

WSPC BOOK SERIES IN UNCONVENTIONAL COMPUTING *Volume 5*

POST-APOCALYPTIC COMPUTING

Editor **Andrew Adamatzky**



 **World Scientific**

© 2025 World Scientific Publishing Company
https://doi.org/10.1142/9789811297144_fmatter

Contents

<i>Preface</i>	v
<i>About the Editor</i>	vii
1. The Clock of the Apocalypse: The Rhythms of Life and the Ending of Time <i>José Félix Costa and Paula Gouveia</i>	1
2. What to Compute Before an Apocalypse: Knowledge as Cryptocurrency at the End of Civilisation <i>Hector Zenil</i>	75
3. Evolutionary Techniques in Analyzing Thom's Catastrophes: Insights into Complex System Collapses <i>Ivan Zelinka</i>	93
4. Computing with Stones and Sticks <i>Jerzy Gorecki</i>	129
5. Computing Without Electronics <i>Andrew Adamatzky</i>	141
6. Post-Apocalyptic Computing and Technology Towards Computing from Natural Colloids and Micro-Fragments <i>A. Chiolerio, M. Crepaldi, D. Torazza, N. Raeisi Kheirabadi, and A. Adamatzky</i>	175

7.	Analogue Computation after Apocalypse <i>Bernd Ulmann</i>	201
8.	Computing with Clocks <i>Jonathan Edwards, Alex Yakovlev, and Simon O'Keefe</i>	213
9.	Bio-inspired Fault-tolerance in Electronic Systems <i>Martin Albrecht Trefzer</i>	229
10.	Before and after Big Crunch in a Reversible Discrete Cellular Universe <i>Kenichi Morita</i>	259
11.	Post-Apocalyptic Computing from Cellular Automata <i>Genaro J. Martínez, Andrew Adamatzky, and Guanrong Chen</i>	283
12.	Smoke Signals to Silicon: Unravelling the Threads of Emerging Computing Paradigms <i>Ioannis K. Chatzipaschalis, Iosif-Angelos Fyrigos, and Georgios Ch. Sirakoulis</i>	297
13.	Towards Computational Apocalypse: Computing with Minimal Resources and Earth-abundant Materials <i>Lulu Alluhaibi, Pier Luigi Gentili, Wiktor Głab, Ewelina Kowalewska, Sébastien Pecqueur, Anurag Pritam, Andrzej Sławek, and Konrad Szaciłowski</i>	323
14.	Challenges of Unconventional Computing: A Personal Perspective <i>Gari Owen</i>	369

Contents

xi

15. Self-decomputing: The Lack of Meaning Among Information 377
Jordi Vallverdú
16. A Metaphysical Approach to the Apocalypse to Come 389
Catarina Pombo Nabais
17. Listen 407
*Hilary Ritz, Esteban Montero, Flora Moon,
and Brandon Baylor*
18. The Buddha and Biomass 479
Andrew Schumann
19. Neosentience Production and A Set of
Conversations With ChatGPT Exploring
Speculative Post-Apocalyptic Questions 493
Bill Seaman
20. Domsday Machines Computer and Computing in
Cold-War Science Fiction 505
Stefan Höltgen
21. Always Already Post-Apocalyptic: On the History
of Computing 521
Jens Schröter

Chapter 1

Post-apocalyptic computing from cellular automata

Genaro J. Martínez^{1,2,3} & Andrew Adamatzky^{2,1} & Guanrong Chen^{3,1}

1. *Artificial Life Robotics Laboratory, Escuela Superior de Cómputo,
Instituto Politécnico Nacional, México.*

gjuarezm@ipn.mx

2. *Unconventional Computing Laboratory, University of the West of
England, Bristol, United Kingdom.*

andrew.adamatzky@uwe.ac.uk

3. *Centre for Complexity and Complex Networks, City University of Hong
Kong, Hong Kong, China.*

eegchen@cityu.edu.hk

Cellular automata are arrays of finite state machines that can exist in a finite number of states. These machines update their states simultaneously based on specific local rules that govern their interactions. This framework provides a simple yet powerful model for studying complex systems and emergent behaviors. We revisit and reconsider the traditional notion of an algorithm, proposing a novel perspective in which algorithms are represented through the dynamic state-space configurations of cellular automata. By doing so, we establish a conceptual framework that connects computation to physical processes in a unique and innovative way. This approach not only enhances our understanding of computation but also paves the way for the future development of unconventional computing devices. Such devices could be engineered to leverage the inherent computational capabilities of physical, chemical, and biological substrates. This opens up new possibilities for designing systems that are more efficient, adaptive, and capable of solving problems in ways that traditional silicon-based computers cannot. The integration of cellular automata into these domains highlights their potential as a transformative tool in the ongoing evolution of computational theory and practice.

1. Introduction

How do you decide to choose an algorithm or not? (Fig. 1) This selection could be random, influenced, or inadvertent. When buy a mobile, a number of algorithms pre-installed are running. Consequently, you do not know what is happening with your personal information if you have no experience in computer science. Surely, several of these pre-installed applications upload information to the internet.

Every day, we encounter or discover problems, and we promptly find solutions for some of them. In the realm of computer science, this process involves the development of algorithms. Consequently, the community has generated a number of algorithmic solutions that can be implemented across various devices. Many of these algorithms can be stored in databases such as GitHub, The Algorithms, Reddit, OpenML, and others.

Indeed, some artificial intelligence-based applications scour the internet, extracting code snippets from these repositories to answer queries for users. Examples of such services include ChatGPT, BING chat, Copilot, Bard, or similar platforms. A collapse in our information technology landscape would not signify the commencement of a new era but rather, a challenge to enhance the work of our previous developers.

Several years ago, one of the authors experienced an anecdote while collaborating with Prof. Harold McIntosh in Puebla. In the midst of a routine maintenance session for his NextStep computer, which involved creating a backup of the hard disk, an unfortunate accident transpired, resulting in the burning of the hard disk. Over ten years of work were stored on that hard disk, and Harold's response was remarkably. Subsequently, he acknowledged the loss of the information and began contemplating how to initiate fresh work—recreating programs, rewriting reports, and so on. This incident presented a potential scenario for starting anew, particularly in a hypothetical post-apocalyptic computing stage.

How can one systematically generate algorithms, especially when considering the essence of computation—the development of programs and algorithms? The systematic and automatic definition of these processes can be both costly and intricate (Fig. 1). This paper explores examples illustrating the intricacy associated with certain codes. Within this framework, we employ cellular automata theory to generate programs based on personal decisions, examining their complexity in a randomised fashion.

Cellular automata field was initiated by John von Neumann in the 1950s. It was inspired by the exploration and resolution of problems such as inher-

ent parallel computing, nervous systems, self-reproduction of machines, and biological cellular processes.¹ Over more than 60 years of research, this theory has found numerous applications in various academic fields, including biology, physics, economics, mathematics, chemistry, sociology, computer science, patterns, tilings, astronomy, nature, artificial life, complex systems, and so on.^{2–21}



Fig. 1. Are you free to make a decision regarding the algorithm to be employed?

The logic to implement a cellular automaton is simple; however, certain rules have the capability to produce highly complex behaviour, giving rise to intricate patterns during the system's evolution. The most renowned instance is Conway's Game of Life, a binary two-dimensional cellular automaton that evolves with orthogonal and diagonal 1-range neighbours (Moore neighbourhood). Despite the simplicity of the conditions, the resulting behaviour is complex. Over 50 years later, researchers of Life continue to

report, construct, and discover new patterns and configurations.^{22–26a}

Cellular automata typically are studied in one, two, three dimensions. However some studies were done in six, seven, eight dimensions discovering very interesting non-trivial collective behaviour.²⁷ At the same time most complex rules can be designed increasing the number of symbols in its alphabet, they can be explored with two powerful and free versions of software: *Golly*^b and *DDLab*.^{28c}

Figure 2 displays some kinds of cellular automata. Figure 2a shows a totalistic two-dimensional hexagonal cellular automaton evolving with six states is calculated the free software DDLab. Figure 2b shows a semi-totalistic three-dimensional cellular automaton evolving with two states calculated the free software *Ready*.^d Figure 2c shows a two-dimensional Penrose cellular automaton evolving with four states calculated the free software *Ready*. Figure 2d shows a binary one-dimensional cellular automaton the elementary evolution rule 22 is calculated the free software *CAViewer*.^e

An algorithm is a string. In the 1960s, Marvin Minsky discussed which algorithms have histories of evolution and histories can be drawn in a state machine diagram, as an attractor.²⁹

In this manner, we can represent the famous *C* program,³⁰ commonly known as “the hello world,” as a binary string by translating each ASCII symbol of the program into strings of eight symbols. If we were without computers in a post-apocalyptic scenario, the probability of retrieving this program from a binary language would be the order of 2.19×10^{-191}

```
#include <stdio.h>
int main(void)
{
    printf( 'Hello world\n' );
    return 0;
}
```

This way, the program *hello world* written in *C* can be expressed in 640 binary symbols given for the next word:

^aLifeWiki. <https://conwaylife.com/wiki>

^bGolly. <https://golly.sourceforge.io/>

^cDiscrete Dynamics Lab. <http://www.ddlab.com/>

^dReady. <https://github.com/GollyGang/ready>

^eCAViewer. https://www.comunidad.escom.ipn.mx/genaro/Cellular_Automata_Repository/Software.html

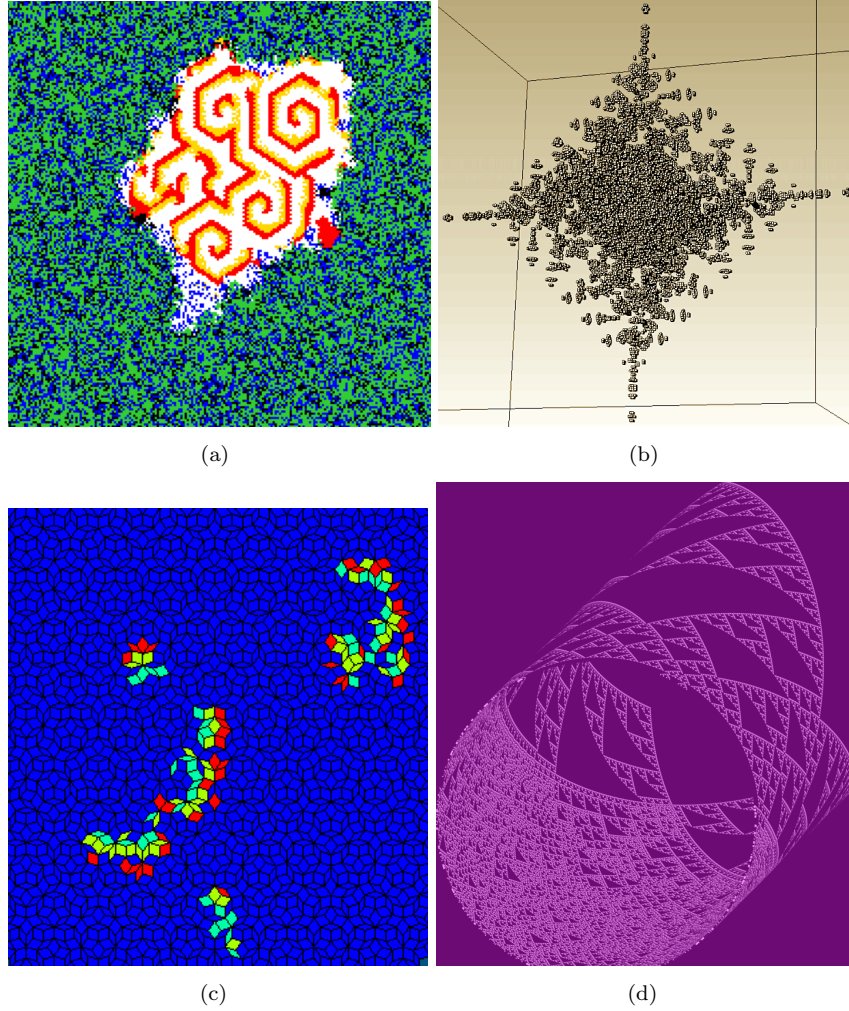


Fig. 2. Examples of cellular automata configurations. (a) Hexagonal lattice with multiple states (DDLab), (b) Three-dimensional space with two states (Ready), (c) Penrose lattice with multiple states (Ready), (d) One-dimensional space with two states (CAViewer).

```

00100011011010010110111001100011011011000111010101100100011001010010000000111100011100110111010
00110010001101001011011110010111001101000001111100000101001101001011011100111010000100000011011
01011000010110100101101110001010000111011001101111011010010110010000101001000010100111101100001
0100010000001000000010000001110000011100100110100101101110011101000110011000101000001001110010

```

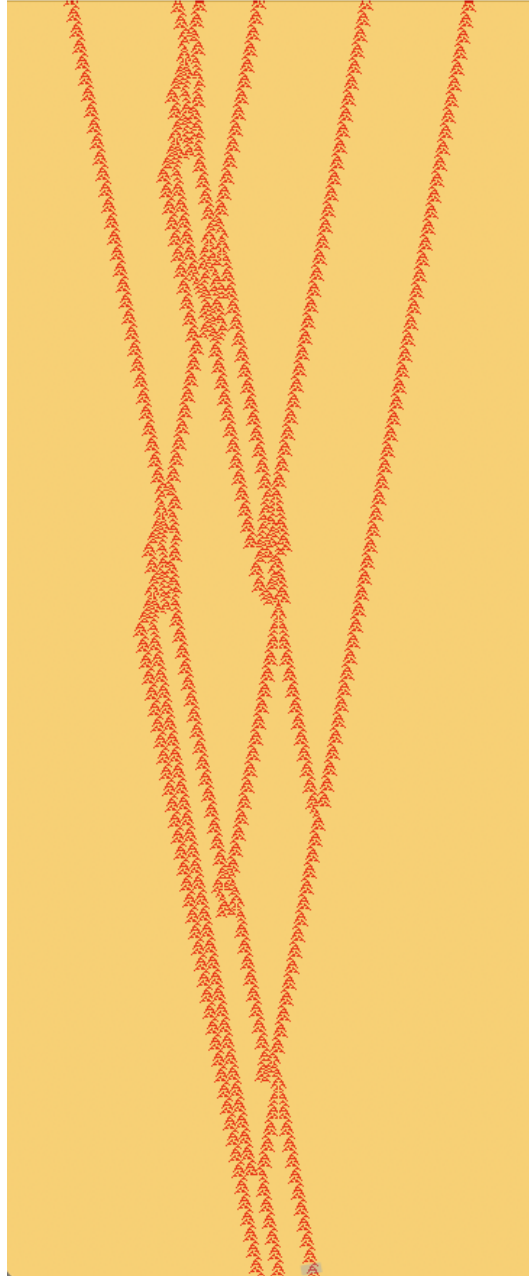



Fig. 4. Fredkin gate in elementary cellular automaton with memory rule 22. It is the relation $(c = 1, p = 1, q = 1) \rightarrow (x = 1, y = 1, z = 1)$ running on an initial condition of 413 cells for 882 generations.

$$\begin{array}{ll}
\varphi(1, 1, 1) \rightarrow 0 & \varphi(0, 1, 1) \rightarrow 1 \\
\varphi(1, 1, 0) \rightarrow 1 & \varphi(0, 1, 0) \rightarrow 1 \\
\varphi(1, 0, 1) \rightarrow 1 & \varphi(0, 0, 1) \rightarrow 1 \\
\varphi(1, 0, 0) \rightarrow 0 & \varphi(0, 0, 0) \rightarrow 0
\end{array}$$

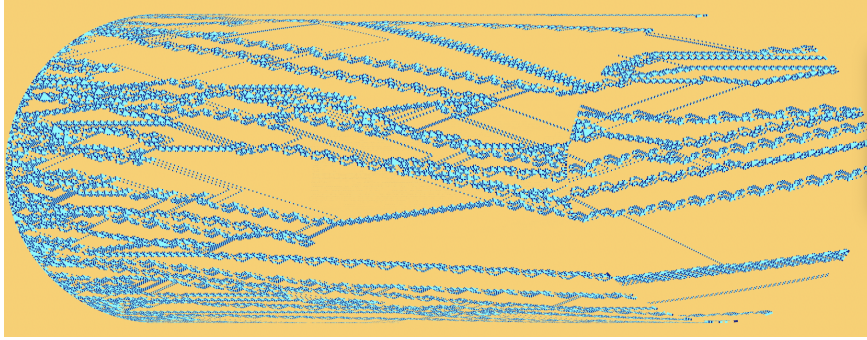


Fig. 5. Typical evolution from a random initial condition elementary cellular automaton rule 110 (filtered). It shows the emergence of non-trivial patterns named gliders, travelling and colliding in different scenarios. This snapshot is calculated on a tube of 1,500 cells for 584 generations.

dimensional binary cellular automaton in which the local function assesses three variables, defined by equations in Tab. 1.

This function has the ability to produce complex patterns: gliders, particles, waves, mobile self-localisations (Fig. 5). These complex patterns emerge during the evolution and the interactions (collisions), determine this artificial universe. A particular interest to study such systems is the capacity to simulate computers in their own dynamics.³⁶ Gliders in rule 110 have a wide variety of types, extensions and combinations. We can handle packages of them in one or several phases. These gliders have important characteristics useful to define distances, slopes, speeds, periods, collisions, and phases.³⁷

Several years ago, a computation was developed in rule 110, implementing a cyclic tag system written by Cook.³⁸ Cyclic tag system is a variant of the classic tag system developed by Emil Post. It is a substitution system that reads the first value of a string, deletes a constant number of them, and inserts new symbols at the end of the string. This way, a cyclic tag system is able to handle more substitutions for every symbol cyclically.^{9,34,35,38}

The next string represents the initial configuration to simulate the

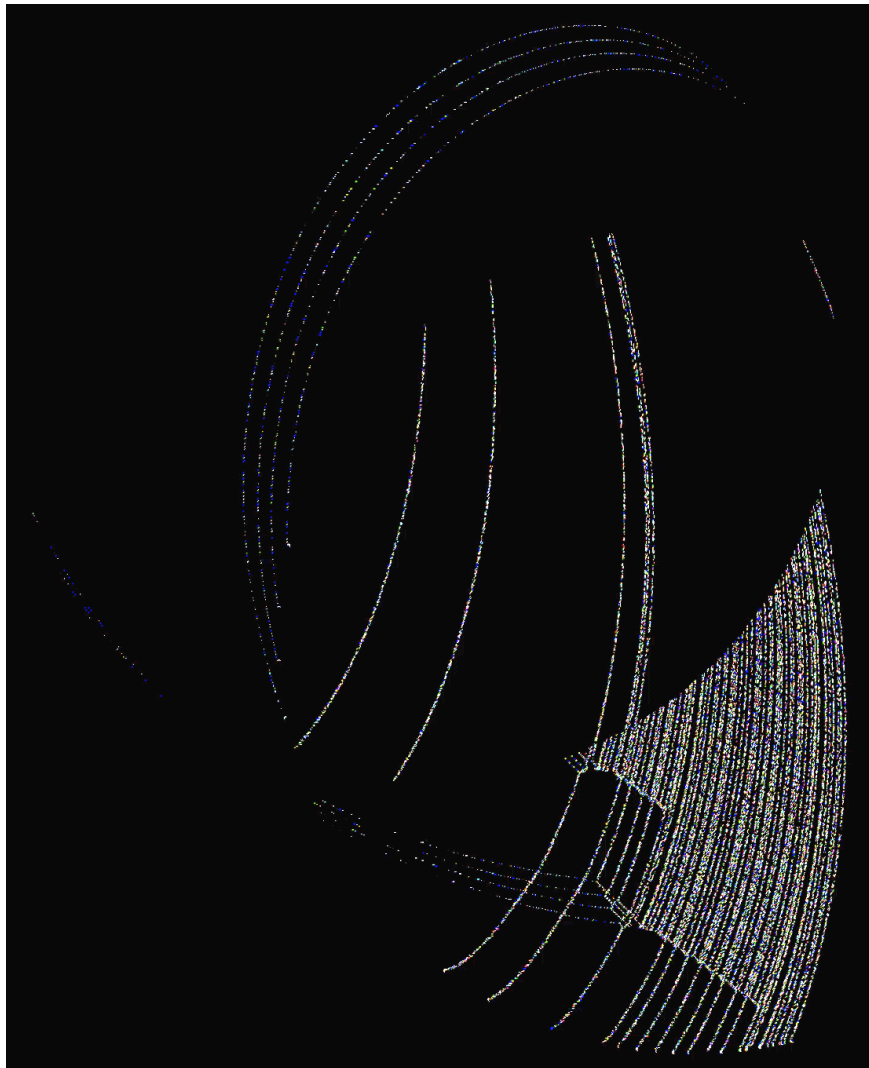


Fig. 6. A cyclic tag system evolving in cellular automaton collider elementary rule 110. This snapshot shows the main operations as: read, insert and delete data in the vertical type. To insert new values on the type a lot of particles travel from the left side and they need to cover long distances. This simulation was calculated with DDLab. This simulation implies a space more than 3×10^6 cells.

function of a cyclic tag system in the elementary cellular automaton rule

13

[illegible]

[illegible]

Short title

23

[illegible]

Short title

25

100010011011110001001101111000100110111100010011011110001001101111000100110111100010011011
111000100110111100010011011110001001101111000100110111100010011011110001001101111000100110
11110001001101111000100110111100010011011110001001101111000100110111100010011011110001001
10111100010011011110001001101111000100110111100010011011110001001101111000100110111100010
01101111000100110111100010011011110001001101111000100110111100010011011110001001101111000
10011011110001001101111000100110111100010011011110001001101111000100110111100010011011110
00100110111100010011011110001001101111000100110111100010011011110001001101111000100110111
100010011011110001001101111000100110111100010011011110001001101111000100110111100010011011
111000100110111100010011011110001001101111000100110111100010011011110001001101111000100110
111110001001101111000100110111100010011011110001001101111000100110111100010011011110001001
101111000000100110111100010011011110001001101111000000100110111100010011011110001001101111
100000010011011110001001101111000110110101111000100110111101110001001101111000100110010001
11001100011001101111000100110100111111010111100010011011110001001101111000100110111100010
01100100011001100011110001001101111000100110011101110011000100110111100010011010001101001100
01001101111000100110111100010011010001011110001001101111001111011100110111100010011011111
00010011011101011101011100110111110011100010011011110001110111101001111000100110111100010
01101110001110110001001101111000101101000011011110001001101111000100110111110000110111001101
11110001001101111000100110111100111000001001101110110111001101111000100110111100010011
0111011100110000100110111100010110100001101111101111101111101011110001001101111000100110111
1001111101110011011111000100110111100010011011101101110011011110001001100100011100110001
1100110111100010011010011111110101111000100110111100010011011110001001101111000100110010001
110011000111100010011011110001001100111011100110001001101111000100110100011010011000100110111
11000100110111100010011010001011110001001101111001111011100110111100010011011110001001101
1101101110101110011011111001111000100110111100011101111010011110001001101111000100110111000
1110110001001101111000101101000011011110001001101111000100110111110000110111001101111000100
11011110001001101111001110000010011011101101110101110011011110001001101111000100110111011100
11000010011011110001011010000110111110111110111110101111000100110111100010011011111001111101
110011011110001001101111000100110111011011101011100110111110011100010011011110001001101111
0001001101111000111011001011011111011111010111100001011001001101111000100110111100010
011011101110011000010011011110001001101111000100110111110011100010011011110001001101111000
100110111110011100000100110111100010011011110001001101111100111101110011011100011101100010011
011110001011010000110111100010011011110001001101111100001101110011011110001001101111000100
11011111001110000010011011101101110011011110001001101111000100110111011100110000010011011
111000101101000011011111011111011111010111100010011011110001001101111001111011100110111100
010011011110001001101110110111010111001101111100111000100110111100010011011110001001101111
0001110110010110111110111110111110101111000010110010011011110001001101111000100110111001
100001001101111000100110111100010011011111001110001001101111000100110000001111000001111101
01111000100110111111110001111000100110111100010011011110001001101111000100110000001111100
000111100010011011110001001101100010111100010011011110001001100110001111100010011011110001

00110111110001001101110001110110001001101110111001100001001101111100010011011111000100110111110
0111100010011011110100110111100110111100010011011110011100000100110111100010011011110001011
01000011011110111110111110101111000100110111100010011011110011110111001101111000100110111
110001001101110101110101110011011111001110001001101111000100110111100010011011110001110110
01011011111011111011111010111100001011001001101111000100110111100010011011101110011000010011
01111100010011011110001001101111100111100010011011110001001100000011111000001111101011110001
001101111111100011110001001101111000100110111100010011011110001001100000011111000001111100
01001101111100010011011000101111100010011011110001001100110001111100010011011110001001101111
0001001101110001110110001001101110110011000010011011110001001101111000100110111110011100010
0110111110100110111100110111100010011011110011000001001101111000100110111100010110100001101
11110111110111110101111000100110111100010011011110011110111001101111000100110111100010011
0111011011101011100110111110011110001001101111000100110111100010011011110001110110010110111
1011111011111010111100001011001001101111000100110111100010011011101110011000010011011110001
00110111110001001101111100111100010011011111010011011110011011110001001101111000100110111100
0100011001100011111000010110010011011110001110111100101111000100110111100010011011110001110
1100101101111100010011011110001001101111010011011110011011110001001101111000100110111011011
010111001101111000100110111100010011011101100110000100110111100010110100001101111011111011
1110101111100010011011110001001101111001111011100110111100010011011110001001101110110111010
111001101111100111100010011011110001001101111000100110111100011101100101101111101111101111
01011110000101100100110111110001001101111000100110111011100110000100110111100010011011110001
001101111110011110001001101111101001101111001101111000100110111100010011011110001000110011000
111110000101100100110111100011101111001011110001001101111000100110111100011101100101101111
00010011011110001001101111010011011110011011110001001101111010000111001100011100110001001101
11110001000000011011100110111100010011011110001001101111010000111001100011110011
011110001001101111000010110010011011110001001101111011101011100011011110001001101111000100
1101111000111011110010111100011101111010011110001001101111000100110111100010001111011011
1110001001101110001011100010011011110001001111110001111000100110111100010011011000001001100
010011010001101001100010011011110001001101111000100111100110111100010011011110001001101111
00011100001110001111100010001111101101111000100110111100010011011110001001101111001110000010
0110100011010011000100110011000111110001001101111000100110111100011101111010011110001001101
11110001001101111000100011111011011110001001101111010000111001100011100110001001101111000100
0000011011100110111110001001101111000100110111101000011100110001110011011110001
001101111100001011001001101111000100110111101101011100011011110001001101111000100110111100
011101111100101111000111011111010011110001001101111000100110111100010001111011011110001001
101111100010011011110001001101111000100110111100010011011110001001101111000100110111100010
011011111000100110111100010011011110001001101111000100110111100010011011110001001101111000
100110111110001001101111000100110111100010011011110001001101111000100110111100010011011110
001001101111000100110111100010011011110001001101111000100110111100010011011110001001101111
1000100110111100010011011110001001101111000100110111100010011011110001001101111000100110111

Table 1. Probability to get an algorithm systematically or randomly.

machines	
C hello world	$2.19 \times 10^{-191}\%$
1D Fredkin gate ECAM rule 22	$2.36 \times 10^{-620}\%$
Cyclic tag system ECA rule 110	$1.18 \times 10^{-16928}\%$

to an apocalypse are necessary for significant advancements to take place.

References

1. J. von Neumann, *Theory of Self-Reproducing Automata (edited and completed by A.W. Burks)*. University of Illinois Press (1966).
2. T. Toffoli and N. Margolus, *Cellular Automata Machines: A New Environment for Modeling*. The MIT Press (1987).
3. H. Gutowitz (Ed.), *Cellular Automata: Theory and Experiment*. The MIT Press (1991).
4. S. Wolfram, *Cellular Automata and Complexity: Collected Papers*. CRC Press (1994).
5. C. G. Langton (Ed.), *Artificial Life: An Overview*. The MIT Press (1995).
6. M. Sipper, *Evolution of Parallel Cellular Machines: The Cellular Programming Approach*. Springer (1997).
7. A. Adamatzky, *Computing in Nonlinear Media and Automata Collectives*. CRC Press (2001).
8. A. Ilachinski, *Cellular Automata: a Discrete Universe*. World Scientific Publishing Company (2001).
9. S. Wolfram, *A New Kind of Science*. Wolfram Media Champaign, IL (2002).
10. B. Chopard and M. Droz, *Cellular Automata Modeling of Physical Systems*. Cambridge University Press (1998).
11. A. Deutsch and S. Dormann, *Cellular Automata Modeling of Biological Pattern Formation*. Birkhäuser Boston (2005).
12. L. B. Kier, P. G. Seybold, and C.-K. Cheng, *Modeling chemical systems using cellular automata*. Springer Science & Business Media (2005).
13. M. Margenstern, *Cellular Automata in Hyperbolic Spaces: Theory. Volume 1*. Old City Publishing (2007).
14. H. V. McIntosh, *One Dimensional Cellular Automata*. Luniver Press (2009).
15. N. Boccara, *Modeling Complex Systems*. Springer (2010).
16. K. Mainzer and L. Chua, *The Universe as Automaton: From Simplicity and Symmetry to Complexity*. Springer Briefs in Complexity (2011).
17. R. Alonso-Sanz, *Discrete Systems with Memory*. World Scientific (2011).
18. E. Goles and S. Martínez (Eds.), *Cellular Automata and Complex Systems*. Springer Science & Business Media (2013).
19. K. Morita, *Theory of Reversible Computing*. Springer (2017).

20. S. Das and G. J. Martínez (Eds.), *Proceedings of Second Asian Symposium on Cellular Automata Technology: ASCAT 2023*. Springer Nature (2023).
21. H. Zenil (Ed.), *A Computable Universe: Understanding and Exploring Nature as Computation*. World Scientific (2013).
22. M. Garden, The fantastic combinations of john conway's new solitaire game "life", *Scientific American*. **223**, 120–123 (1970).
23. W. Poundstone, *The Recursive Universe: Cosmic Complexity and the Limits of Scientific Knowledge*. Contemporary Books (1985).
24. A. Adamatzky (Ed.), *Game of Life Cellular Automata*. Springer (2010).
25. N. Johnston and D. Greene, *Conway's Game of Life: Mathematics and Construction*. Lulu.com Press (2022).
26. S. Das, S. Roy, and K. Bhattacharjee (Eds.), *The Mathematical Artist: A Tribute to John Horton Conway*. Springer Nature (2022).
27. H. Chaté and P. Manneville, Collective behaviors in spatially extended systems with local interactions and synchronous updating, *Progress of Theoretical Physics*. **87**(1), 1–60 (1992).
28. A. Wuensche, *Exploring Discrete Dynamics*. Luniver Press (2011).
29. M. L. Minsky, *Computation: Finite and Infinite Machines*. Prentice-Hall Englewood Cliffs (1967).
30. B. W. Kernighan and D. M. Ritchie, *The C Programming Language*. Prentice-Hall, Inc. (1988).
31. G. J. Martínez and K. Morita, Conservative computing in a one-dimensional cellular automaton with memory, *Journal of Cellular Automata*. **13**(4), 325–346 (2018).
32. G. Chen, M.-F. Danca, X. Yang, G. J. Martínez, and H. Yu, Research frontier in chaos theory and complex networks, *Entropy*. **20**(10), 734 (2018).
33. G. J. Martínez, A. Adamatzky, and R. Alonso-Sanz, Designing complex dynamics in cellular automata with memory, *International Journal of Bifurcation and Chaos*. **23**(10), 1330035 (2013).
34. G. J. Martínez, H. V. McIntosh, J. C. Seck-Tuoh-Mora, and S. V. Chapa-Vergara, Reproducing the cyclic tag system developed by matthew cook with rule 110 using the phases f_i-1 , *Journal of Cellular Automata*. **6**(2-3), 121–161 (2011).
35. O. L. Barrientos, G. J. Martínez, A. Adamatzky, and S. Ninagawa. Synchronizing a virtual cellular automaton collider. In *2022 Tenth International Symposium on Computing and Networking (CANDAR)*, pp. 1–10 (2022).
36. C. Moore and S. Mertens, *The Nature of Computation*. OUP Oxford (2011).
37. G. J. Martínez, H. V. McIntosh, and J. C. Seck-Tuoh-Mora, Gliders in rule 110, *International Journal of Unconventional Computing*. **2**(1), 1–49 (2006).
38. M. Cook, Universality in elementary cellular automata, *Complex Systems*. **15**(1), 1–40 (2004).