On Nonogram and Graph Planarity Puzzle Generation

Marc van Kreveld*

There are many puzzles in the world, both physical and digital ones. From the computational perspective, a lot of attention has been given to combinatorial puzzles and how to (algorithmically) solve them. We will focus on puzzles with a geometric component where techniques from discrete and computational geometry can be employed to generate them. We also present new nonogram and graph planarity puzzles.

1 Nonogram puzzles

In a basic nonogram, each row and each column has a clue that gives an abstract description as to which cells are to be filled. Suppose a row is eight cells long. Then a clue 1 3 specifies that in the row, one cell should be filled, and later in that row another three cells in sequence. Between the one and the three filled cells, there is at least one non-filled cell. Also, before the one filled cell and after the three filled cells, there are zero or more non-filled cells. In binary, 00101110 and 10011100 are both valid filling choices for the row of cells corresponding to the clue. A nonogram puzzle originally consists of a grid with no cells filled, see Fig. 1. A solution (or filling) is correct if for every row and column, the filled cells are an option for the given clue.

The scientific study of nonograms usually focuses on the algorithmic complexity of solving them [1, 2, 11, 13].

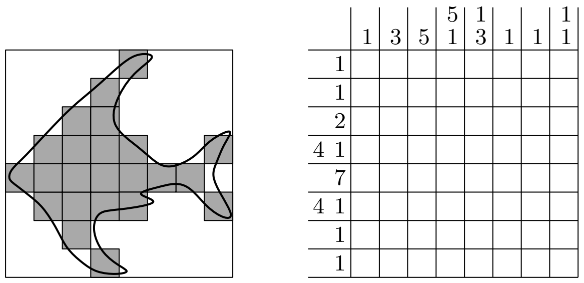

Figure 1: Basic nonogram (right) of the pixel image shown left.

In this talk we will discuss the problem of converting a simple polygon, representing some shape, into a nonogram. In essence, the polygon will become a low-res pixel image. We analyze how a pixel polygon can be formed that is simple and has small Hausdorff or Fréchet distance to the input. It is based on research by Bouts et al. [4].

We will also introduce new types of nonograms that generalize the basic type. Puzzles can be constructed based on any set of lines and even of curves. For these new types of generalized nonograms we review the design choices and let these inspire an automated method to generate a nonogram from a drawing. For nonograms based on curves, the curves replace the grid lines of basic nonograms and form an *arrangement* of cells with varying shape and size. Since rows and columns no longer exist, we need a new way to give clues indicating which cells should be filled by the puzzler. It allows for new reasoning steps that do not exist in a grid-based nonogram. This part is based on research by van de Kerkhof et al. [7].

2 Graph planarity puzzles

PLANARITY [10] is a popular abstract puzzle game that is widely available. The idea is that a tangled graph is given with intersecting edges, and the objective is to untangle the graph by dragging vertices to other locations. If the graph is planar, then the objective can always be realized, and we never need more vertex drags than there are vertices.

Algorithmically, planarity of a graph can be tested in linear time, and the algorithm returns an embedding of the graph in which it is drawn planar. So for an algorithm, an instance of PLANARITY is easily solvable in linear time. Minimizing the number of moves, however, is NP-hard [6, 12], see also [3].

In this talk we propose several variations on the game PLANARITY. These variations essentially limit the freedom of the operations that can be done on the drawn graph. Since the puzzle type is abstract, it is preferable that the interaction and operations themselves be simple. The puzzle might then become an elegant abstract puzzle of which there are many already (Move, Lines/Flow, Zengrams, Nintaii, Fling, ...).

Besides interacting with a vertex like in PLANARITY, it is natural to interact with an edge. Clicking or selecting is arguably the easiest interaction. We list a number of ways in which the drawing can change when an edge is selected:

- *Swap:* the two endpoints of the selected edge swap locations. Intuitively, the edge turns around while the endpoints drag all incident edges with them.

*Department of Information and Computing Sciences, Utrecht University, `m.j.vankreveld@uu.nl`

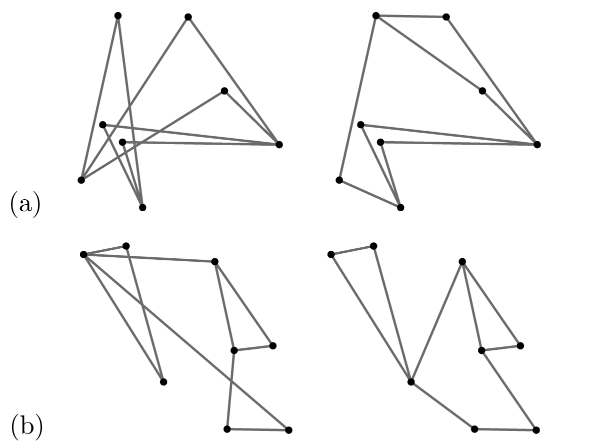

Figure 2: (a) Puzzle and solution after one swap (the left edge). (b) Puzzle and solution after two swaps.

- *Rotate:* like swap, but now the selected edge rotates over 90 degrees around its center. Since a single edge can be selected consecutively three times, it does not matter whether we rotate clockwise or counter-clockwise.
- *Stretch:* the selected edge is scaled by a factor 2 from its center, or by a factor 1/2.
- *Mid collapse:* the endpoints of the selected edge are united. The united vertex is placed in the middle of the edge and gets all edges incident to the original vertices. The selected edge is removed.
- *End collapse:* Same but the united vertex is placed at a selected endpoint.

We will investigate the first of the new variations closely: Swap Planarity. Examples are shown in Fig. 2. We show that quadratically many swaps are sometimes necessary (even if the input has just one edge crossing) and always sufficient; the latter follows from [14]. The decision (solvability) question is NP-complete for general graphs; this follows from [5]. Simple graphs like trees can always be made planar by swaps, but minimizing the number of swaps needed is NP-hard.

We discuss the automated generation of good puzzle instances for Swap Planarity by describing a five-step process which yields such a puzzle instance. Some of the considerations of a good instance are puzzle (complexity) based and some are geometry based. This part is based on research by Kraaijer et al. [8].

Acknowledgements. Research is supported by the Netherlands Organisation for Scientific Research (NWO) on grant no. 612.001.651

References

[1] Kees Joost Batenburg and Walter A. Kosters. Solving Nonograms by combining relaxations. *Pattern Recognition* 42(8):1672–1683, 2009.

[2] Daniel Berend, Dolev Pomeranz, Ronen Rabani, and Ben Raziel. Nonograms: Combinatorial questions and algorithms. *Discrete Applied Mathematics* 169:30–42, 2014.

[3] Prosenjit Bose, Vida Dujmovic, Ferran Hurtado, Stefan Langerman, Pat Morin, and David R. Wood. A polynomial bound for untangling geometric planar graphs. *Discrete & Computational Geometry* 42(4):570–585, 2009.

[4] Quirijn W. Bouts, Irina Kostitsyna, Marc van Kreveld, Wouter Meulemans, Willem Sonke, and Kevin Verbeek. Mapping polygons to the grid with small Hausdorff and Fréchet distance. In *24th Annual European Symposium on Algorithms, ESA*. LIPIcs, 22:1–22:16, 2016.

[5] Sergio Cabello. Planar embeddability of the vertices of a graph using a fixed point set is NP-hard. *Journal of Graph Algorithms and Applications* 10(2):353–363, 2006.

[6] Xavier Goaoc, Jan Kratochvíl, Yoshio Okamoto, Chan-Su Shin, and Alexander Wolff. Moving vertices to make drawings plane. In *Graph Drawing, 15th International Symposium, GD 2007*, volume 4875 of *Lecture Notes in Computer Science*, pages 101–112. Springer, 2008.

[7] Mees van de Kerkhof, Tim de Jong, Marc van Kreveld, Maarten Löffler, Raphael Parment, and Amir Vaxman. *Design and automated generation of Japanese picture puzzles.* Manuscript, 2018.

[8] Rutger Kraaijer, Marc van Kreveld, Wouter Meulemans, and André van Renssen. Geometry and generation of a new graph planarity game. In *Proceedings of the IEEE Conference on Computational Intelligence and Games.* To appear, 2018.

[9] Emilio G. Ortíz-García, Sancho Salcedo-Sanz, José M. Leiva-Murillo, Ángel M. Pérez-Bellido, and José Antonio Portilla-Figueras. Automated generation and visualization of picture-logic puzzles. *Computers & Graphics* 31(5):750–760, 2007.

[10] John Tantalo. Planarity. http://planarity.net/, 2007. Accessed: 2018-05-25.

[11] Jinn-Tsong Tsai. Solving Japanese nonograms by Taguchi-based genetic algorithm. *Applied Intelligence* 37(3):405–419, 2012.

[12] Oleg Verbitsky. On the obfuscation complexity of planar graphs. *Theoretical Compututer Science*, 396(1-3):294–300, 2008.

[13] I-Chen Wu, Der-Johng Sun, Lung-Ping Chen, Kan-Yueh Chen, Ching-Hua Kuo, Hao-Hua Kang, and Hung-Hsuan Lin. An Efficient Approach to Solving Nonograms. *IEEE Transactions on Computational Intelligence and AI in Games* 5(3):251–264, 2013.

[14] Katsuhisa Yamanaka, Erik D. Demaine, Takehiro Ito, Jun Kawahara, Masashi Kiyomi, Yoshio Okamoto, Toshiki Saitoh, Akira Suzuki, Kei Uchizawa, and Takeaki Uno. Swapping labeled tokens on graphs. *Theoretical Computer Science* 586:81–94, 2015.