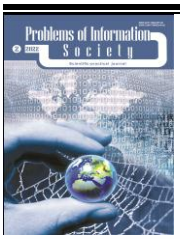

www.jpis.az

13 (2)
2022

Robust massive parallel information processing environments in biology and medicine: case study

Jiří Kroc

Charles University in Prague, Ovocný trh 560/5, 116 36 Praha 1, The Czech Republic

dr.j.kroc@gmail.com

orcid.org/0000-0002-9662-7646

ARTICLE INFO
http://doi.org/10.25045/jpis.v13.i2.02

Article history:
Received 8 February 2022
Received in revised form
11 Aprel 2022
Accepted 13 June 2022

Keywords:
Complex Systems
Cellular Automata
Emergence
Robustness
Artificial Life

ABSTRACT

The current frontiers in the description and simulation of advanced physical and biological phenomena observed within all scientific disciplines are pointing toward the importance of the development of robust mathematical descriptions that are error resilient. Complexity research is lacking deeper knowledge of the design methodology of processes that are capable to recreate the robustness, which is going to be studied on massive-parallel computations (MPCs) implemented by cellular automata (CA). A simple, two-state cellular automaton having an extremely simple updating rule, which was created by John H. Conway called the 'Game of Life' (GoL) is applied to simulate and logic gate using emergents. This is followed by simulations of a robust, generalized GoL, which works with nine states instead of two, that is called R-GoL (open-source). extra states enable higher intercellular communication. The logic gate is simulated by the GoL. It is destroyed by injection of random faulty evaluations with even the smallest probability. The simulations of the R-GoL are initiated with random values. several types of emergent structures, which are robust to injection of random errors, are observed for different setups of the R-GoL rule. The GoL is not robust. The R-GoL is capable to create and maintain oscillating, emergent structures that are robust under constant injection of random, faulty evaluations with up to 1% of errors. The R-GoL express long-range synchronization, which is together with robustness facilitated by designed intercellular communication.

1. Introduction

The main goal of this study is to open novel research vistas. The current computers and computational methods applied in science and technology are employing sophisticated approaches that are not computationally robust (error-resistant). Contrary to biological systems, which are all working with robust emergents. The expected outcome of this study is to explore uncharted areas of complex systems (CSs), which will eventually lead to robust computers and computational methods. Such methods are studied on a specific class of cellular automata (CA), where robustness of emergent structures is clearly demonstrated. The emergents generated by the 'Game of Life' (GoL) will be shown not to be error resilient. The generalized, robust GoL will demonstrate to maintain emergents despite the injection of 1% of random evaluation errors. In the rest of the introduction, the relevant information about CSs is briefly overviewed.

Why are complex systems important? The current understanding of information processing, which is observed within a wide range of natural phenomena, that are operating within non-living and living systems, is directly pointing towards the presence of massive-parallel computations (MPCs) that are operating within those systems. Despite the fact that various MPCs are observed within completely different media, based on different physical and biochemical processes, they possess common features. MPCs and their information processing are naturally calling for CSs

descriptions, as those provide the most suitable toolkit capable to describe and predict behavior of such phenomena. MPCs-based models are creating a subset of all CSs models.

CS-models are having the potential to computationally describe all naturally observed phenomena that express: self-organization, emergence, self-replication [1, 2], self-repair (aka healing), and many other phenomena. Additionally, CS based descriptions provide means to experiment with various potentially harmful scenarios that cannot be tested in reality (as in many social, biological, and biomedical applications). This can help to develop both still lacking and needed future theories of CSs, and simultaneously, safely explore ethically forbidden areas of experimental research.

What are the means of CSs description? The scientific area dealing with mathematical and computational descriptions of CSs is undergoing a constant, often explosive development with many open, challenging frontiers spanning across all scientific disciplines. We know that step-by-step development of applied mathematical descriptions is constantly pushing the envelope of our descriptive capabilities outwards into the unknown. The multidisciplinary nature of CS research unavoidably raises the confusion and inability of researchers originating within different research areas to find efficient communication means, as the fields are constantly morphing.

Hence, there is a need to periodically review those novel achievements by distilling them across all scientific disciplines. Those reviews of theoretical and computational tools can be easily applied within all other disciplines. Such cross-fertilization of disciplines can be achieved by defining the common ground to all of them. The best known, highly-recommendable tool providing such common ground is provided by CA [3-7] and their movable counterpart that are called agent-based models (ABM) [8].

Currently there are existing several generic computational methods applied to the description of CSs phenomena along with many of their variants that are known under various names, see Table 1:

Table 1: A brief list of existing generic classes of descriptive methods used to quantitatively and qualitatively study complex systems [5].

Method Name	*Brief Description*	*References*
CA: cellular automaton	*A lattice of constituting elements (cells) that are interacting with their neighbors according to a common evolution rule.*	*Illachinsky, 2001; Kroc et al. 2019*
ABM: agent-based model	*Same as above with moving agents instead of fixed automata.*	*Illachinsky, 2004*
CSM: complex systems measure	*CSMs are utilizing the concept of Boltzmann's and Shannon's entropies and their generalizations.*	*Kroc et al., 2020*
SM: statistical method	*Simple and advanced statistical methods.*	*Kroc et al., 2020*
ML: machine learning	*ML methods often take features, which are distilled from the original CS system using SM, as the input.*	*Kroc et al., 2020*
AI: artificial intelligence	*Incorporate neuronal networks, other perceptors, and ML.*	*Kroc et al., 2020*

All those methods are applying various approaches. Both, CAs and ABM methods apply a simple approach that is based on utilization of local, mutual interactions of their low-level, systemic, constituting elements with a restricted number of their neighbors according to a uniform set of interaction rules. Some mentioned methods' features can be generalized in advanced approaches. CA work above a fixed lattice of their constituting elements (called cells) whereas ABM work with movable constituting elements (called agents).

Statistical methods (SM) alone are typically found insufficient in description and capturing the basic properties of complex systems; nevertheless, they are often providing valuable features used as inputs to artificial intelligence (AI) and machine learning methods (ML) methods. Finally, complex systems measures (CSM) have the capability to capture subtle trends, which are operating inside of huge complex systems, that are otherwise inaccessible to simple and advanced SMs.

AI & ML methods are possessing the capability to discern even more details within CSs, which are either measured by SMs or CSMs. The major disadvantage of AI & ML methods is laying in the fact that in the majority of cases they are utilizing black-box methods that are lacking human interpretable forms.

Hidden history of CSs descriptions. The latest CSs descriptions are standing on the shoulders of their older, successful predecessors developed in physics & mathematics: dynamical systems,

synergetics, statistical physics, and fractal theory. The early beginning of CSs are associated with the development of dynamical systems, which had been initiated by the three-body problem in the celestial mechanics of by gravity bound bodies, that was defined and theoretically studied by Henry Poincaré in the end of the 19th century. This research clearly demonstrated that movements of three and more celestial bodies described by deterministic equations acquire long-term, unpredictable trajectories. Even the slightest change in the initial conditions leads to a dramatic change in the future paths of the involved bodies. As another example, later in 50s of the 20th century the Lorenz weather model [9] demonstrated its extreme sensitivity to the initial conditions that yielded the term 'butterfly-effect'. Both approaches led to a development of dynamic systems theory (DS) and relevant numerical methods.

Going back in time, yet another stream of thoughts that eventually led to the foundation of CSs theory appeared in the Boltzmann theory of gases—developed around the mid of the 19th century—, which founded the statistical physics and the notion of entropy. Later, entropy becomes the fundamental tool in measuring CSs properties. Boltzmann's approach stirred the contemporary establishment within physics because it, for the first time, had proven that a set of fully deterministically defined movements of systemic parts (atoms) produce unpredictable statistical behavior of said atoms, which went against one of the main physics' dogmas. Simultaneously, Boltzmann's approach led to the development of the mathematical description of entropy as the notion dealing with unavailable energy [10, 11, 12, 13] that was later generalized by Shannon [14, 15] within the scope of information theory.

Both streams of thought and theories eventually led to our current understanding of CSs descriptions, which are representing a vital paradigm, that are very suitable and sufficiently flexible to describe natural phenomena observed within the whole spectrum of scientific fields. The onset of the massive use of personal computers enabled every researcher to employ CS-models in his own research.

Importance of Complex Systems in Biology. CSs had been demonstrating their usefulness in description of fairly complex phenomena observed in biology—and consequently in medicine—that all was accomplished using relatively simple models. Those models require preliminary knowledge of complex systems modeling. Therefore, there is a need to start with some simple, prototype model, which is easy to grasp, that simplify understanding of all modeling steps: design, algorithm, programming, and refining of all the previous steps until success is reached. See the GoL cellular automaton discussed in this text [16, 17], which is expressing emergent behavior, as such prototypical example. After digesting all necessary steps in the CS-model's design, it becomes much easier to understand and design more complicated CS-models.

Why are CSs better in the description of biological phenomena? CSs posses several key features that are providing biological models formulated by them crucial advantages in comparison to other descriptive approaches (like differential equations, or statistics- and probability-based ones). Those advantages are massive parallelism, independence of their constituting parts, locality of mutual interactions of constituting parts, local information storage, and uniform evolving/updating rule through the whole system, see e.g. [18] for real biological and physical examples. Simultaneously, we know that many of currently used CS-models are unable to reproduce robustness of biological systems, it means, their ability to resist perturbations and the capability to recover from them. This task is going to be addressed in the second method of this paper.

We know that in many situations, CSs and MPC models are the only way to model biological phenomena. Examples encompass ecosystems, immune system, cytoskeleton and exoskeleton of the cells, tissues, morphological growth (tooth, skin), and many other [5].

The paper outline. The introduction describes CSs, their history, and their importance in biology. The introduction into CAs follows. Two methods are defined: the GoL, and its generalized, robust GoL (r-GoL) variant. This is followed by results, which presents the GoL simulation of AND logic gate using emergents, its by-errors destroyed variant, and three r-GoL simulations with different levels of injected errors, where emergents are demonstrated to be resilient against injection of errors. All is finalized by the conclusion.

2. Cellular automata

Cellular Automata (CAs) play the prototyping role within CSs modeling. CAs are operating above a lattice of elements (squares, triangles, and hexagons in 2D; cubes in 3D) that are called cells. Each cell has a list of its close neighbors, which total number is much lower than the total number of cells, that is uniform throughout the whole lattice. Cells are updated synchronously according to the identical

rule that defines the state(s) of the cell in the next step.

Despite the simplicity of this computational approach, there is existing the endless complexity within it. Actually, when a CA-model of natural phenomena is designed, the author of it usually face the problem of excessively wild behavior of the CA-model until all settles down to the final version of the model that becomes more nicely 'behaving' [4]. This effect is well-known to all CA modelers and is caused by the tendency of CA-models to become chaotic. Roughly said, it is known that complex rules of CA are located somewhere between the periodic behavior and chaotic one.

The expressiveness of the CAs approach is so high that it is known that most of, if not all, computable systems can be described using CAs. This had been proven in the 'Game of Life' [16, 17], which can principally simulate anything that we are currently able to compute by current computers [19]. See the animation of AND logic gate using glider guns emitting colliding streams of gliders, which is provided at the software [20, 21, 22] and snapshots in Fig. 1. Similarly, OR and NOT logic gates can be emulated in the GoL, see animations in [20] and links there. To provide an example of the flexibility of the CA method, the evaluation of partial differential equations where a differentiation scheme is implemented by a rule within a CA is shown in [23, 24].

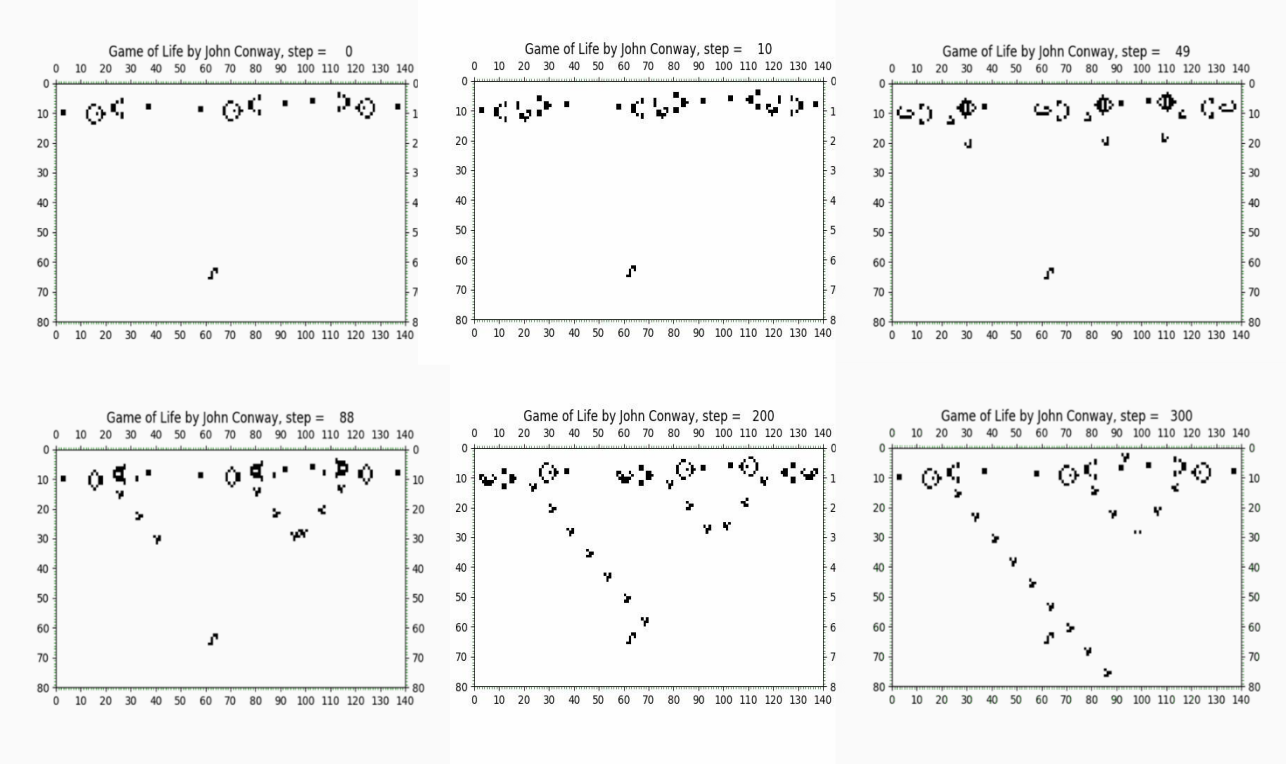

Fig. 1. A sequence of snapshots, which is depicting the AND logic gate, that is taken at steps 0, 10, 49, 88, 200, and 300. Animations of AND, OR, and NOT logic gates are available at the link [20], and related data.

3. Methods

Two methods had been studied in this paper: the 'Game of Life', called GoL, cellular automaton employing massively parallel, emergent computations, and its robust generalization called r-GoL (robust GoL) which emergents wisthand injection of faulty evaluations.

3.1 Method of John H. Conway's 'Game

John Horton Conway created a simple CS model of alive-like entities called the 'Game of Life' (GoL)– (sometimes called simply as 'Life'), which is implemented using a CA, that led mathematicians and computer enthusiasts to endless studies of complexity arising from simplicity. The GoL works above a two-dimensional lattice of squares with periodic boundary conditions (left and right edges, and top and down are connected) where each cell updates its own state according to the states of its eight neighbors and itself. Only the total number of alive neighbors is counted, which is inserted into the updating rule; hence, it is called the totalistic rule. The GoL has only two states: alive (one, black) and dead (zero, white). The number of cells within its neighborhood in the state of one is counted, and the following rule is applied [20], see Table 2. Different cases of the rule can be related to survival (I.a & I.b), the birth of new life (II.), and death due to insufficient cooperation

(III.a) and overpopulation (III.b). Surprisingly, such a very simple rule is providing means to create very complex emergent structures within the simulated lattice.

Table 2: A list of all inputs that are affecting the evolution of each cell's state within the lattice in the GoL is provided bellow along with their biological interpretation. This rule is having just the right combination of survival and birth rates to express complex behavior [20].

Case	Input configuration	Output value	Biological interpretation
I.a	cell(t) = 1 & neighbors = 2	cell(t+1) = 1	surviving
I.b	cell(t) = 1 & neighbors = 3	cell(t+1) = 1	surviving
II.	cell(t) = 0 & neighbors = 3	cell(t+1) = 1	birth
III.a	neighbors <= 2 (except the Case I.a)	cell(t+1) = 0	death, insufficient cooperation
III.b	neighbors > 3	cell(t+1) = 0	death, overpopulation

3.2 Method of robust generalization of the 'Game of Life': r-GoL

As will be shown, the GoL is not robust due to its extreme sensitivity to random noise. One of the possible lines of attack to make it more robust is to increase the number of cell states from two to some higher value. It was decided to use ten states (0, 1, 2, …, 9) as they can be equally spread among the cell and its eight neighbors, which brings a higher flexibility into the CA-rule governing the evolution. Changes of the CA-rule are summarized in the following list:

- The state of each cell is laying within the interval <0, 9>. The zero value is expected as the ground state, which is followed by nine higher levels of excitation.
- A threshold that is stating the value above which the cell becomes alive is defined as T_A.
- A threshold that represents silencing of an overexcited cell is defined as T_D.

When those generalizations are implemented into the GoL, an improved, more robust algorithm is defined that is called the r-GoL, see Algorithm 1 [27].

```
neighbor_Alive()
   counting alive neighbors within Moore neighborhood
using thresholds T_A, T_D

if neighbor_Alive() == 3
   cell = 9
else if (neighbor_Alive() == 2) and (cell >=
lower_threshold) and (cell <= upper_threshold)
   cell = 9
else
   if (rand_number() < 0.01) ### Random Swapping of
cell's value with p = 0.01
         cell = 5
      else
         cell = 0
```

Algorithm 1: The algorithm of the r-GoL is working with 10 states. A not shown sub-step applying the distribution of states to neighboring cells from an excited cell is applied just right before the sub-step shown here, which is applying the r-GoL algorithm (T_A = lower_threshold; T_D = upper_threshold). When random noise having the probability of 0.01% or 1% is introduced, see Figs. 5 and 6, or 7 and 8, respectively. Without this noise, see Figs. 3 and 4.

4. Results and discussion

Both methods had been tested and evaluated as documented in this section. Robustness of emergent structures observed within r-GoL is clearly demonstrated, see the open-source codes and animations of both algorithms in [20,27].

4.1 John H. Conway's 'Game of Life'

During over 50 years of studies, a plethora of emergent structures was discovered [25]: gliders, glider guns, ships, blinkers, eaters, and logic operations such as AND, NOT, OR, NAND, etc. [20].

The GoL was proved to be Turing equivalent [19]. In other words, it is proven that the GoL is capable to evaluate all that is computable by Turing machine (current von Neumann architecture based computers). In theory, it is possible to build a whole computer within the GoL using the above AND, OR, and NOT logic gates where gliders serve as 'electrons' that carry on bitwise information: the presence of a glider is equal to the state one, and its absence is equal to the state zero. There are attempts to recreate a processor using the GoL logic gates based on glider guns and gliders, see e.g. [26].

Emergent structures. Emergents in one or other form are operating in the majority of observed natural phenomena. Nevertheless, most of us are

unaware of them. A list of simple examples of emergents encompass temperature and pressure of gases & liquids, physical properties of crystalline structures as strength, properties of proteins and other bio-structures, cells, tissues, organs, bodies, ant & insect colonies, societies, and ecosystems.

To become familiar with emergents, to raise our own capability of their detection in any observed natural phenomenon, we need both generic examples and understanding of their ways of becoming into the existence. Both aspects are provided by the GoL. By studying, playing with, and modifying the GoL, everyone can gain insights into the generation of emergent structures, use the open-source Python software [20].

This contribution helps to start to uncover the beauty of emergent structures that are existing and operating in virtual environments, arising from and through mutual interactions of large numbers of simple processes that are storing information locally. Emergents are arising from two opposite processes: creation, and destruction, or multiplication and erasing, respectively. Only when those opposing processes are balanced, they give rise to emergents.

The GoL create new living cells only when there are three alive cells around a dead one. When there are more alive cells, the cell stay or become dead. It is caused by the competition for resources due to overcrowding. Alternatively, when there are two or less alive cells in the neighborhood, the cell will stay or become dead; with a single exception: an alive cell having two alive neighbors stay alive. This is caused by lack of cooperation among neighboring cells.

Only when this subtle balance between creation and destruction is maintained in the whole system, emergents are created and sustained. Each emergent in the GoL has a topology, which is the key to its survival. Only certain topologies can survive indefinitely unless they get disrupted.

Effects of random Errors. Let us check the effect of random errors on the stability of the simulation when they are present in the evaluation of the GoL-rule with a certain probability. Even introduction of errors with as minuscule probability as $p \approx 0.0001$ are leading to a quick destruction of all emergents, see Fig. 2. Obviously, larger emergent structures are destroyed faster because the probability of a structure being affected by a random error roughly increases with the square of the structure's linear size.

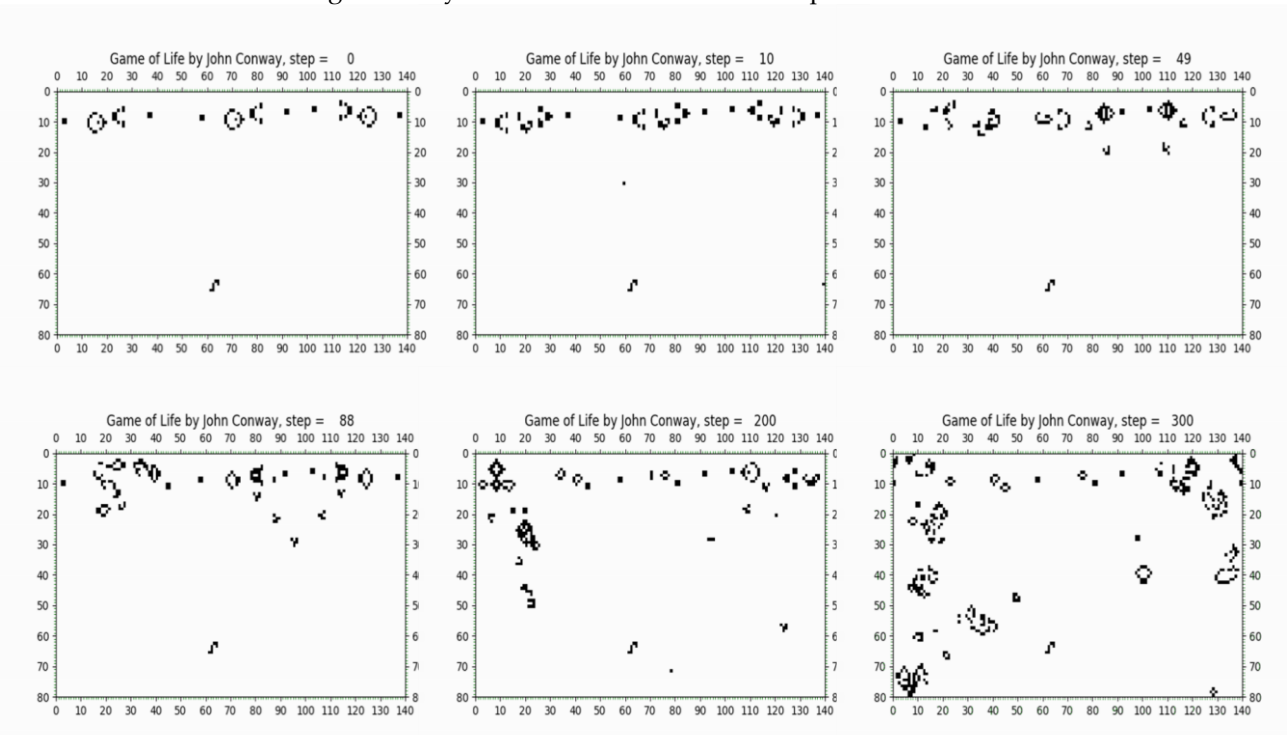

Fig. 2. The identical initial configuration of AND logic gate as in Fig. 1 is simulated with randomly occurring faulty evaluations (swaps). A randomly chosen cell with the probability p = 0.0001 get reversed its value. The sequence of snapshots taken at steps 0, 10, 49, 88, 200, and 300 that are depicting a glider gun emitting gliders, which are quickly disrupted and destroyed, see videos [20].

The random errors were created in the following way. A cell gets randomly swapped its value with a certain probability p, the evolution of the same initial topology as used in Fig. 1 was carried on and compared to the undisrupted GoL simulation. With decreasing probability p (0.1, 0.01, 0.001, and 0.0001), the effect of disruption of the GoL evolution gets only postponed to a more distant moment. Nevertheless, all simulations get disrupted even with the smallest p values. This brings us to the notion of robustness.

Robustness. In the previous section, we clearly demonstrated that the GoL posses an extremely low robustness against random disruptions realized by swapping of cells' value. By robustness, it is understood the ability of the simulation to recover from random changes of cells' values. A simulation expressing zero robustness is unable to recover from even a single random state change within emergents, which leads to the disappearance of those structures.

Hence, every computation based on the GoL or a similar type of simulations must be from the principle error free; otherwise, such simulations get easily disrupted and emergents are lost. This property of the GoL is rising a natural question: "Are there existing rules that lead to robust GoL-like simulations?" Let us look in this direction and explore rules that can be observed in physical and biological systems, which are both expressing robustness.

4.2 Results of robust generalization of 'Game of Life': r-GoL

During simulation of the r-GoL from a random initial condition, various static and oscillating structures having the period of two (blinkers) emerges/evolves after some transition period. The topology and behavior of those emergent structures differ from the original GoL substantially. Blinkers are existing within corralled regions where alternating patterns of equidistant points along with various alternating emergent structures called arrows are observed, see Figs. 3 and 4. Similar structures can be seen in Figs. 5-8. All emergents became persistent when the corral is stable, unless it gets disrupted by neighboring structures. All emergents arise quite naturally in the r-GoL simulations. Surprisingly, emergents are abundant. Moving structures or gliders were not observed; their potential existence is unknown at this moment.

It was observed that structures initiate their growth from the initial random configuration, followed by the growth phase, reach their maximal span, and then when they have an imperfect boundary called the corral, they shrink. Stability of all of those domains is defined by the stability of their boundaries. It is possible to manually define stable boundaries with its content (not done in this study). The spacing of alternating points within the stable domains is 2 cells horizontally and 2 cells vertically.

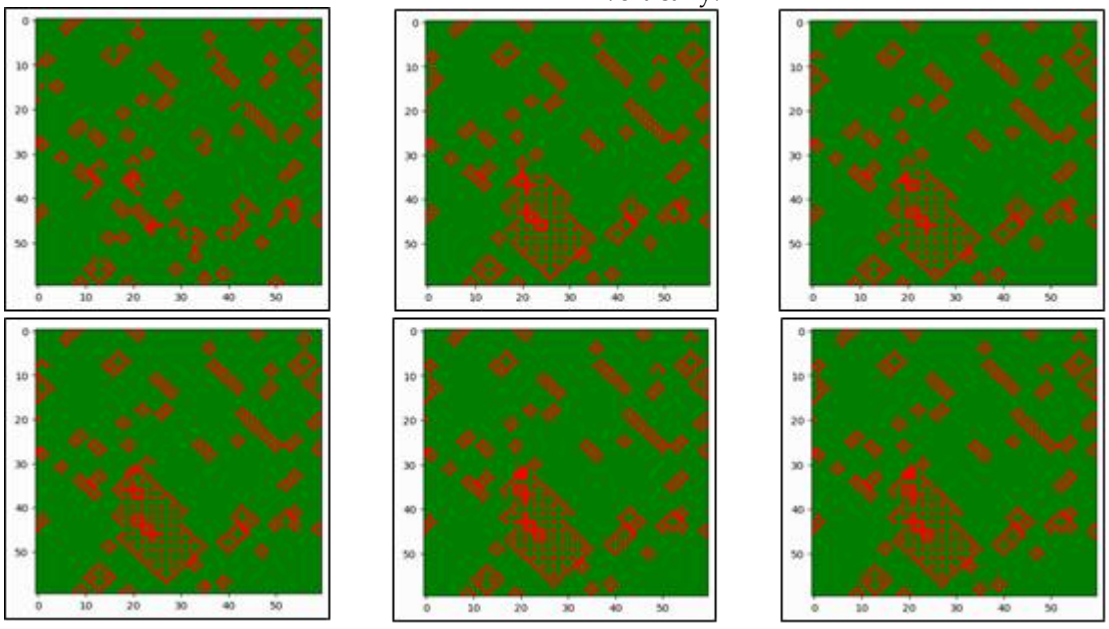

Fig. 3. A sequence of snapshots (left-right, top-down, same in all pictures) depicting the evolution of the random initial condition (random gen. PCG64(124)) as it progresses into a set of stable corralled domains of alternating patterns: points and arrows. Snapshots are taken at the times: 0, 35, 36, 100, 101, and 185, see [27].

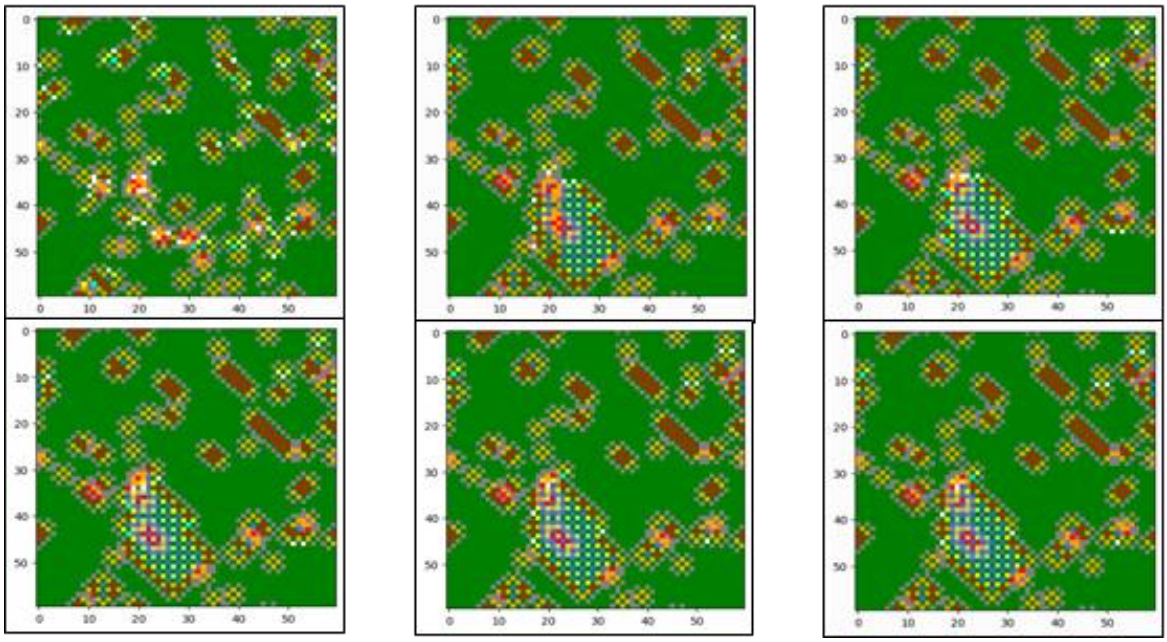

Fig. 4. Snapshots of the diffusion patterns evolution are shown for the identical initial conditions and times as in Fig. 3. Displayed sub-figures correspond one-to-one in both figures, see [27].

As already said, all large alternating structures are corralled. All well-defined emergent structures had been observed only inside the corralled areas. There had been observed minor alternating structures encompassing small number of excited cells located out of corrals. Interestingly, moving emergents were not observed.

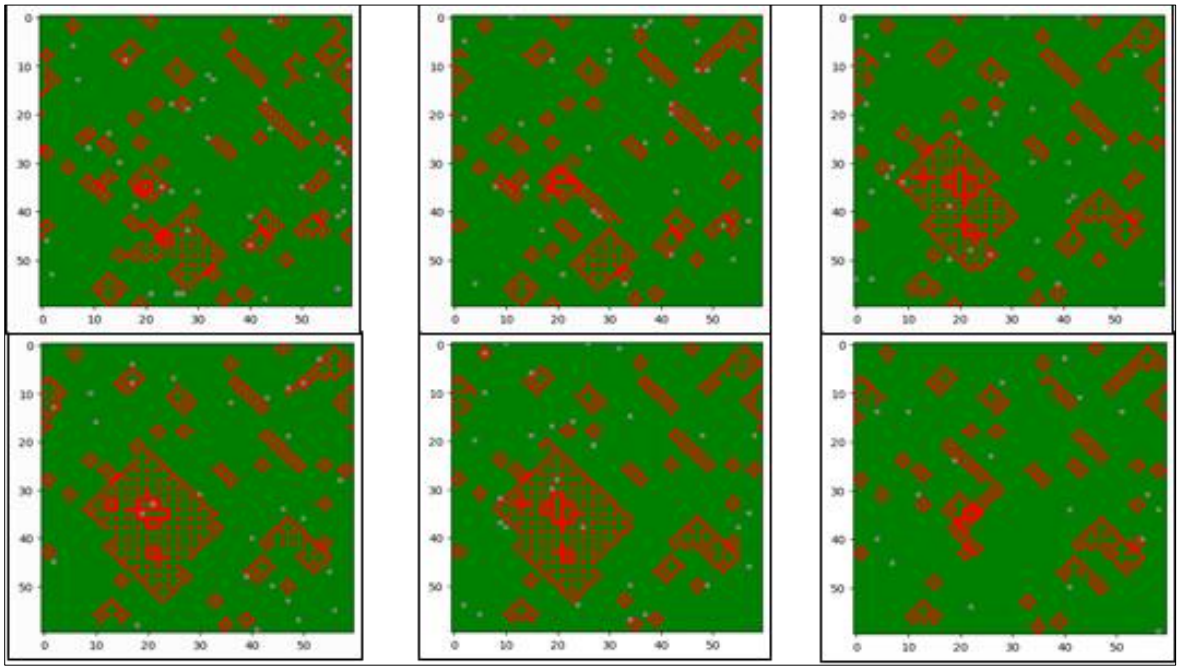

Fig.5. A sequence of snapshots depicting the evolution of the random initial condition (PCG64(124)) same as Fig. 3 where 1% of randomly chosen cells' values are randomly swapped between values 0 and 1. Snapshots are taken at the times: 21, 38, 74, 89, 90, and 196, see [27].

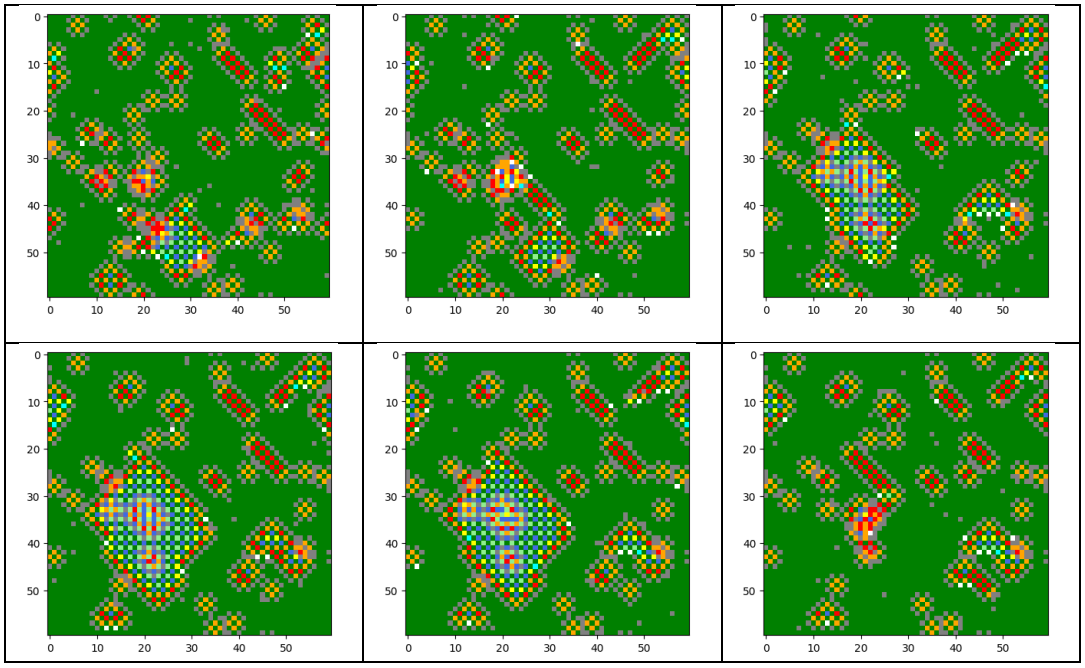

Fig.6. Snapshots of the evolution of diffusion patterns are shown for the identical initial conditions and times as in Fig. 5. Displayed sub-figures correspond one-to-one in both figures, see [27].

Figs. 5 and 6 differ from Figs. 3 and 4 only by swapping of randomly chosen cells' value between values of 1 and 0 in 1% of cases. Figs. 7 and 8 display the case when the values are swapped between values of 5 and 0 in 1% cases. All cases shown in Figs. 3 to 8 are using the cell-alive lower threshold = 2 and the cell-alive upper threshold = 8. When the cell-alive upper threshold = 9, nothing changes in the evolution. The cell-alive upper thresholds < 8 leads to the collapse of the simulations.

Therefore, there are two means of simulations disruption in the r-GoL that express a relatively high level of resilience: random disruption and change of thresholds when the cell is supposed to be alive. The emergent topologies are changed by applying random changes of cells' state, yet, simultaneously, novel emergents are qualitatively same as those existing in undisrupted simulations.

From the diffusion patterns, it is evident that static and alternating structures are rooted in static and alternating diffusion patterns, respectively.

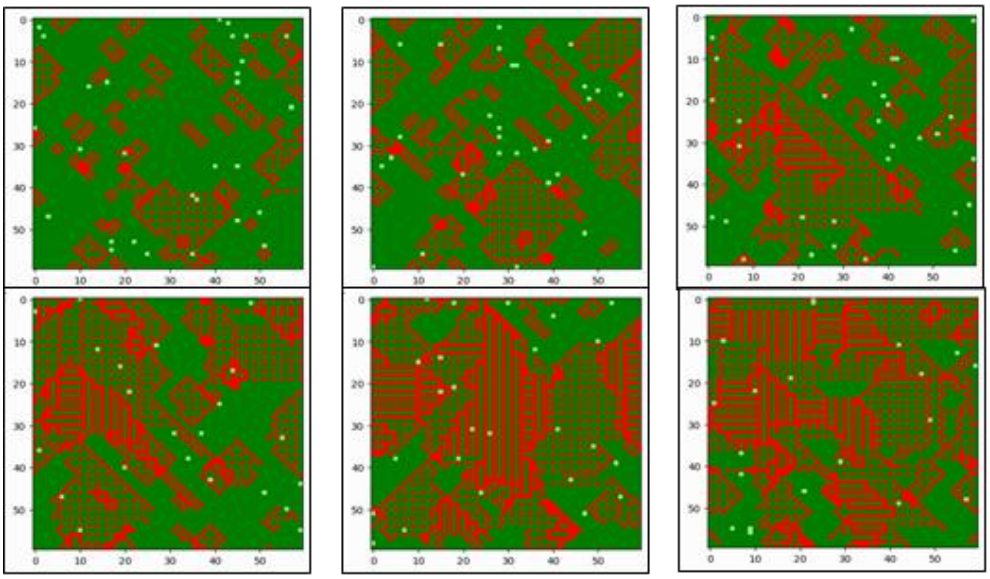

Fig.7: A sequence of snapshots depicting the evolution of the random initial conditions (PCG64(124)) same as Fig. 3 where 1% of randomly chosen cells' values are randomly swapped between values 0 and 5. Snapshots are taken at the times: 10, 25, 54, 82, 130, 189, see [27].

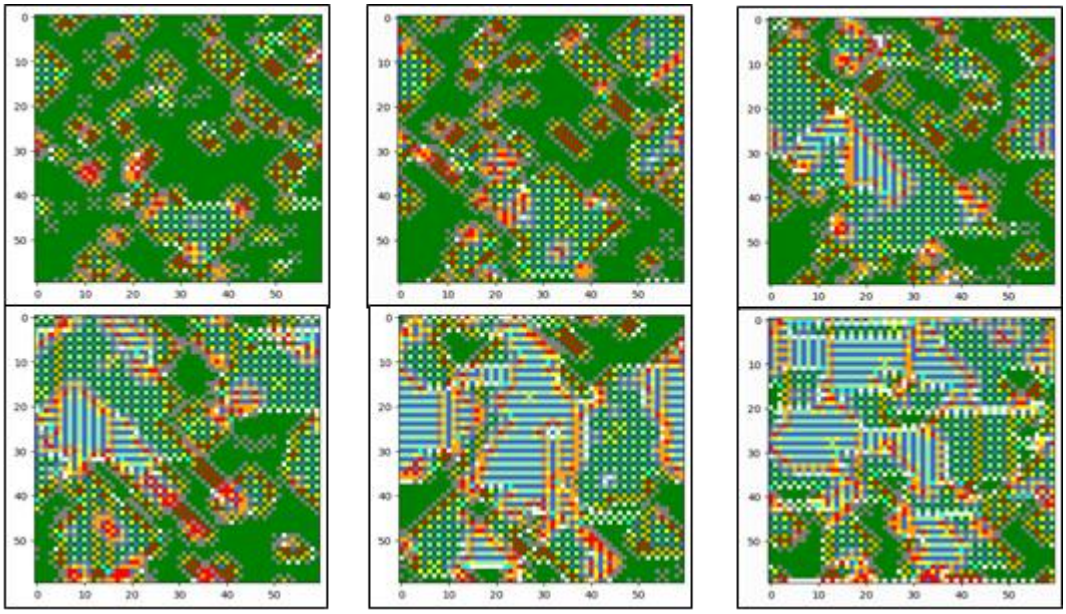

Fig.8. Snapshots of the evolution of diffusion patterns are shown for the identical initial conditions and times as in Fig. 7. Displayed sub-figures correspond one-to-one in both figures, see [27].

There are existing persistent, underlying diffusion patterns that are governing the evolution of the simulation, which remain invisible when we observe only the excited cells. It is understandable, as tracking all exited links among all cells is by orders of magnitude is more complicated than tracking excited cells themselves!

From this insight, it is easily seen that information fluxes within networks interconnecting cells are fundamental for the correct understanding and interpretation of the underlying dynamics of the observed excitation of emergent patterns.

When r-GoL is compared with real biologically observed excitatory networks that are observed within the brain, eye, heart, intestines, some striking similarities occur. It would be interesting to search for similarities between biological networks and the simplified model.

5. Conclusion

The paper demonstrated the existence of a robust generalization of the 'Game of Life' named the r-GoL, which is opening a promising area of algorithms, that can be applied in the design of the future, robust, massive parallel computational approaches and computers that will be increasingly resistant to random errors. The robustness of the r-GoL was demonstrated by applying varying levels of random changes of cells' states within different proportions of randomly chosen cells (0.0%, 0.01%, 1%) where the original GoL fully fails to maintain its emergents. The qualitative behavior of the emergents is robust, while the emergents' detailed topology is changing.

References

1. Von Neumann, J. (1966). Theory of Self-Reproducing Automata. Illinois: University of Illinois Press (edited and completed by A.W. Burks).
2. Langton, C. (1984). Self-reproduction on a Cellular Automaton. Physica D, 10, 135–144.
3. Ilachinski, A. (2001). Cellular Automata: A Discrete Universe, Singapore: World Scientific Pub.
4. Kroc, J., Jiménez-Morales, F., Guisado, J.L., Lemos, M.C. & Tkáč, J. (2019). Building efficient computational cellular automata models of complex systems: backgrounds, applications, results, software, and pathologies. Advances in Complex Systems 22(05) 1950013
5. Kroc, J., Balihar, K., Matejovic, M. (2020). Complex Systems and Their Use in Medicine: Concepts, Methods and Bio-Medical Applications. preprint, ResearchGate, doi: 10.13140/RG.2.2.29919.30887
6. Kroc, J., Sloot, P.M., & Hoekstra, A. (2010). Simulating Complex Systems by Cellular Automata (Understanding Complex Systems). Berlin, Heidelberg: Springer. DOI: 10.1007/978-3-642-12203-3_1
7. Kroc, J., Sloot, P.M., & Hoekstra, A. (2010). Introduction to Modeling of Complex Systems Using Cellular Automata. In Kroc, J., Sloot, P.M., & Hoekstra, A. Simulating Complex Systems by Cellular Automata (Understanding Complex Systems). Berlin, Heidelberg: Springer. DOI: 10.1007/978-3-642-12203-3_1
8. Ilachinski, A. (2004). Artificial War: Multiagent-based Simulation of Combat, Singapore: World Scientific Pub.
9. Lorentz, E.N. (1963). Deterministic nonperiodic flow. Journal of the Atmospheric Sciences 20(2) 130-141. doi: 10.1175/1520-0469(1963)020<0130:DNF>CO;2
10. Boltzmann, L. (1896). Vorlesungen über Gastheorie, vol. I. Leipzig:J.A. Barth. (1st ed).

11. Boltzmann, L. (1898). Vorlesungen über Gastheorie, vol. II. Leipzig:J.A. Barth. (1st ed).
12. Jaynes, E.T. (1965). Gibbs vs Boltzmann entropies. American Journal of Physics, 33(5) 391–398.
13. Ben-Naim, A. (2015). Information, Entropy, Life and the Universe: What We Know and What We Do Not Know. Singapore: World Scientific Pub.
14. Shannon, C. E. (1948). A mathematical theory of communication. The Bell System Technical Journal, 27(3), 379-423.
15. Shannon, C. E., & Weaver, W. (1998). The Mathematical Theory of Communication. Urbana: University of Illinois Press.
16. Gardner, M., (1970). The Fantastic Combinations of John Horton Conway's New Solitaire Game 'Life'. Scientific American 223(4) 120–123. DOI 10.1038/scientificamerican1070-120
17. Wikipedia. (Dec 7, 2021). Conway's Game of Life. Retrieved from Wikipedia
18. Adamatzky, A. (2019). A brief history of liquid computers. Philosophical Transactions of the Royal Society B: Biological Sciences 374(1774) 20180372.
19. Wainwright, R.T. (1974). Life is Universal!. In, Proceedings of the 7th Conference on Winter Simulation - Vol 2 (pp. 449–459). Washington, DC. Doi 10.1145/800290.811303
20. Kroc, J. (2021). The simplest Python program simulating a cellular automaton model of a complex system: the 'Game of Life'. ResearchGate, open-source software. Retrieved from Source Code GoL
21. Sundnes, J. (2010). Introduction to Scientific Programming with Python. (Simula SpringerBriefs on Computing) Cham: Springer. https://doi.org/10.1007/978-3-030-50356-7
22. Langtangen, H.P. (2016). A Primer on Scientific Programming with Python. (Texts in Computational Science and Engineering 6) Berlin, Heidelberg: Springer, DOI: 10.1007/978-3-662-49887-3
23. Toffoli, T. (1984). Cellular automata as an alternative to (rather than an approximation of) differential equations in modeling physics. Physica D, 10(1–2), 117-127.
24. Vichniac, G. (1984). Simulating Physics with Cellular Automata. Physica D, 10, 96–115.
25. Silver, S. (2018). Life Lexicon. Retrieved from Life Lexicon
26. Carlini, N. (2021). A Simple CPU on the Game of Life – part 4. Retrieved from CPU made of GoL
27. Kroc, J. (2022). Python program simulating cellular automaton r-GoL that represents robust generalization of 'Game of Life'. ResearchGate, open-source software. Retrieved from Source Code of r-GoL'