Hindawi
Mathematical Problems in Engineering
Volume 2017, Article ID 8251305, 7 pages
https://doi.org/10.1155/2017/8251305

Research Article

Algorithm for Identification of Infinite Clusters Based on Minimal Finite Automaton

Biljana Stamatovic[1] and Goran Kilibarda[2]

[1]*Faculty of Information Systems and Technologies, University of Donja Gorica, Podgorica, Montenegro*
[2]*Faculty of Project and Innovation Management, Educons University, Belgrade, Serbia*

Correspondence should be addressed to Biljana Stamatovic; biljana.stamatovic@udg.edu.me

Received 5 April 2017; Revised 25 August 2017; Accepted 27 September 2017; Published 24 October 2017

Academic Editor: Yakov Strelniker

We propose a finite automaton based algorithm for identification of infinite clusters in a 2D rectangular lattice with $L = X \times Y$ cells. The algorithm counts infinite clusters and finds one path per infinite cluster in a single pass of the finite automaton. The finite automaton is minimal according to the number of states among all the automata that perform such task. The correctness and efficiency of the algorithm are demonstrated on a planar percolation problem. The algorithm has a computational complexity of $\mathcal{O}(L)$ and could be appropriate for efficient data flow implementation.

1. Introduction

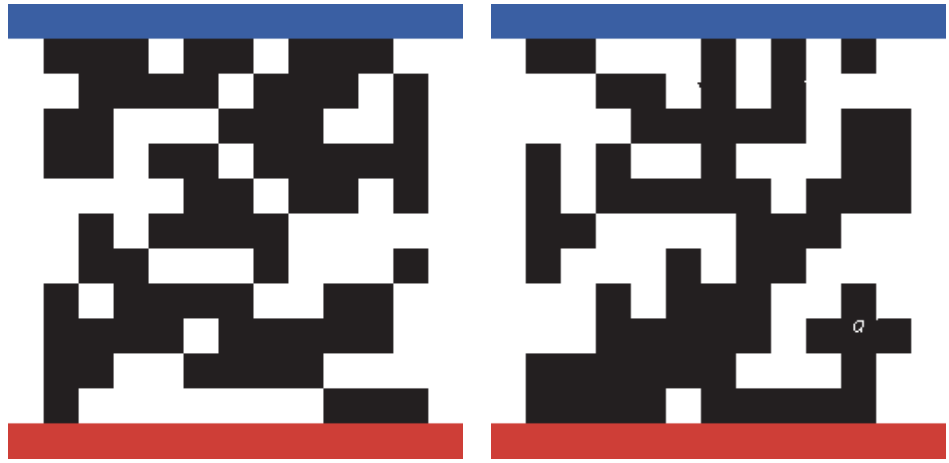
After the publication of Shannon's paper in 1951 [1], many scientists considered the behavior of systems of finite automata (FAs) in labyrinths, for example, in connected sets of cells of a grid or lattice. An overview of the obtained results is given in [2, 3]. Some problems of finite automaton (FA) traversing labyrinths can be found in [4, 5]. Many results indicate a limited power of FA. But, by introducing a collective of FAs, it was proved that there exists a collective of two FAs which can traverse all planar mosaic finite labyrinths [6]. Also, in [7, 8], the problem of FA recognition of some infinite classes of labyrinths (e.g., digits and letters) is studied.

Recognizing and labeling connected regions in binary images are important problems in image processing, machine vision, porous matter analysis, and many other fields of science. Studying site and bond percolation on any lattice is an important problem in computational physics [9–12]. Some algorithms from these studies are presented in [13–18]. In [19–21], polymerizing systems are presented, with branched or cross-linked polymers, which may go through distinct gelation transitions; a formed gel is seen as a two-dimensional lattice-percolation cluster and the authors use a "hull-generating walk" (HGW) algorithm.

The similarity between the papers [19–21] and the present paper consists in the fact that here we use a similar algorithm (in fact, HGW algorithm and the algorithm from our paper are in the class of the so-called "maze solving algorithms" [22]). But in contrast to papers [19–21], where HGW algorithm is used for generating perimeters of clusters in a 2D random binary grid, we use our algorithm for identifying and counting infinite clusters in such grid. This is a new, purely mathematical approach, in terms of the theory of finite automata, to the problem considered in our paper. It consists in the construction of a minimal finite automaton (FA) which can traverse the border of any cluster on any 2D binary grid and always stops in the planar site percolation model.

The planar site percolation model is represented by a 2D grid of square cells and each cell can exist in two different states, white or black. Any cell is colored black with probability p and white with probability $1 - p$. These probabilities are independent for each cell. We use open boundary conditions with boundary cells on the top and on the bottom of the grid in different states, blue and red, respectively. Cells on the left and right boundaries are white.

We construct an initial FA $\mathcal{A}_{q_0}$ which can traverse the border of any cluster in every planar 2D grid in accordance with the left-hand rule (which is also a kind of a HGW



(a) (b)

Figure 1: Examples of percolation site models on 13×13 grid, generated with probability $p = 0.55$ without infinite cluster (a) and generated with probability $p = 0.60$ with infinite cluster (b).

algorithm). The automaton $\mathcal{A}_{q_0}$ can be launched from any cell of a grid. In any moment, the input of $\mathcal{A}_{q_0}$ is information about von Neumann neighborhood of the present cell given in the form of a 5-tuple representing the state of the central cell and the states of its four neighbors. In the next moment, depending on its present state and its input, the FA proceeds from the present cell in one of the directions, defined by the output symbols e, n, w, or s (denoting the corresponding directions, east, north, west, or south, resp.), and moves into a corresponding neighbor cell or stays at the same place if output symbol is 0 (which means "no move") and passes into a new state. On the bases of this automaton, we give an algorithm which can be used for finding/identifying and counting infinite clusters in the given grid. Such algorithm is simple to implement and does not need extra memory or stacks. The time complexity of the algorithm is linear and not larger than the time complexity of already existing algorithms. The FA has only four states, including the initial state. We have proved that this is the minimal FA according to the number of states.

The paper is organized as follows. In Section 2, general grid-related definitions and definitions of the FA and its properties are given. In Section 3, the FA-based algorithm for counting connected paths is described and its complexity is proved. The main conclusions and results are summarized in Section 4.

2. Definition

2.1. General Definitions. The planar site percolation model (see Figure 1) is represented by a two-dimensional lattice network (grid) of unit squares (cells) whose centers are in the integer lattice with $L = X \times Y$ cells, $X, Y \geq 4$. The cell position is determined by its center. Each cell can exist in two different states, 0 or 1, where state 0 is usually called "white" and state 1 is usually called "black." The boundary cells of the simulated grid are in a fixed state, which remains constant throughout the simulation. The boundary cells on the bottom edge of the grid are in state 2, which is called "red," and the boundary cells on the top edge of the grid are in state 3, which is called "blue." The boundary cells on the left and right edge of the grid are white. State 4, which is called "orange," is used for the notation of labeled cells in the grid.

Two different cells (i_1, j_1) and (i_2, j_2) from a grid are *adjacent (weakly adjacent)* if $|i_1 - i_2| + |j_1 - j_2| = 1$ ($\max\{|i_1 - i_2|, |j_1 - j_2|\} = 1$). Two black (white) cells (i_1, j_1) and (i_n, j_n) are *connected* if there exists a sequence of black (white) cells (i_k, j_k), $2 \leq k \leq n$, such that each pair (i_{k-1}, j_{k-1}) and (i_k, j_k) are adjacent; such sequence is called a *path*. *Blue-red path* is a path whose one end is adjacent to a blue and the other to a red cell. A set of cells is connected if any two cells from the set are connected. A maximum connected set of black cells is called a *cluster*. The *infinite cluster* is a cluster that contains a blue-red path.

Similarly, black (white) cells (i_1, j_1) and (i_n, j_n) are *weakly connected* if there exists a sequence of black (white) cells (i_k, j_k), $2 \leq k \leq n$, such that each pair (i_{k-1}, j_{k-1}) and (i_k, j_k) are weakly adjacent. A maximum weakly connected set of white cells is called a *hole*.

2.2. Definition of the Initial Finite Automaton $\mathcal{A}_{q_0}$. Let $e = (1, 0)$, $n = (0, 1)$, $w = (-1, 0)$, and $s = (0, -1)$ be basic Euclidean vectors. Denote by V the set $\{e, n, w, s\}$. We use the notations $e = \overline{w}$, $w = \overline{e}$, $s = \overline{n}$, and $n = \overline{s}$. For a cell $c = (i, j)$ in a grid, its neighbors are cells $c + e = (i + 1, j)$, $c + n = (i, j + 1)$, $c + w = (i - 1, j)$, and $c + s = (i, j - 1)$. Hence, the von Neumann neighborhood is used in this paper (see Figure 2(a)).

Let $s_c = s_{(i,j)}$ be the state of the cell c and $\mathcal{N}(c) = \{\alpha \in V \mid s_{c+\alpha} = 1\}$.

Let $\pi = (e, n, w, s)$ be a cyclic permutation. For $M \subseteq V$, $M \neq \emptyset$, and $\alpha \in V$, denote by $\pi_M(\alpha)$ the element $\pi^{n_0}(\alpha)$, where $n_0 = \min\{n \in \mathbb{N} \mid \pi^n(\alpha) \in M\}$, where $\mathbb{N}$ is the set of natural numbers.

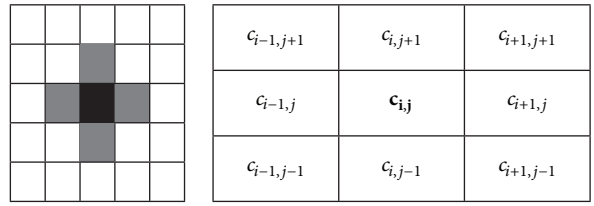


Figure 2: Von Neumann neighborhood and enumeration.

By a finite automaton (FA) $\mathscr{A}$, we mean a quintuple (A, Q, B, φ, ψ), where the finite nonempty sets A, Q, and B are the input alphabet, the set of states, and the output alphabet of the automaton, respectively; $\psi : Q \times A \to B$ is its output function; and $\varphi : Q \times A \to Q$ is its state-transition function. If we mark a state $q_0 \in Q$ as an initial state of $\mathscr{A}$, we get an initial automaton $\mathscr{A}_{q_0} = (A, Q, B, \varphi, \psi, q_0)$ (in other words, $\mathscr{A}_{q_0}$ is a Mealy machine).

Further, we consider the initial FA $\mathscr{A}_{q_e} = (A, Q, B, \varphi, \psi, q_e)$ supposing that $A = \{0, 1, 2, 3, 4\}^5$, $Q = \{q_e, q_n, q_w, q_s\}$, and $B = \{0, e, n, w, s\}$. The starting cell of this automaton in a grid is always the lowest right corner cell of the grid. If the output symbol of $\mathscr{A}_{q_e}$ is $\alpha \in \{e, n, w, s\}$, we say that FA proceeds in the direction α. If the output symbol is 0, the FA stops. This way, FA simulates movements (goes around, traverses) through a grid.

An input symbol $a = (a_0, a_1, a_2, a_3, a_4) = (s_{(i,j)}, s_{(i+1,j)}, s_{(i,j+1)}, s_{(i-1,j)}, s_{(i,j-1)}) \in A = \{0, 1, 2, 3, 4\}^5$ represents the state of a cell and the states of its von Neumann neighbors, consistent with the enumeration in Figure 2(b). For the input symbol a, by $c(a)$ we denote the central cell.

The FA starts in state q_e and uses states from set Q until it finds a boundary cell on the top of the grid (blue one). In this moment, the algorithm increments the counter of infinite clusters and starts labeling cells with an orange color. The algorithm labels a path, until the FA finds a boundary cell on the bottom edge of the grid (red one). After finding the lowest left end of an infinite cluster, the FA proceeds in the w (west) direction and it is on a white cell and in state q_e. After that, the algorithm breaks labeling and starts with new counting of infinite clusters and the FA repeats the process of traversing. The algorithm stops when the output of the FA is 0.

The functions φ and ψ are defined in the following way. Let $c = (i, j)$ be a cell and let $a = (s_{(i,j)}, s_{(i+1,j)}, s_{(i,j+1)}, s_{(i-1,j)}, s_{(i,j-1)})$, $x \in \{0, 1, 2, 3\}$, and $y \in \{1, 4\}$. Then, for $\alpha \in \{e, n, w, s\}$ and input $a = (y, x, x, x, x)$, the functions φ and ψ are defined by $\varphi(q_\alpha, a) = q_{\overline{\beta}}$ and $\psi(q_\alpha, a) = \beta$, where $\beta = \pi_{\mathcal{N}(c(a))}(\alpha)$ (see Figure 3) except for some "starting," "ending," or "changing" parts which are given by the following:

(i) If $i < X$, then

  (a) for $a = (0, x, x, x, 2)$ or $a = (1, x, 0, x, 2)$, $\varphi(q_e, a) = q_e$ and $\psi(q_e, a) = w$;
  (b) for $a = (1, x, 1, x, 2)$, $\varphi(q_e, a) = q_s$ and $\psi(q_e, a) = n$;
  (c) for $a = (y, x, x, 0, 2)$ and $q \in \{q_n, q_e\}$, $\varphi(q, a) = q_e$ and $\psi(q, a) = w$.

(ii) If $i = X$, then, for $a = (y, x, x, x, 2)$, $\varphi(q_e, a) = q_e$ and $\psi(q_e, a) = 0$.

Notice two properties of the FA $\mathscr{A}_{q_e}$:

(1) In the initial state, the FA is on the cell that has red cell on the south ($s_{(i,j-1)} = 2$). The FA goes in the west direction until it finds the black cell with black cell on the north. If it cannot find it, the FA stops.

(2) The meaning of the fact that the FA is on the cell c and is in the state q_α is that the FA has arrived on the cell c from the direction $\overline{\alpha}$, $\alpha \in V$ (see Figure 3).

The initial automaton $\mathscr{A}_{q_e}$ and the initial planar site percolation model $(C_0, c(0))$, where $c(0)$ is the cell on the lowest right corner of a grid C_0, create the sequence $S(A_{q_e}, (C_0, c(0))) = ((c(0), q_e, a_0, b_0), (c(1), q_1, a_1, b_1), \ldots, (c(i), q_i, a_i, b_i), (c(i+1), q_{i+1}, a_{i+1}, b_{i+1}), \ldots)$, where a_i represents the neighborhood of the cell $c(i)$ $(c(i) = c(a_i))$, $b_i = \psi(q_i, a_i)$, $c(i+1) = c(i) + b_i$, $q_{i+1} = \varphi(q_i, a_i)$, $i \geq 0$.

Let $R_s = \{(i, j) \in C_0 \mid s_{(i,j)} = 1,\ s_{(i+1,j)} = 0,\ s_{(i,j-1)} = 2\}$.

Properties of the FA $\mathscr{A}_{q_e}$, which will be crucial for the algorithm, are defined by the following theorem.

Theorem 1. *A blue-red path exists in the planar site percolation model C_0 if and only if there exists a cell $c'(0) \in R_s$ such that the sequence $(c'(0), c'(1), \ldots, c'(n))$, $n \geq 0$, of the cells in the sequence $S(\mathscr{A}_{q_e}, (C_0, c(0)))$ creates a blue-red path.*

Choosing the cyclic permutation π in the definition of the FA $\mathscr{A}_{q_e}$, we define an orientation. Namely, if the FA comes on the cell c from direction $\alpha \in V$, then the right, ahead, left, and behind cell (from the cell c) are $c + \pi(\alpha)$, $c + \pi^2(\alpha)$, $c + \pi^3(\alpha)$, and $c + \pi^4(\alpha) = \alpha$, respectively.

Let P be a blue-red path in the grid C_0 and K_P be the infinite cluster that encloses path P. Let $c_{rs} \in R_s$ be the lowest right cell in the cluster K_P and H_P be the subset of the infinite hole which is on the right side of the cluster. Right boundary of the cluster K_P, $K_{H_p} = \{c \in K_P \mid c$ is weakly adjacent to some cell from $H_P\}$, is connected and encloses a blue-red path. Notice that $c_{rs} \in K_{H_p}$.

It is not hard to see that the FA $\mathscr{A}_{q_e}$ implements "left-hand-on-wall" algorithm ([23]). Starting from the cell

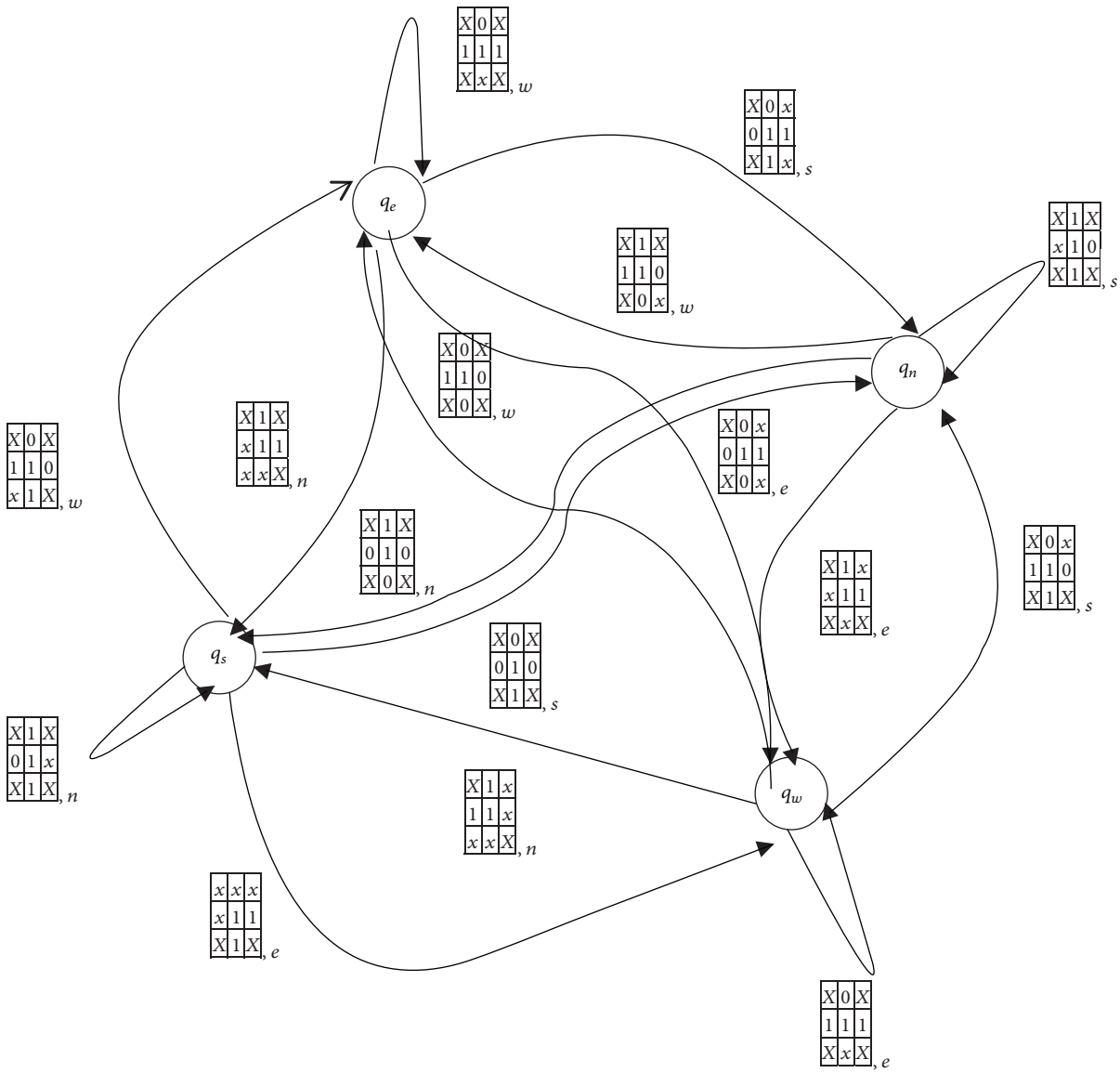


Figure 3: Traversing part of the FA $\mathscr{A}_{q_e}$, where $x \in \{0, 1, 2, 3\}$ before starting labeling.

c_{rs}, the FA traverses K_{H_p}, and the subsequence $(c'(0) = c_{rs}, c'(1), \ldots, c'(n))$, for some $n \in \mathbb{N}$, of the sequence $S(\mathscr{A}_{q_e}, (C_0, c(0)))$ creates a blue-red path.

If there is not a connected path between red and blue cells in the grid C_0, then, for all cells $c \in R_s$, the sequence $(c = c_0, c_1, \ldots, c_n, \ldots)$, $n \geq 0$, will not contain a cell with blue north neighbor. The grid is finite and the FA will stop on the lowest left cell of the grid.

Theorem 2. *According to the number of states, the FA $\mathscr{A}_{q_e}$ is the minimal initial finite automaton.*

In the example from Figure 1(b), notice cell a. It is in the infinite cluster and it is weakly adjacent by the right infinite hole. Traversing the cluster by the left-hand rule, the FA $\mathscr{A}_{q_e}$ is four times in the cell a (from south, east, north, and west sides). All the visits must be done in different automaton states (input is the same). Otherwise, the FA makes a cycle. Hence, for traversing a border of a cluster, it is necessary to have 4 states.

3. The FA Traversing Algorithm and Its Time Complexity

The algorithm for identification of infinite clusters is based on the FA $\mathscr{A}_{q_e}$. To ensure appropriate counting of infinite clusters, the algorithm uses three global variables: *count*, *labeling*, and *countingCluster*. The *count* counts the number of infinite clusters, *labeling* is an indicator for labeling cells, and *countingCluster* is an indicator for the counting of the infinite cluster.

Let $(C_0, c(0))$ be an initial grid, where C_0 is a grid and $c(0)$ is its lowest right corner. Let q_e be the initial state of the FA $\mathscr{A}_{q_e}$. The algorithm is presented as Algorithm 1.

Data: Initial grid $(C_0, c(0))$
Result: *count* is number of infinite clusters and labeled blue-red paths
count = 0;
currentState = q_e;
currentCell = $c(0)$;
Labeling = *false*;
countingCluster = *true*;
a = neighbor of the *currentCell*;
while $\psi(currentState, a) <> 0$ **do**
  if *labeling* == *true* **then**
    label *currentCell*;
  end
  if *north neighbor of currentCell is blue cell and countingCluster* **then**
    count = *count* + 1;
    labeling = *true*;
    countingCluster = *false*;
  end
  if *s*, *w neighbor of currentCell is red*, *white*, *respectly*, *and* *currentState* $\in \{q_e, q_n\}$ **then**
    labeling = *false*;
    countingCluster = *true*;
  end
  currentCell = *currentCell* + $\psi(currentState, a)$;
  currentState = $\varphi(currentState, a)$;
  a = neighbor of the *currentCell*;
end

Algorithm 1: FA traversing algorithm.

The FA starts in state q_e and uses states from set Q until it finds a boundary cell on the top of the grid (blue one). In this moment, the algorithm increments the counter *count* and sets the indicator *labeling* on *true*, and the variable *countingCluster* is set on *false*. The algorithm labels a path until the FA finds the lowest left cell on the bottom edge of the cluster. After finding the lowest left end of an infinite cluster, the FA goes in state q_e and sets *labeling* on *false* and the *countingCluster* is set on *true*. After that, the algorithm repeats the process of traversing.

The algorithm stops when the FA outputs 0.

Lemma 3. *The algorithm has linear time complexity.*

The FA uses von Neumann neighborhood with four neighbors for each grid cell. The automaton $\mathcal{A}_{q_e}$ is never on the same cell in the same state. Therefore, the FA cannot be more than four times in a particular cell during the execution of the algorithm. The complexity of the algorithm is therefore $\mathcal{O}(L)$, where L is the number of grid cells.

The algorithm was simulated in NetLogo programming environment. The algorithm was extensively evaluated on various test cases for different sizes of grid and initial density of black cells in initial configuration. In Figure 4, the last states on the initial grids from Figure 1 are shown. In Figure 5, a grid of dimension 201×201 with probability 0.62, for a black cell, is shown with two infinity clusters, after termination of the FA algorithm. By increasing the probability of black cells, infinite clusters more often appear, as predicted also by the percolation theory. Also, the results of the algorithm are in accordance with site percolation thresholds in the square lattice ([16, 24]).

4. Conclusions

A FA-based algorithm for identification of connected paths between top and bottom boundary cells in an arbitrary square grid is proposed. The constructed FA is minimal according to the number of states. The algorithm is based on four essential states: q_e, q_n, q_w, and q_s. Using these states as FA input and assuming von Neumann neighborhood of the grid cells, we can traverse the boundaries of clusters. Because the FA is independent, there is a possibility of putting more than one automaton in different initial cells. Such a parallel approach could further speed up the proposed algorithm. Combining the property of FA, which is never on the same cell in the same state, with labeling, there is a possibility of using the FA in the detection of articulation cells ([25]). Finally, further investigation can be focused on the necessary extensions of the algorithm for 3D grids.

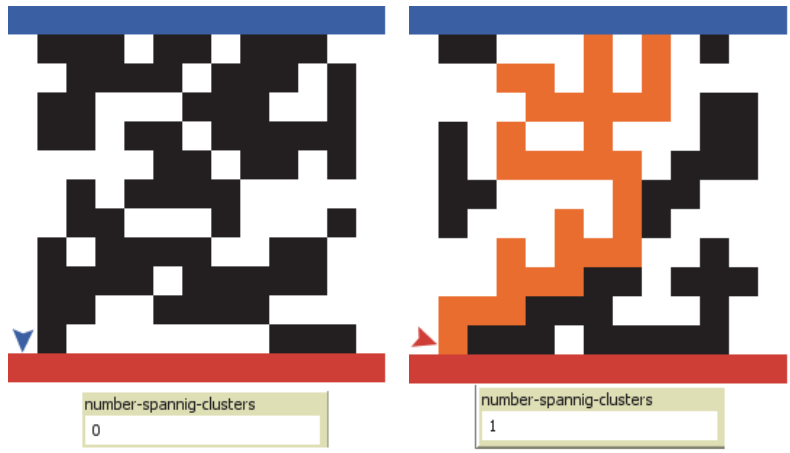


Figure 4: The FA traversing algorithm for the two examples from Figure 1.

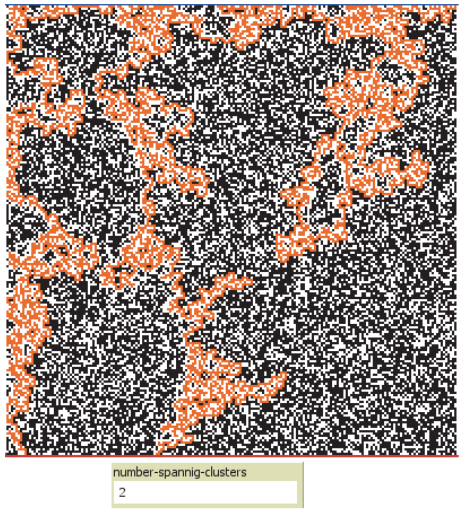


Figure 5: The FA traversing algorithm finds two infinite clusters in a grid of dimensions 201×201.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work is supported by the Program of Science-Technological Cooperation between governments of Montenegro and Serbia, 2016–2018.

References

[1] C. E. Shannon, "Presentation of a maze-solving machine, Cybernetics Trans," in *proceedings of the 8th Conf. of the Josiah Macu Jr. Found*, pp. 173–180, 1951.

[2] V. B. Kudryavtsev, S. M. Uscumlic, and G. Kilibarda, "The behavior of automata in labyrinths," *Discrete mathematics and applications*, vol. 4, pp. 3–28, 1992.

[3] G. Kilibarda, V. B. Kudryavtsev, and S. M. Uscumlic, "Collectives of automata in labyrinths," *Discrete Mathematics and Applications*, vol. 15, pp. 3–39, 2003.

[4] G. Kilibarda, "Universal labyrinth traps for finite sets of automata," *Discrete Mathematics and Applications*, vol. 2, pp. 72–79, 1990.

[5] G. Kilibarda, "On the complexity of traversing labyrinths by an automaton," *Discrete mathematics and applications*, vol. 5, pp. 116–124, 1993.

[6] M. Blum and D. Kozen, "On the power of the compass," in *Proceedings of the 19th Annual Symposium on Foundations of Computer Science*, pp. 132–142, 1978.

[7] B. Stamatovic, "On recognizing labyrinth with automata," *Discrete Mathematics and Applications*, vol. 12, pp. 51–65, 2000.

[8] B. Stamatovic, "Automata recognition two-connected labyrinth with finite cycle diameter," *Programming and Computer Software*, vol. 36, pp. 149–157, 2010.

[9] S. Kirkpatrick, "Percolation and conduction," *Reviews of Modern Physics*, vol. 45, no. 4, pp. 574–588, 1973.

[10] B. I. Shklovskii and A. L. Efros, *Electronic Properties of Doped Semiconductors*, Springer-Verlag, 1984.

[11] D. Stauffer, *Introduction to Percolation Theory*, Taylor & Francis, Abingdon, UK, 1992.

[12] , *Fractals and Disordered Systems*, A. Bunde and S. Havlin, Eds., Springer-Verlag, Berlin, Germany, 1996.

[13] J. Hoshen, P. Klymko, and R. Kopelman, "Percolation and cluster distribution. III. Algorithms for the site-bond problem," *Journal of Statistical Physics*, vol. 21, no. 5, pp. 583–600, 1979.

[14] E. Stoll, "A fast cluster counting algorithm for percolation on and off lattices," *Computer Physics Communications*, vol. 109, no. 1, pp. 1–5, 1998.

[15] R. C. Brower, P. Tamayo, and B. York, "A parallel multigrid algorithm for percolation clusters," *Journal of Statistical Physics*, vol. 63, no. 1-2, pp. 73–88, 1991.

[16] M. E. J. Newman and R. M. Ziff, "Efficient Monte Carlo algorithm and high-precision results for percolation," *Physical Review Letters*, vol. 85, no. 19, pp. 4104–4107, 2000.

[17] R. Trobec and B. Stamatovic, "Analysis and classification of flow-carrying backbones in two-dimensional lattices," *Advances in Engineering Software*, vol. 103, pp. 38–45, 2017.

[18] B. Stamatovic and R. Trobec, "Cellular automata labeling of connected components in n-dimensional binary lattices," *The Journal of Supercomputing*, vol. 72, no. 11, pp. 4221–4232, 2016.

[19] R. M. Ziff, P. T. Cummings, and G. Stells, "Generation of percolation cluster perimeters by a random walk," *Journal of Physics A: General Physics*, vol. 17, no. 15, article no. 018, pp. 3009–3017, 1984.

[20] R. M. Ziff, "Hull-generating walks," *Physica D: Nonlinear Phenomena*, vol. 38, no. 1-3, pp. 377–383, 1989.

[21] R. M. Ziff, "Spanning probability in 2D percolation," *Physical Review Letters*, vol. 69, no. 18, pp. 2670–2673, 1992.

[22] H. Fleischner, "Eulerian graphs and related topics," *Annals of Discrete Mathematics, No. 50, Part 1*, vol. 2, 1991.

[23] T. Y. Kong and A. Rosenfeld, "Digital topology: Introduction and survey," *Computer Vision Graphics and Image Processing*, vol. 48, no. 3, pp. 357–393, 1989.

[24] M. J. Lee, "Pseudo-random-number generators and the square site percolation threshold," *Physical Review E: Statistical, Nonlinear, and Soft Matter Physics*, vol. 78, no. 3, Article ID 031131, 2008.

[25] W.-G. Yin and R. Tao, "Algorithm for finding two-dimensional site percolation backbones," *Physica B: Condensed Matter*, vol. 279, no. 1-3, pp. 84–86, 2000.