On using cellular automata for modeling the evolution of dynamic-link network parameters

Erick Petersen, Jorge Lopez, Natalia Kushik, Claude Poletti, Djamal Zeghlache

On using Cellular Automata for Modeling the Evolution of Dynamic-Link Network Parameters

Erick Petersen[1,2], Jorge López[1], Natalia Kushik[2], Claude Poletti[1], and Djamal Zeghlache[2]
[1] *Airbus*, Issy-les-Moulineaux, France
[2] *Télécom SudParis, Institut Polytechnique de Paris*, Palaiseau, France

***Abstract*—We present a novel formalism for describing the evolution of dynamic-link network parameters; it is based on the Cellular Automaton (CA) model. Such formalism is of wide-use for modeling natural (e.g., physical, chemical, etc.) processes. We propose a particular model and survey the related work, with respect to the use of CA to simulate various communication networks. We showcase the flexibility of the proposed approach to model different evolution patterns. These patterns can be used to emulate / simulate different network scenarios (states of the network parameters), and test novel implementations under distinct conditions. Additionally, we propose an algorithm for guaranteeing that the described patterns hold properties of interest, within a bounded time.**

***Index Terms*—Dynamic-link networks, network prediction / evolution, Cellular automata, Formal models**

I. Introduction

The rapid growth of network communications and their capabilities is driven by the creation of innovative solutions that require flexible, dynamic and manageable capabilities to operate [1]. As a consequence, the verification and validation process of those novel solutions is critical to reduce risks and save time before their final deployment into a real network [2]. A well-known strategy is to evaluate new components on an emulated environment. In contrast to using operational networks for testing, emulations allow to replicate the behavior of real networks without extensive hardware components [3]. Moreover, emulation facilitates testing novel implementations under different scenarios, topologies, environment, and overall input values.

However, the emulation of *dynamic-link* networks, i.e., networks whose parameters may change at different time instances, is usually time consuming and computationally demanding. This is mainly because emulators *try to* replicate all the hardware and software functions of a network environment with its components as if those were real ones. For example, satellite networks require not only the emulation of the hardware components of the network, but also the links whose parameters may vary due to external interference, propagation conditions (weather), traffic variations (due to the shared medium) or transmitters' displacement [4]. Therefore, in some instances where many emulations are required for finding interesting configurations, alternative methods and approaches are needed. Such methods should on the one hand, take away certain hardware / software constrains and requirements, and on the other, decrease the time to produce the network instances of interest. The latter represents one of the objectives of this work. We thus rely on the network simulation and related mathematical models.

Note that simulation has proven to be more flexible, scalable and less time consuming on large networks (we provide a comparison in Section IV), but with limited applicability[1]. In this paper, we propose a novel approach for simulating the evolution of dynamic-link network parameters based on cellular automata [5]. Indeed, CA have shown their effectiveness in modeling dynamic processes, such as physical or chemical ones [6]–[8], for example. They allow both to showcase the evolution of the process itself, and to visualize the process accordingly. A current configuration of the automaton represents the state of the process at time t, that is afterwards changed according to the update functions defined in each *cell*.

In this paper, we show how a CA formalism is suitable for modeling dynamic-link network parameters, i.e, parameters that can change iteratively w.r.t. some *update* pattern. This pattern is thus utilized for the update function construction. Note that we rely on a two dimensional (2D) CA where the possible space has only two components and each cell contains the current parameter value (or several of them, if needed). We take advantage of the visualization capabilities of the 2D CA and showcase several static network instances (for different time instances). At the same time, we use the CA to quickly model the network evolution and potentially *fast transfer* to a configuration where network parameters reach some critical values. This provides a way of generating initial network configurations that can be further used in testing novel (e.g., routing or network prediction) solutions for dynamic-link networks.

The paper is organized as follows. Section II gives a brief introduction to the concepts of this work. Section III summarizes the related work. Our CA approach for modeling dynamic-link networks is presented in Section IV. Finally, we conclude and describe foreseen future work in Section V.

II. Preliminaries

A. Cellular Automata

Cellular Automata [8] are mathematical systems constructed from many (identical) components, thought of as *cells*, that

979-8-3503-9730-7/22/$31.00 ©2022 IEEE

[1]For example, run-time performance, resource usage or protocol interoperability evaluations are difficult to qualify at this level of abstraction.

have proved useful to model complex behavior of non-linear dynamics from cells cooperative effects. Each cell state evolves in time steps according to *update functions*, also called local rules, based on the states of their neighboring cells. CA were first introduced by J. von Neumann [5] as formal models of self-reproducing organisms, systems capable of producing exact copies of themselves. Nevertheless, the applicability of CA extends not only to biology [9], but also to a wide range of scientific fields such as diffusion models in chemistry and physics [6], urban sprawl in geography [10], non equilibrium dynamics in physics [7], and simulation games as the very well-known "Game of Life" [11], [12] in which it was demonstrated that CA models are capable of producing dynamic patterns.

Formally, a Cellular Automaton $\mathcal{A}$ is a four-tuple $\langle d, S, N, f \rangle$ [13], where:

- $d \in \mathbb{Z}_+$ is the dimension of the space;
- S is a finite set of states;
- $N \in (\mathbb{Z}^d)^m$ is a neighbourhood index (m-dimensional vector of d-dimensional vectors);
- $f : S^m \to S$ is a transition or update function.

The neighbours of a cell at location $\mathbf{x} \in \mathbb{Z}^d$ are defined by the neighbourhood index as: $\mathbf{y_i} = \mathbf{x} + \mathbf{x_i}, \forall i \in \{1, \ldots, m\}$. A CA configuration c is a function that maps a cell index to a state, i.e., $c : \mathbb{Z}^d \to S$. The cellular automaton at time $t+1$ has a configuration $c_{t+1}(\mathbf{x}) = f(c_t(\mathbf{x} + \mathbf{x_1}), \ldots c_t(\mathbf{x} + \mathbf{x_m}))$, for the neighbourhood index $N = \langle \mathbf{x_1}, \ldots, \mathbf{x_m} \rangle$. In this work, we denote an initial configuration as $c_0(\mathbf{x})$. Furthermore, we denote the configuration of the cells of interest at time t as the matrix $\mathbf{C}_t$, for example the matrix $\mathbf{C}_0$ denotes the initial configuration for all the cells of interest.

B. Dynamic-link networks

As introduced in [14], a static network is a computer network where each link has a set of parameters that do not change, for example bandwidth (capacity) or delay. Differently from static networks, the parameters of the links may change in dynamic-link networks (in the scope of our current work, we assume the network topology does not change); such change can be the consequence of the physical medium (e.g., in wireless / radio frequency networks) or due to logical changes (e.g., rate limiting the capacity of a given link).

Static networks can be modeled as (directed) weighted graphs $(V, E, p_1, \ldots, p_k)$, where V is a set of nodes, $E \subseteq V \times V$ is a set of directed edges, and p_i is a link parameter function $p_i : E \to \mathbb{N}$, for $i \in \{1, \ldots, k\}$; without loss of generality, we assume that the parameter functions map to non-negative integers (denoted by $\mathbb{N}$) or related values can be encoded with them. Similarly, dynamic-link networks can be modeled as such graphs, however, p_i maps an edge to a non-empty set of integer values, i.e., $p_i : E \to 2^{\mathbb{N}} \setminus \emptyset$, where $2^{\mathbb{N}}$ denotes the power-set of $\mathbb{N}$. An example dynamic network is depicted in Fig. 1, and its model $\mathcal{N} = (V, E, p_1(e), p_2(e))$, where:

$$
\begin{aligned}
V &= \{1, 2, 3, 4\} \\
E &= \{(1,2), (2,1), (1,3), (3,1), (1,4), \\
&\quad (4,1), (2,4), (4,2), (3,4), (4,3)\} \\
p_1(e) &= b((s,d)) = \mathbb{N}_{256} = \{0, 1, \ldots, 255\} \\
p_2(e) &= d((s,d)) = \begin{cases} \{1,2\} & \text{if } d = 2 \\ \{9,10\} & \text{otherwise} \end{cases}.
\end{aligned}
$$

Semantically, this model represents a dynamic-link network in which the link's available bandwidth can vary according to the function b (for *bandwidth*), and the link's delay can vary according to the function d (for *delay*). In this work, we take advantage of such representation to further build a CA simulating the evolution of the network itself, i.e., the dynamic changes of the parameters.

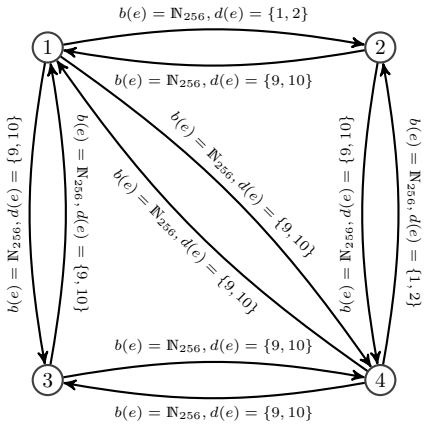

Fig. 1. Example dynamic-link network

III. Related Work

In this section, we discuss the existing works on using the CA for modeling and prediction in computer networks. In this context, most of the CA models found in the literature are used for either Wireless Sensor Networks (WSNs) or Mobile Wireless Sensor Networks (MWSNs) [15].

Sen and Barman [16] proposed a CA model for finding minimum spanning tree (in a WSN) with distinct edge weights to obtain an efficient transmission mechanism for data collection and data distribution. Fu et al. [17] proposed a 2D CA model to mimic WSNs failure events. For that matter, special failure states of sensor nodes were introduced, namely, energy exhaustion, hardware/software malfunctions, intentional attacks, random attacks and isolation from the sink node. Heidari et al. [18] proposed a routing protocol named Particle-based SMO to enhance the security while routing the WSN sensor nodes. The algorithm's performance was analysed using a CA based node scheduling model and a CA based malware propagation model. A probabilistic CA model was considered in [19] in order to solve the problem of an optimal coverage of a WSN area. The objective was to find the CA rules able to form a sensor coverage pattern with a minimum number of active sensors that cover the whole WSN. Meanwhile, Graph Cellular Automata (GCA) were studied in

[20] for solving the Maximum Lifetime Coverage Problem (MLCP) in WSN, i.e., using only some knowledge about neighbors, a WSN is able to self-organize in such a way as to prolong its lifetime and preserving the required coverage ratio of the target area.

When it comes to networking applications, the reader can refer to Tavanpour et al. [21], for example, who proposed the related CA model. A mobile network was modeled to track the users' upload status in a given coverage area, while utilizing either a non-cooperative algorithm or Coordinated Multipoint (CoMP) synchronization. A WSN was also modeled to investigate the behavior of malware propagation and its effects on the sensors' energy consumption. Choudhury [22] proposed to use the CA models to solve two well known optimization problems in WSN, namely, to maximize the sensor coverage area and minimize the sensor movement for energy conservation.

CA were also considered in the analysis and prediction for mobile networks. Cardoso et al. [23] proposed a multilayered CA solution for positioning flying cells to improve the network capacity in heavy traffic situations such as congested car avenues, crowded events, disasters, etc. The proposed scheme considered both backhaul and radio access network constraints, as well as user requirements in terms of down link throughput. In [24], a CA based QoS-routing solution for MANETs was proposed, taking into account the nodes' energy and delay constraints. Tsompanas et al. [25] proposed a CA based solution to tackle the shortest path problem in different fields. For communication networks, different path metrics were used to define the links' cost such as propagation delay, link congestion or reliability. Affine CA with linear output operators were studied in [26] from the observability point of view. Such an observer, mobile sensor, would allow the use of the CA for control, diagnosis or general supervision purposes such as, for instance, pollution or traffic monitoring.

To the best of our knowledge, no CA models were considered in the context of dynamic-link networks or the related evolution of network parameters. Thus, this work proposes the first CA model for such purpose.

IV. Cellular Automata for Modeling Dynamic-Link Networks

In this section, we showcase the use of CA for creating patterns of the evolution of dynamic-link networks. We do this in order to be able to control this pattern, instead of simply choosing random values from the set of possible values for the links' parameters. Indeed, this allows us to create natural patterns for parameters such as, for instance, the bandwidth capacity. As an example, consider a satellite link network. The bandwidth capacity varies according to the weather conditions, including the density of the clouds. With this in mind, the CA that models the evolution of a cloud density over time, can also model the bandwidth capacity for that link. Formally, we will consider two-dimensional cellular automata models as follows.

Definition 1: For a given dynamic-link network $\mathcal{N} = (V, E, p_1, p_2, \ldots, p_k)$, a dynamic-link parameter pattern for a network parameter p_j is a cellular automation $\mathcal{A}_{p_j} = \langle 2, S, N, f \rangle$, where:

- $S = \bigcup_{e \in E} p_j(e)$, where p_j is the parameter function for the j-th parameter;
- N is a neighbourhood index defined by all the incoming or outgoing nodes from the cell (x, y), i.e., $N = \langle (0,1), (0,2), \ldots, (0,|V|), (0,-1), (0,-2), \ldots (0,-|V|), (1,0), (2,0), \ldots, (|V|,0), (-1,0), (-2,0), \ldots, (-|V|,0) \rangle$;
- f is an arbitrarily chosen computable function that does not contradict the values of p_j.

Note that the cell (i, j) represents the link from node i to node j, accordingly.

As an example, consider the topology (and related dynamic-link parameters) in Figure 1. Let us focus on the parameter b (or p_1). In this example, the bandwidth takes different values ranging from zero to 255. However, for a given configuration, what is the next configuration? There is no established pattern of the evolution of these values. Likewise, there is no notion of a sequence. Indeed, the formalism allows any possible value (from the set of possible values). However, with the proposed formalism, we can define, for example $S = \{0, \ldots, 255\}$, and the update function:

$$f(i,j) = \left(\sum_{\mathbf{y_i} \in N} c_t((i,j) + \mathbf{y_i}) * \left(\begin{cases} 1 & \text{if } (i,j) + \mathbf{y_i} \in E \\ 0 & \text{otherwise} \end{cases} \right) \right) \%(\mathbf{max}\ S{+}1).$$

With this update function, consider the following initial configuration (of the cells of interest) $\mathbf{C}_0$ and configuration $\mathbf{C}_1$ at the next time instance $t = 1$:

$$\mathbf{C}_0 = \begin{bmatrix} 0 & 5 & 0 & 0 \\ 15 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \quad \mathbf{C}_1 = \begin{bmatrix} 0 & 5 & 5 & 5 \\ 15 & 0 & 0 & 15 \\ 15 & 0 & 0 & 0 \\ 15 & 5 & 0 & 0 \end{bmatrix}.$$

Note that in the introduced model, $\mathbf{C}_j$ configuration represents the adjacency matrix of the graph corresponding to the static network instance for the parameter p_j. We can derive various CA models for various parameters or we can also consider a composite CA, if needed. Furthermore, as an example, we will focus on the CA model introduced for the bandwidth parameter.

As mentioned above, it is interesting to use the CA in order to obtain network parameter patterns of interest. For instance, in order to test and validate novel network implementations it is desirable to set the environment of these implementations to different states. Moreover, not only in different states but, in order to stress test applications, to change them with high frequency and low predictability. For testbeds based on emulations this is perhaps the most desirable feature.

As a particular example, consider the automaton as described before with the configurations as previously mentioned ($c_0(1,2) = 5$ and $c_0(2,1) = 15$). The bandwidth evolution is shown in Figure 2.

Note that no bandwidth (zero) is represented by the color white, and the highest intensity of bandwidth (255) is represented by a solid black color (the darker the more bandwidth is available on the link, ranging in different tones of gray).

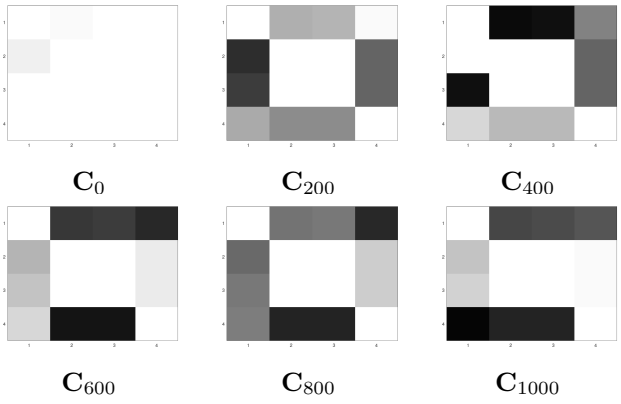

Fig. 2. CA evolution for a sparse initial configuration

Notice that our model is flexible and an easy way to accelerate the frequency of change in the dynamic-link parameter to create a less sparse initial configuration. For a randomly chosen initial configuration, the same cellular automaton yields the pattern shown in Figure 3. We furthermore discuss some advantages of the introduced model.

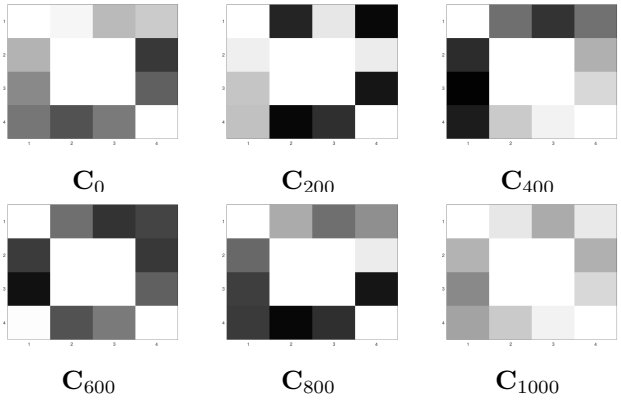

Fig. 3. CA evolution for a random initial configuration

Discussion – On using cellular automata simulation for parameter evolution verification Cellular automata can be useful not only as a proper model for dynamic-link parameter evolution but, also through the use of simulation, different properties of the model can be guaranteed. For instance, consider the update function as previously discussed. The bandwidth of all neighbouring nodes is added and then, the modulo operator is applied. Notice how this function can yield a configuration with value zero for all nodes in the cellular automaton. In this case, the bandwidth is always zero, from that point onward. In a real scenario, it means that the network will never pass any more traffic. For this reason, it is interesting to verify that the cellular automaton configuration holds certain properties during the first τ steps. Algorithm 1 describes the process for this verification. As an example, consider the predicate $\mathbf{C}_t \neq \mathbf{0}_{|V|\times|V|}$ (the zero squared matrix of size $|V|$), and $\tau = 1000$. Using the aforementioned algorithm, we can guarantee that during this amount of steps the bandwidth never gets in a "deadlock" configuration.

Algorithm 1: Parameter evolution verification

input : A dynamic-link network graph $G = \langle V, E, p_1, \ldots, p_k \rangle$, a cellular automaton model for parameter p_j, $\mathcal{A}_{p_j} = \langle 2, S, N, f \rangle$, a predicate over the configuration of the CA $\pi(\mathbf{C})$, an initial configuration c_0, and τ the number of steps

output: TRUE or FALSE

Set $c_t \leftarrow c_0$

Set $\mathbf{C}_t \leftarrow \mathbf{0}_{|V|\times|V|}$

foreach $e \in E$ **do**

  Set $\mathbf{C}_{t_e} \leftarrow c_t(e)$// Set the matrix of the cells of interest according to the current configuration

for $i \leftarrow 1; i \leq \tau; i \leftarrow i + 1$ **do**

  if $\pi(\mathbf{C}_t) =$FALSE **then**

    return FALSE

  foreach $e \in E$ **do**

    Set $c_{t+1}(e) \leftarrow f(c_t(e + \mathbf{x_1}), \ldots c_t(e + \mathbf{x_m}))$, where $N = \langle \mathbf{x_1}, \ldots, \mathbf{x_m} \rangle$

    Set $\mathbf{C}_{t_e} \leftarrow c_{t+1}(e)$// Set the new values for the cells of interest

  Set $c_t \leftarrow c_{t+1}$

return TRUE

Discussion – On the simulation to emulation performance ratio As previously mentioned, one of the objectives of this work is to find interesting patterns to be able to test different network components at emulation time. Nonetheless, it is difficult to launch an emulation and realizing after many minutes or even hours that the result is not as expected. Furthermore, repeating this process many times to find one interesting configuration seems not viable. For this reason, we turned our attention to the model which is easy to simulate and once having *interesting* initial configurations, update functions or both, we can inject them into the emulation testbed. But, how fast is emulation as compared to simulation? Figure 4 shows the running time of both, the application of the bandwidth change in our dynamic-link network emulator [27], and the CA-based simulation time. As can be seen, the simulation is quite advantageous, keeping the running times low, even as the size of the networks grow.

V. Conclusion

In this work, we have presented a cellular automata based approach for dynamic-link network simulation. The proposed method allows to properly model the evolution of different network parameters. Several dynamic-link network parameters (bandwidth, delay) can be modeled with our formalism, and furthermore, we can guarantee that certain properties hold over this evolution for τ time steps.

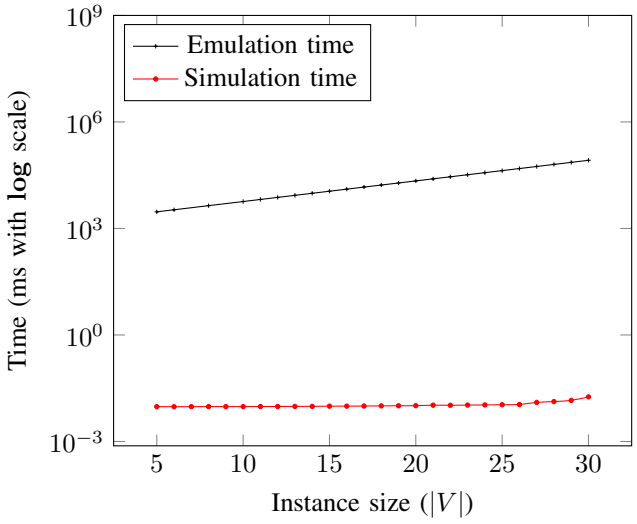

Fig. 4. Simulation vs emulation performance

As for future work, we consider searching for update functions that can be of particular interest. For example, it is known that certain update functions can yield pseudo-random configurations with a large period. This is interesting for testing novel implementations as the overall configurations can be done in difficult-to-predict settings. Machine learning strategies and stochastic properties are foreseen to be taken into account in the future, for the update functions' generation. Likewise, we are interested in update functions that can vary at different frequencies. Additionally, we envision to continue our research in predicate verification over the cellular automata; particularly, with an exact approach, and without the need of simulation. It is also interesting to develop a dynamic-link parameter evolution tool that allows using any pattern that can be expressed as a computable function. Finally, it is interesting to consider incorporating our proposed solution into existing network emulators.

References

[1] B. Deng, C. Jiang, H. Yao, S. Guo, and S. Zhao, "The next generation heterogeneous satellite communication networks: Integration of resource management and deep reinforcement learning," *IEEE Wireless Communications*, vol. 27, no. 2, pp. 105–111, 2019.

[2] Q. Shan, "Testing methods of computer software," in *2020 International Conference on Data Processing Techniques and Applications for Cyber-Physical Systems*. Springer, 2021, pp. 231–237.

[3] J. Lai, J. Tian, D. Jiang, J. Sun, and K. Zhang, "Network emulation as a service (neaas): Towards a cloud-based network emulation platform," in *Simulation Tools and Techniques*, H. Song and D. Jiang, Eds. Cham: Springer International Publishing, 2019, pp. 508–517.

[4] S. Gandhi, R. K. Singh *et al.*, "Design and development of dynamic satellite link emulator with experimental validation," in *2021 12th International Conference on Computing Communication and Networking Technologies (ICCCNT)*. IEEE, 2021, pp. 1–6.

[5] J. v. Neumann, "Theory of self-reproducing automata," *Edited by Arthur W. Burks*, 1966.

[6] N. V. Menshutina, A. V. Kolnoochenko, and E. A. Lebedev, "Cellular automata in chemistry and chemical engineering," *Annual Review of Chemical and Biomolecular Engineering*, vol. 11, pp. 87–108, 2020.

[7] P. Sala, J. Lehmann, T. Rakovszky, and F. Pollmann, "Dynamics in systems with modulated symmetries," *Physical Review Letters*, vol. 129, no. 17, p. 170601, 2022.

[8] S. Wolfram, "Cellular automata as models of complexity," *Nature*, vol. 311, no. 5985, pp. 419–424, 1984.

[9] F. Liu, M. Heiner, and D. Gilbert, "Hybrid modelling of biological systems: current progress and future prospects," *Briefings in Bioinformatics*, vol. 23, no. 3, p. bbac081, 2022.

[10] A. Chakraborty, S. Sikder, H. Omrani, and J. Teller, "Cellular automata in modeling and predicting urban densification: Revisiting the literature since 1971," *Land*, vol. 11, no. 7, p. 1113, 2022.

[11] M. Gardner, "The fantastic combinations of jhon conway's new solitaire game'life," *Sc. Am.*, vol. 223, pp. 20–123, 1970.

[12] C. Bays, "Introduction to cellular automata and conway's game of life," in *Game of Life Cellular Automata*. Springer, 2010, pp. 1–7.

[13] J. J. Kari, *Basic Concepts of Cellular Automata*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 3–24.

[14] E. Petersen, J. López, N. Kushik, C. Poletti, and D. Zeghlache, "On using smt-solvers for modeling and verifying dynamic network emulators: (work in progress)," in *19th IEEE International Symposium on Network Computing and Applications, NCA 2020, Cambridge, MA, USA, November 24-27, 2020*. IEEE, 2020, pp. 1–3.

[15] M. Khanjary, "Cellular learning automata: Review and future trend," *Computational Vision and Bio-Inspired Computing*, pp. 229–238, 2022.

[16] P. Sen and D. Barman, "Cellular automata based model for finding minimum spanning tree in wireless sensor networks," in *Asian Symposium on Cellular Automata Technology*. Springer, 2022, pp. 205–220.

[17] X. Fu, X. He, and Y. Yang, "Invulnerability analysis of wireless sensor networks based on cellular automata," in *2020 IEEE International Conference on Human-Machine Systems (ICHMS)*. IEEE, 2020, pp. 1–4.

[18] E. Heidari, A. Movaghar, H. Motameni, and E. Homayun, "Routing in internet of things using cellular automata," in *International Conference on Innovative Computing and Communications*. Springer, 2021, pp. 875–884.

[19] R. Hoffmann, D. Désérable, and F. Seredyński, "Cellular automata rules solving the wireless sensor network coverage problem," *Natural Computing*, vol. 21, no. 3, pp. 417–447, 2022.

[20] A. Tretyakova, F. Seredynski, and P. Bouvry, "Graph cellular automata approach to the maximum lifetime coverage problem in wireless sensor networks," *Simulation*, vol. 92, no. 2, pp. 153–164, 2016.

[21] M. Tavanpour, B. U. Kazi, and G. Wainer, "Discrete event systems specifications modelling and simulation of wireless networking applications," *Journal of Simulation*, vol. 16, no. 1, pp. 1–25, 2022. [Online]. Available: https://doi.org/10.1080/17477778.2020.1750313

[22] S. Choudhury, "Cellular automata and wireless sensor networks," in *Emergent Computation*. Springer, 2017, pp. 321–335.

[23] E. H. S. Cardoso, J. P. L. De Araújo, S. V. De Carvalho, N. Vijaykumar, and C. R. L. Francês, "Novel multilayered cellular automata for flying cells positioning on 5g cellular self-organising networks," *IEEE Access*, vol. 8, pp. 227 076–227 099, 2020.

[24] M. H. Hassan and R. C. Muniyandi, "An improved hybrid technique for energy and delay routing in mobile ad-hoc networks," *International Journal of Applied Engineering Research*, vol. 12, no. 1, pp. 134–139, 2017.

[25] M.-A. I. Tsompanas, N. I. Dourvas, K. Ioannidis, G. C. Sirakoulis, R. Hoffmann, and A. Adamatzky, "Cellular automata applications in shortest path problem," in *Shortest Path Solvers. From Software to Wetware*. Springer, 2018, pp. 199–237.

[26] T. Plénet, S. El Yacoubi, C. Raïevsky, and L. Lefèvre, "Observability of affine cellular automaton through mobile sensors," in *International conference on cellular automata for research and industry*. Springer, 2020, pp. 36–45.

[27] E. Petersen, J. López, N. Kushik, C. Poletti, and D. Zeghlache, "Dynamic link network emulation: A model-based design," in *Proceedings of the 17th International Conference on Evaluation of Novel Approaches to Software Engineering, ENASE 2022, Online Streaming, April 25-26, 2022*, H. Kaindl, M. Mannion, and L. A. Maciaszek, Eds. SCITEPRESS, 2022, pp. 536–543.