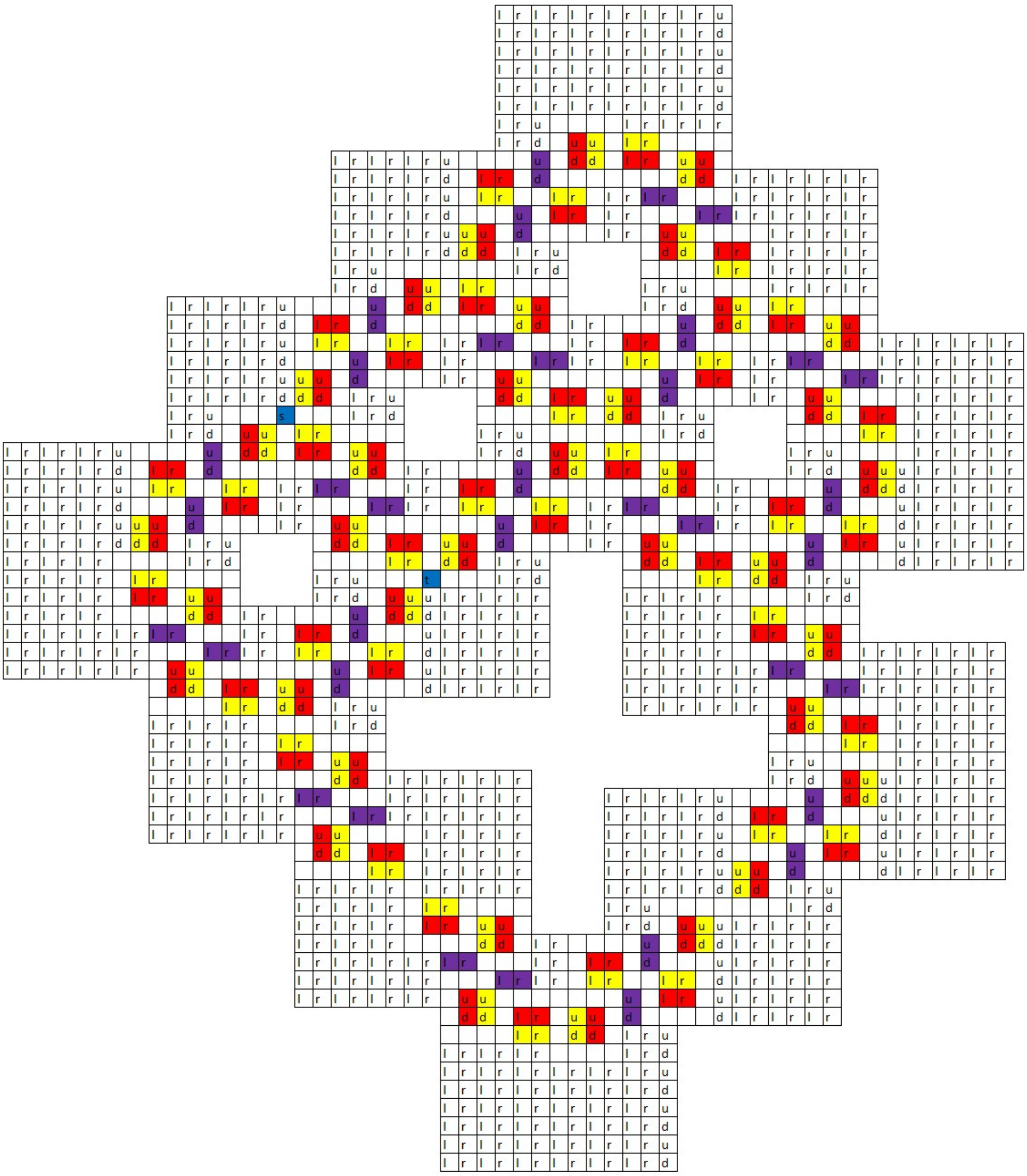

*Figure 14b Translation of the problem in 9a*

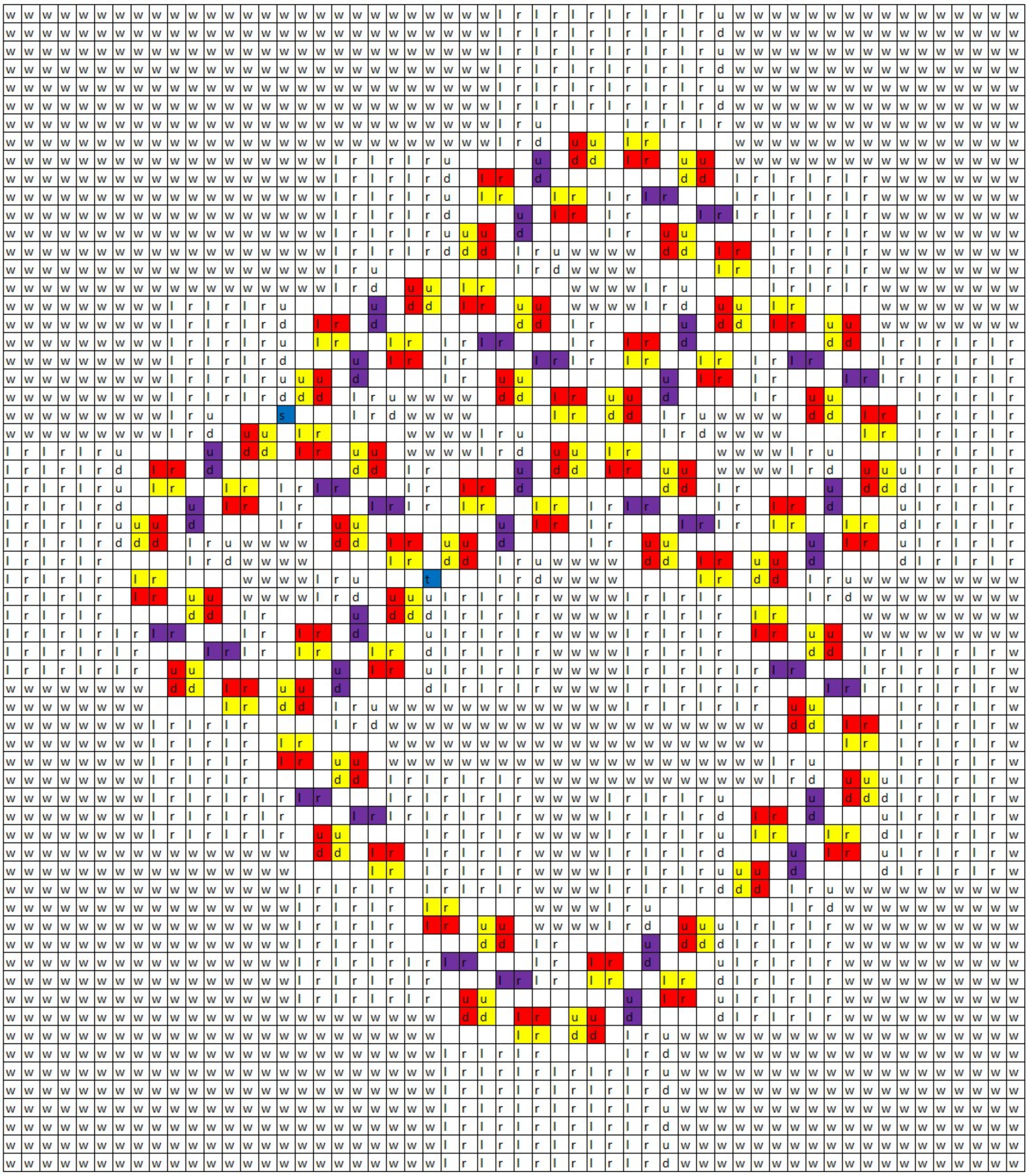

*Figure 14c A complete m x n Arukone$^3$ puzzle*

## 4. Bariasensa

Bariasensa is an n x n pen and paper puzzle on a grid. In the puzzle, some squares contain a number. The goal of the puzzle is to color squares black such that that for every numbered square, there exists at least one black square at distance x from it, where x is the number in that square. Those black squares cannot be horizontally or vertically adjacent. Furthermore, all white squares, including the squares with numbers in them, must form one orthogonally contiguous area. This means that all the white squares need to be horizontally and vertically connected. Finally the white squares can't cover an area of 2x2 or more.

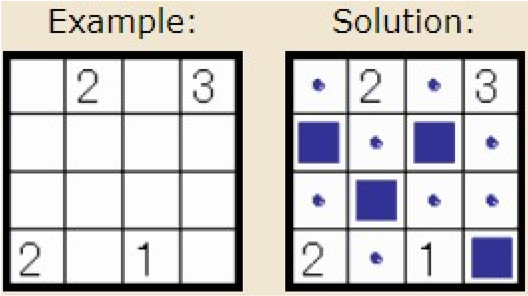


*Figure 18 Bariasensa puzzle with solution [10]*

**Theorem 2**. *Bariasensa is np-complete for k numbered squars in an m x n grid*

### 4.1 Proof idea

First we show that Bariasensa is in NP. We will show that given a solution we can check whether it is correct in at most polynomial time. Next we will try to show that Bariasensa is NP-hard. We will use the Planar NOR Circuit Satisfiability (Planar NOR CircuitSAT) problem as the NP-complete problem from which we reduce. The reduction will be done in the form of a proof by construction. We will build some gadgets in Bariasensa that represent parts of a planar NOR circuit.

A working reduction for Planar NOR CircuitSAT requires several components: a *wire gadget* to simulate a wire in a circuit, a *turn gadget* to bend or redirect the wire (gadget), a *split gadget* that makes sure that all instances of the same variable have the same value, a *crossover gadget* to make sure that wires can cross each other without interfering[1], *gate gadgets* to represent the workings of the Boolean operators and finally a *terminator gadget* to force the final output to be true.

For the gate gadget(s) we look for any combination of gates with which we can express all logical functions. The minimal set of logic gates necessary is exactly one gate . This can either be a NAND-gate or an NOR-gate. Both gates can be used to represent the full set of logic gates (AND-gate, OR-gate and NOT-gate). In this construction we use only NOR-gates, since all Boolean truth-functions can be expressed with just NOR-gates. We connect the gate gadgets in the Bariasensa puzzle in the same way in the Boolean circuit. For each component required in a planar Boolean circuit we will show and explain the corresponding gadget in the Bariasensa puzzle. We will find that the Boolean circuit is satisfiable if and only if the Bariasensa puzzle is solvable.

### 4.2 Proof

To show that Bariasensa is in NP, we show that a solution of Bariasensa is verifiable in polynomial time. To check if a solution is correct we have to check if

(1) all white squares are orthogonally connected (**connectedness rule)**,

(2) no black squares are orthogonally adjacent **(adjacency rule)**,

(3) there are no 2×2 white regions **(2×2 rule)**, and

---

[1] Because the simulated planar circuit is in the shape of tree and should not contain any crossing, but there is some crossing of the wires between the connection of the input variables and the leaves.

(4) each numbered square is exactly that distance from at least one black square **(number rule)**.

The worst case scenario to check is when there are as many number squares and black squares as possible as evenly divided between the two as possible. For an m x m puzzle there are $m^2$ number of squares. Let $n = m^2$. To check condition 1 the contiguity of white squares, it suffices to check for each white square whether there is a path (via orthogonal steps to other white squares) to the top left white square in the grid. Since pathfinding is polynomial-time computable and there are n total squares, this check is possible in polynomial time. To check condition 2 the 2×2 condition we need to check every 2x2 area individually if they fulfill the condition. There are $(m\text{-}1)^2 < n$ many 2x2 areas in the grid and each check takes constant time to do. So to check the 2x2 condition takes polynomial time. To check condition 3 (number rule) we need to check for every numbered square if there is a black square in the given distance. There are at most $m^2 = n$ many numbered cells. Each check for a black cell requires checking the color for up to 4 squares. So the time it takes for n numbered cells to be checked is polynomial time. For the final condition, condition 4 Black square adjacency, we need to check for each black square if there is not a black square orthogonally adjacent to it. Again there are at most n black cells and to check all orthogonally adject neighbor cells takes constant time. So this condition is also verifiable in polynomial time.

The total time to verify a Bariasensa solution takes polynomial time. So Bariasensa is in NP.

Next we show that Bariasensa is NP-hard. We do this by giving a reduction from the Planar NOR CircuitSAT problem to Bariasensa. It will be clear from the construction that, given a planar NOR circuit, the size of the corresponding Bariasensa puzzle instance will vary linearly with the size of the planar NOR circuit. Further, each individual gadget is simple to construct. Thus the reduction can be carried out in polynomial time.

Below we will explain the use and construction of each gadget. Note that the use of color in any of the images below is purely for the reader visualize the workings easier.

A **wire gadget** is used to represent the wires in a circuit. The gadget works as follows. The wire gadget is a combination of two pairs of numbers--e.g. in figure 16 we see two pairs of 3s--and some natural numbers (shown as n in the picture). It is possible to widen the gadget and make the left and/or right pair of numbers larger, if we desire a longer wire. The squares labeled *n* contain numbers, which are there solely to force those squares to be white. The number in one *n*-square need not be the same as the number in another. As will be explained later, these numbers will all be chosen in such a way that the black squares in the solution which satisfy them do not interfere with the rest of the puzzle solution. In figure 16, the input variable is represented by the leftmost square labeled *a*. Here, *a* is not a number, but rather a blank square that can be filled in either white or black in the solution, where white represents “on” or “true” and black represents “off” or “false”.

The wire gadget in figure 16 is oriented horizontally, but wire gadgets can also be horizontal by rotating the entire structure 90 degrees.

We now argue that the color of the square labeled *a* on the left must be the same as the color of the square labeled *a* on the right, as required for the wire gadget. (It will also turn out that the colors of the squares labeled *!a* will have the opposite color.) Suppose the input variable square *a* on a4 is white. Then the g4 square (labeled *!a*) must be black, by condition 4 and the number 3 on square d4 (note that the *n* squares cannot be black). Thus, by condition 2, it follows that g5 is white. Hence, the

3s on d5 and j5 make a5 and m5 black by condition 4. Finally, the latter fact and condition 2 make m4 white. The situation with the input variable square a4 being black is similar but with colors reversed.

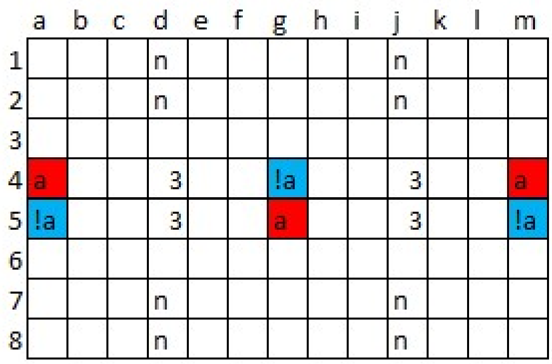


*Figure 16 A wire gadget*

A **turn gadget** consists of a vertical and a horizontal wire gadget with the output of one connected to the input of the other. See figure 17.

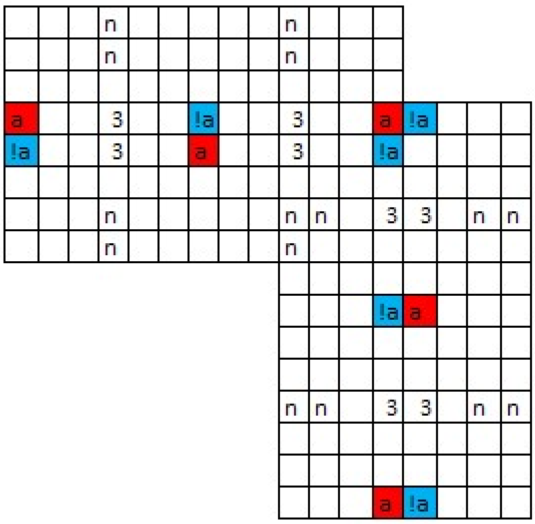

*Figure 17 A turn gadget*

A **split gadget** is used to represent a split wire; it has one input and multiple outputs, which all have the same value as the input. Their purpose in our reduction is to force all occurrences of a variable to have the same value. In the gadget for the Bariasensa puzzle we have an area where all variables of the same kind (e.g. the red variable a in multiple occurrences) are gathered. The split gadget consists of several attached wire gadgets of different sizes.

The size of the split gadget is equal to the width of the puzzle, which is determined by the width of the circuit.

Note that the split gadget in figure 18 is straight, but a split gadget can be combined with a turn gadget if a change of direction is desired.

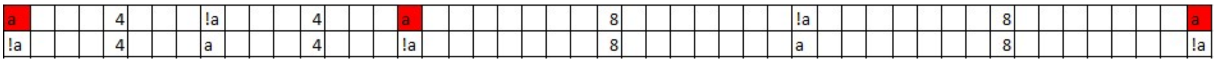


Figure 18 A split gadget (consisting of two wire gadgets)

The **crossover gadget** is simple to effectuate in Bariasensa, as vertical and a horizontal wire gadgets can easily intersect without interfering. Most of the wire gadget is open white space and won't block or interfere with any other wire gadget. The number pairs and some *n*-squares are the only squares that need to be treated carefully. However, because wire gadgets are easily stretched by adding extra white space and increasing the numbers, the intersection can be set in an area where both gadgets have white space. See figure 19 for an example.

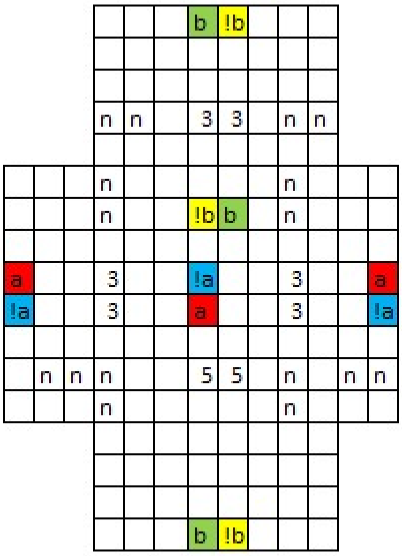


Figure 19 A crossover gadget (crossing an a and b wire gadget)

A **NOR-gate gadget** is constructed in 2 steps. First we place numbers to construct a wall of squares that are forced to be black by the 2x2 rule. These squares are highlighted in figure 20a.

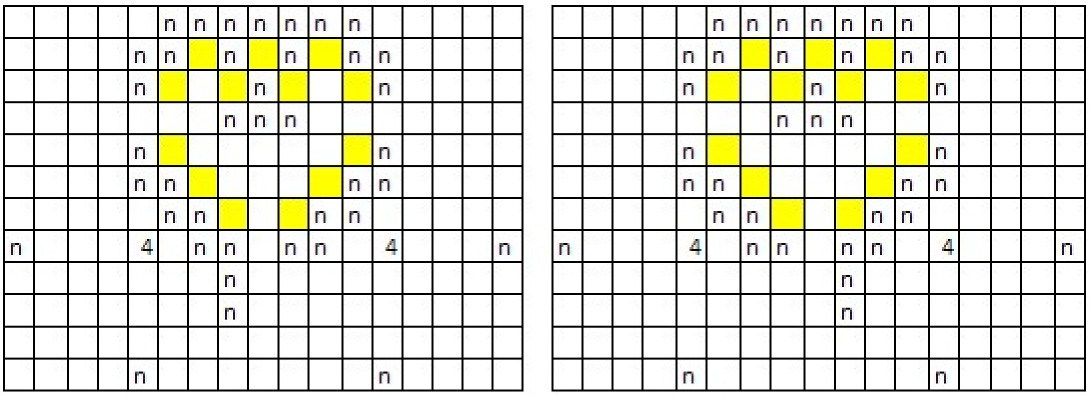


Figure 20a Step 1 NOR-gate construction

Conditions 2 (the adjacency rule) and 3 (the 2×2 rule) now force the squares highlighted in figure 20b to be black.

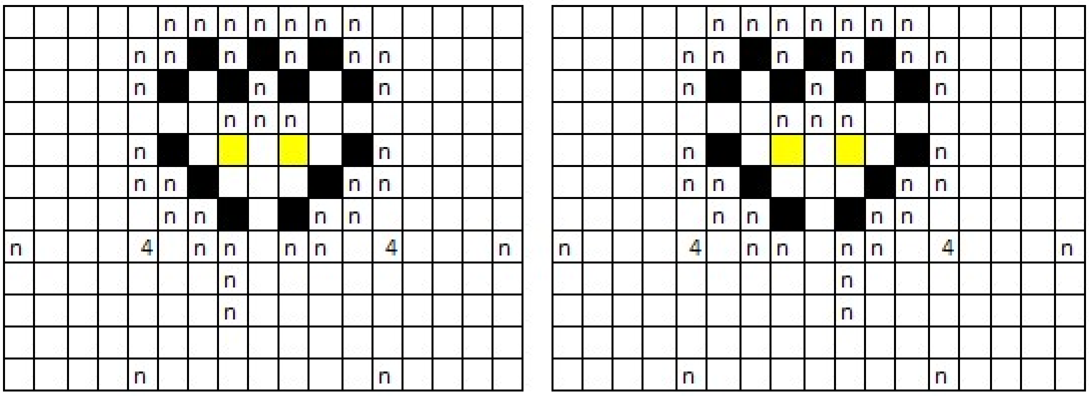

*Figure 20b Step 2 NOR-gate construction*

There are more squares that are forced to be black (e.g. the squares above the 4s), but those are not relevant for the NOR-gate so we did not color them in the image.

Next for the mechanism we have: variable input squares (green in figure 20c), sub-formula output (lower red square), specific numbers for the NOR-gate mechanism that depend on the gadget size (in figure 20c, the orange 4-squares) and some natural numbers (*n*-squares) to force whiteness.

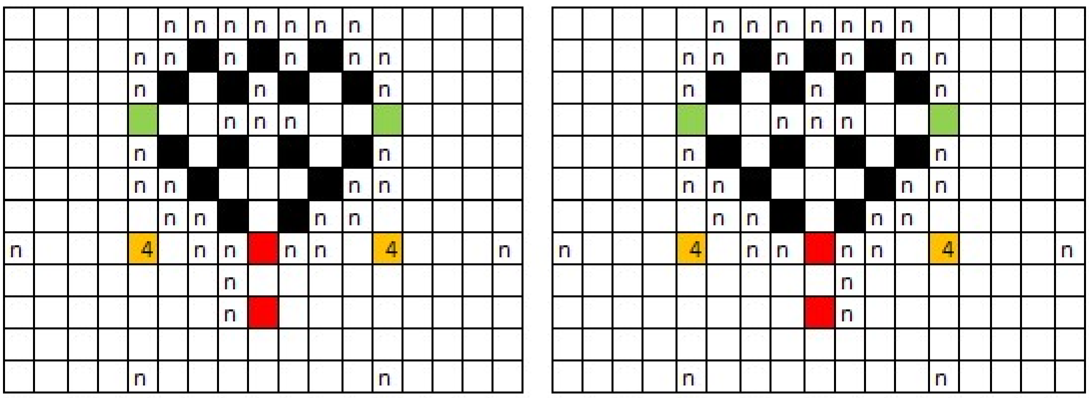

*Figure 20c NOR-gate workings*

If either one of the variable input squares is true (white) then the upper red square is forced to be false (black) because of condition 4 and the specific-numbered cells (since the *n*-squares at the edges of the image are not black). If both of the input squares are false (black) then because of condition 1 (connectedness), the upper red square is forced to be true (white). The output (lower red) square is guaranteed to have the same value as the upper red square by conditions 2 and 3, thanks to the nearby *n*-squares. The reason for including the lower red square in the gadget, rather than use the upper red square as the “output” square, is that the upper red square has adjacent *n*-squares, which interfere with the possibility of connecting them into other gadgets (e.g. wire gadgets leading to other NOR-gate gadgets). In figure 20c we show versions of the NOR-gate gadget with the *n*-squares on the left and on the right side of the output square. The version chosen for any part of the constructed puzzle will depend on the output value is being wired into the right or left side (respectively) of the next NOR-gate. See figure 22 for several examples of connected NOR-gate gadgets.

Also seen in figure 22 is that the NOR-gate gadgets may have different sizes depending on their location in the simulated circuit. Gate gadgets closer to the simulated circuit’s final output node have size equal to or greater than gate gadgets farther from the final output. The required size of the largest gate gadget grows linearly with the height of the circuit.

The **terminator gadget** is meant to be the determining gadget for whether the Bariasensa puzzle instance is solvable or not. We represent it in the Bariasensa puzzle with an area containing a square that, if black, would invalidate the first condition (connectedness). This square, highlighted yellow in figure 21, is wired to have the same value as the final output of the simulated circuit. The terminator gadget isolates a single white square (the central square in figure 21) if the output of the circuit is false (i.e. the yellow square is black) and keeps the square connected when it is true (e.g. the yellow square is white).

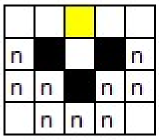

*Figure 21 A terminator gadget (with closing square in yellow)*

With these gadgets, it is straightforward to see that for any planar NOR circuit, the puzzle instance created by our construction will be solvable if and only if the circuit is satisfiable.

### 4.4 Completing the reduction

The size of the resulting puzzle from the reduction can be calculated based on the SAT formula we are trying to answer. The width of the puzzle is equal to 7 * (number of variable instances – 1) + number of variable instances + 2. We now consider the height. The Bariasensa puzzle represents a binary tree which consists of three parts. The variable inputs enter splitters, whose outputs are the leaves (on top). The root (at the bottom) is the output of the completed circuit with the terminator gadget. Finally, every node in the binary tree represents an output of a NOR-gate. The first part, consisting of the splitter and leaves, has for every different kind of variable in the formula a splitter gadget with a height of two rows. To make sure they don't influence each other, we need to have a number of white rows between them equal to at most half the width. The second part, the terminator gadget, only needs three rows (not counting the row containing the simulated circuit's final output square). Finally, the NOR-gates vary in size depending on their distance from the leaves. Each NOR-gate will be 6 + 4*n* in height, where *n* is the distance from the leaves. Between each layer we only need eight white rows. So in the worst case scenario (when the tree is fully skewed to one side) the height of the puzzle is $2a + \frac{1}{2}b(a-1) + 14c + 4d - 5$, where

—*a* is the number of different variables,

—*b* is the width of the puzzle,

—*c* is the number of NOR-gate layers,

—*d* is $\sum_{n=0}^{layers} n$ (number of NOR-gate layer (e.g. 3 layers = 6))

Given these height and width values, we see that the Bariasensa puzzle instance resulting from the reduction is no more than polynomially larger than the size of the circuit. Because every gadget takes at most polynomial time to construct, we can conclude that the reduction takes polynomial time. So Bariasensa is NP-hard. □

The puzzle instance corresponding to the circuit ((a NOR b) NOR (a NOR c)) NOR (b NOR a) is depicted in figure 22 below as an example. Note we use color and filled in some black squares to highlight some important aspects of the puzzle, namely the input variables (red), the space between split gadgets (yellow) and the NOR-gates and terminator (filled in black squares). These are not part of the puzzle.

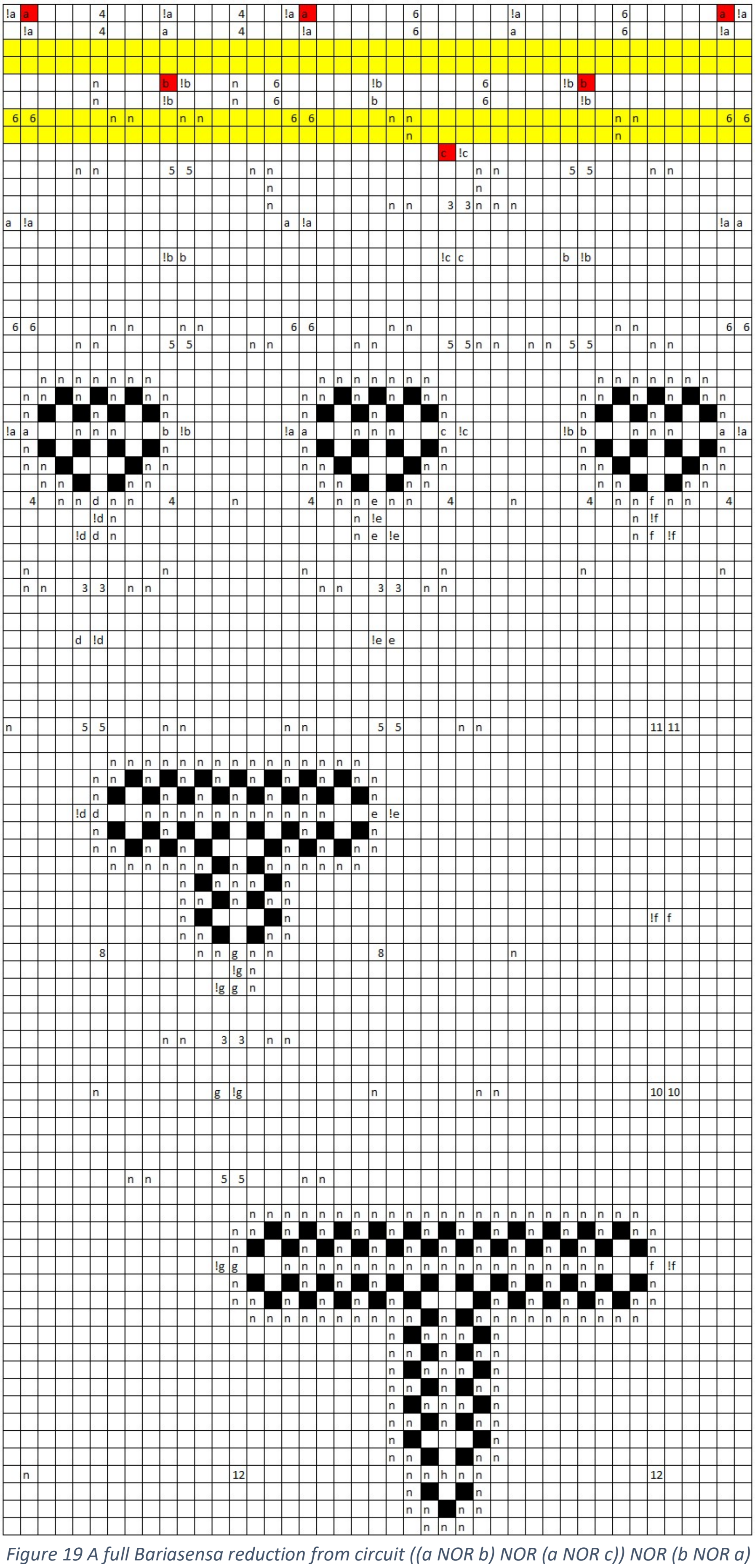

*Figure 19 A full Bariasensa reduction from circuit ((a NOR b) NOR (a NOR c)) NOR (b NOR a)*

The letters *a,b,c,* etc. denote blank squares containing representations of the input variables. They do not represent numbers. The labels *!a, !b, !c,* etc. denote squares in which the values are forced to be opposite those of (respectively) *a, b, c,* etc. The *n* labels do denote numerical values, not all necessarily the same, which are included solely to force the square containing the *n* to be white. These numerical values will be chosen so that they have no influence on the solution to the puzzle beyond the fact that the squares containing them are white. This can be done in several ways. One way is to choose them to be sufficiently high (well beyond the boundaries of the simulated circuit) so that their fulfillment by black squares has no effect on the rest of the puzzle. (This would raise the overall size of the puzzle beyond the size of the tree.) Another way is to choose numbers so that they are already fulfilled by squares we know will be black. For example, the *n*-square in the center of a NOR-gate gadget can contain a 2, and thus be fulfilled by the known black square two spaces above it.

## 5. Conclusion

In this paper we have proven that Arukone$^3$ and Bariasensa are both NP-complete. We did this via proof by construction. For Arukone$^3$ we found a reduction from the Hamiltonian path on a grid graph problem to the puzzle, while for the Bariasensa the reduction came from the Planar NOR CircuitSAT problem.

An interesting topic for future research is to try to find a reduction that preserves the number of solutions between the problems. By this we mean that for every satisfying set of inputs in the circuit, there should be a unique solution to the Bariasensa puzzle. In the present paper, given a uniquely satisfiable NOR circuit, the reduction to Bariasensa may output a puzzle instance that is solvable, but not uniquely so. The reason for this is that the large areas of blank space in the puzzle can be filled in many ways. A reduction that preserves the number of solutions, however, would need to include numbers placed in the white space in such a way that there is only way to fill the puzzle correctly with black squares.

A related question concerns the problem UNIQUE-CircuitSAT. UNIQUE-CircuitSAT is the set of circuits that have exactly one satisfying set of inputs. (Equivalently, we could consider UNIQUE-SAT.) It would be interesting to find a reduction from UNIQUE-CircuitSAT to Bariasensa (note that the project described in the above paragraph would lead us a reduction to UNIQUE-Bariasensa, but not to Bariasensa). The UNIQUE-CircuitSAT problem is complete for the complexity class US [11], which is not known to equal NP.

By contrast, the reduction for Arukone$^3$ in the present paper does preserve the number of solutions, in the sense that for every Hamiltonian path in the grid graph, there is precisely one solution to the corresponding Arukone$^3$ puzzle instance.

## 6. References

[1] Millenium problems: https://www.claymath.org/millennium-problems/millennium-prize-problems [Online; accessed 02-07-2021]

[2] Takayuki Yato and Takahiro Seta. “Complexity and Completeness of Finding Another

Solution and Its Application to Puzzles”. Imai.is.su-tokyo.ac.jp. [Online; accessed 02-07-2021].

[3] Akihiro Uejima, Hiroaki Suzuki and Atsuki Okada. “The Complexity of Generalized Pipe Link Puzzles”. In: Journal of Information Processing 25 (2017).

[4] Charles J. Colbourn. “The complexity of completing partial Latin squares”. In: Discrete Applied Mathematics 8, 25-30 (1984)

[5] Richard Kaye. “Minesweeper is NP-complete”. In: The Mathematical Intelligencer volume 22, 9–15 (2000)

[6] Jan N. van Rijn. “Playing Games The complexity of Klondike, Mahjong, Nonograms and Animal Chess”. master's thesis, Leiden Institute of Advanced Computer Science, Leiden University, [Online; accessed 02-07-2021].

[7] V. Nguyen. ”Rikudo is NP-complete”. (2021, January 22): https://arxiv.org/pdf/2101.09332.pdf [Online; accessed 02-07-2021]

[8] B. McPhail. “Light Up is NP-complete”. (2005, February 28): http://austiny.snu.ac.kr/down/Algorithm/PvsNP/Light%20Up%20is%20NP-complete.pdf [Online; accessed 02-07-2021]

[9] Arukone[3] puzzles: https://www.janko.at/Raetsel/Arukone-3/index.htm [Online; accessed 02-07-2021]

[10] Bariasensa puzzles: https://www.janko.at/Raetsel/Naoki/Bariasensa.htm [Online; accessed 02-07-2021]

[11] Andreas Blass and Yuri Gurevich. “On the Unique Satisfiability Problem”. In: Information and Control 55, 80-88 (1982)

[12] Mieke Maarse. “The NP-completeness of some lesser known logic puzzles”. (2019, June 28): https://dspace.library.uu.nl/handle/1874/383821 [Online; accessed 02-07-2021]