Evolutionary Search for Cellular Automata Logic Gates with Collision-Based Computing

Emmanuel Sapin
Larry Bull
Faculty of Computing, Engineering, and Mathematical Sciences,
University of the West of England,
Bristol, BS16 1QY, UK

We aim to search for cellular automata candidates using an automatic system for the demonstration of collision-based universality. Some of these candidates are able to simulate Turing machines in their space-time dynamics using gliders and glider guns.

In this paper, we present an evolutionary algorithm that searches for the simulation of a logic gate by cellular automata. Different parameters are tried and other search techniques are also presented to provide a benchmark.

Results show how a large number of simulations of logic gates were found.

1. Introduction

The emergence of computation in complex systems with simple components is a hot topic in the science of complexity [1]. A uniform framework to study emergent computation in complex systems are cellular automata [2]. They are discrete systems in which an array of cells evolves from generation to generation on the basis of local transition rules [3].

The well-established problems of emergent computation and universality in cellular automata have been tackled by a number of people in the past 30 years [4–8] and remains an area where amazing phenomena at the edge of theoretical computer science and nonlinear science can be discovered.

The best-known universal automaton is the Game of Life [9]. It was shown to be universal by Conway in [10] who simulated logic gates by the Game of Life. He employed *gliders* and *glider guns* for this simulation. Gliders are mobile self-localized patterns of nonresting states, and glider guns are patterns which, when evolving alone, periodically recover their original shape after emitting some gliders.

We aim to construct an automatic system for the discovery of computationally universal cellular automata, spatially. We are interested in the automatic construction of AND gates simulated by cellular automata.

https://doi.org/10.25088/ComplexSystems.17.4.321

An evolutionary algorithm is used to modify the cellular automata transition rules in order to simulate AND gates. Tabu search and Monte Carlo methods are used as benchmarks.

The paper is arranged as follows: section 2 describes previous related work, section 3 sets out the characteristics of the search methods, and then the simulations of AND gates are described in section 4. The last section summarizes the presented results and discusses directions for future research.

2. Previous work

2.1 Game of Life

In 1970, Conway discovered a special cellular automaton (that he called the Game of Life) that was later popularized by Gardner [9]. Conway, Berlekamp, and Guy showed that the Game of Life can implement any function calculable by a Turing machine. Their proof of the universality of the Game of Life uses gliders and glider guns [10]. A glider gun is a pattern that emits a stream of gliders (used to carry information). It is possible to combine glider guns in order to simulate logic gates and circuits.

An AND gate is simulated in Figure 1 with two inputs streams *A* and *B*.

Data streams are encoded by glider streams (i.e., the absence of gliders represents the value 0 and the presence of gliders represents the value 1). A glider gun that emits a new glider every 30 generations is used by

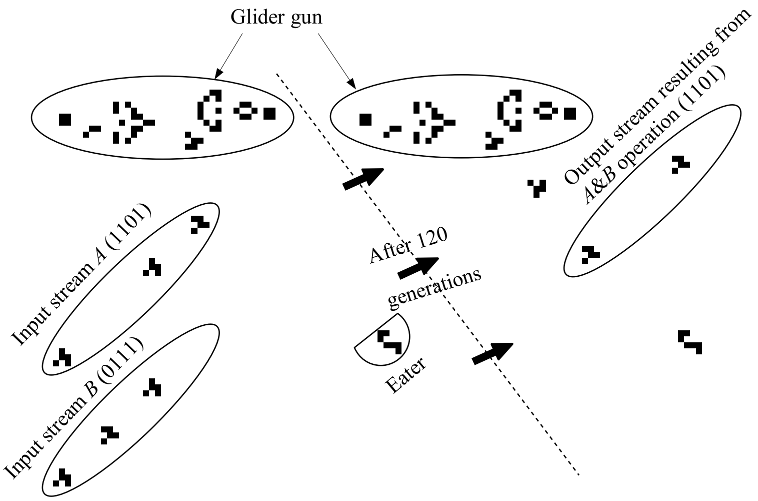


Figure 1. AND gate simulated by the Game of Life [10].

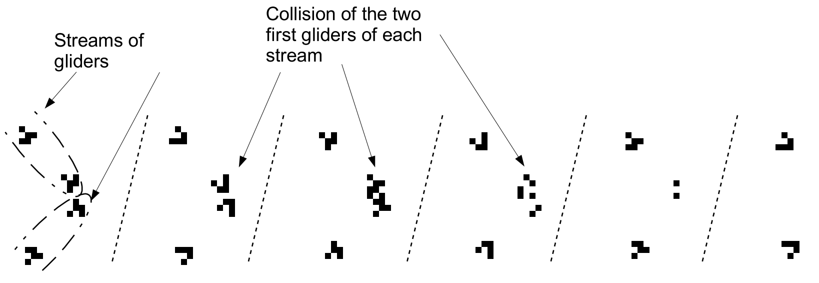


Figure 2. Collision between two streams of gliders.

this simulation. The glider gun creates a glider stream that "crashes" stream A. If a glider is present in stream A the two gliders are destroyed, otherwise the glider emitted by the gun continues its run. The stream resulting from this first collision, at a right angle to stream A, is not(A). This stream crashes stream B producing a stream aligned with the stream B, which is the result of the operation A *and* B. The synchronization and the position of the different components are critical to the proper function of the simulation. This simulation is possible because the collision of two gliders destroys them both, letting the other gliders continue their runs as shown in Figure 2.

2.2 Cellular automata

In [11], Wolfram studies the space $\mathcal{I}$ of two-dimensional isotropic cellular automata, with rectangular eight-cell neighborhoods: if two cells have the same neighborhood states by rotations and symmetries, then these two cells take the same state at the next generation. There are 512 different rectangular eight-cell neighborhood states as shown in Figure 3.

The neighborhood states can be divided into two subsets of 256 neighborhood states. The neighborhood states with the central cell in state zero are in the first subset and the others are in the second subset. All these neighborhood states can be put into 102 subsets of isotropic neighborhood states as shown in Figure 4 for the first subset of 256 neighborhood states.

The first element of each subset of isotropic neighborhood states is used to represent a rule of I as shown in Figure 5 to determine what will become of a cell in the next generation, depending on its subset of isotropic neighborhood states.

There are 102 subsets of isotropic neighborhood states, meaning that there are 2^{102} different cellular automata in $\mathcal{I}$.

For conciseness and readability, the following convention has been adopted for the cellular automaton presentation. Each neighborhood state is referred to by the number of cells from 0 to 8 among it and a

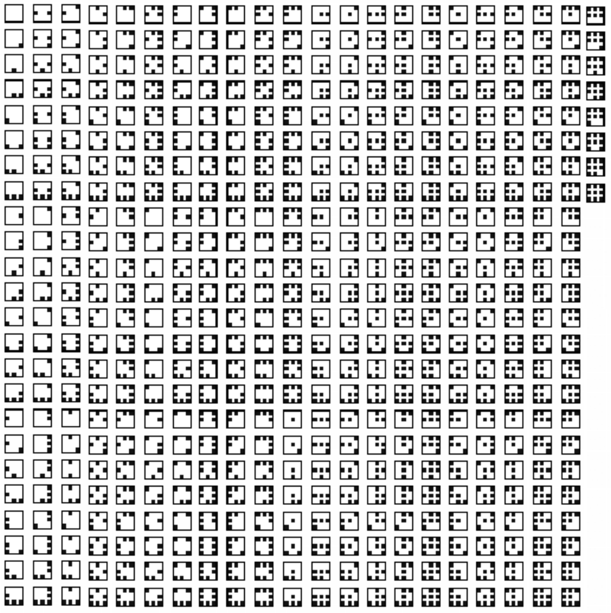


Figure 3. 512 Moore neighborhood states.

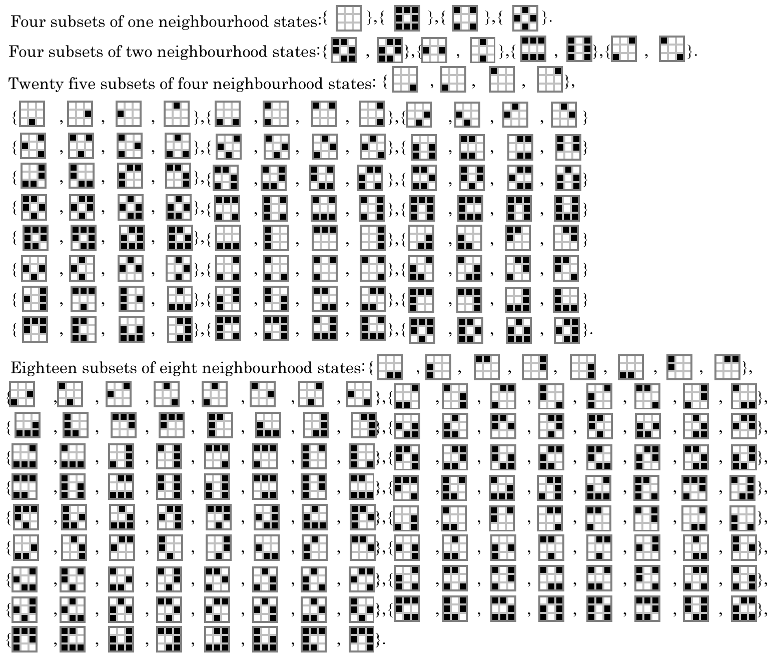

Four subsets of one neighbourhood states:
Four subsets of two neighbourhood states:
Twenty five subsets of four neighbourhood states:
Eighteen subsets of eight neighbourhood states:

Figure 4. Subset of isotropic neighborhood states.

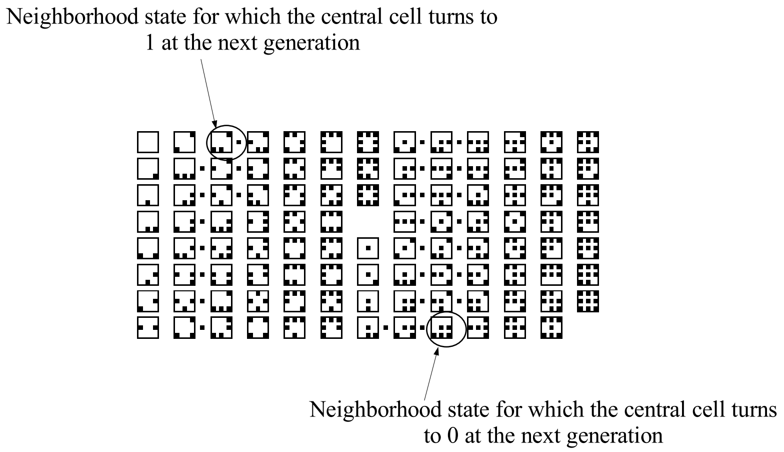

Neighborhood state for which the central cell turns to 1 at the next generation
Neighborhood state for which the central cell turns to 0 at the next generation

Figure 5. The squares are the 102 neighborhood states describing a cellular automaton of $\mathcal{I}$. A black cell on the right of the neighborhood state indicates a future central cell.

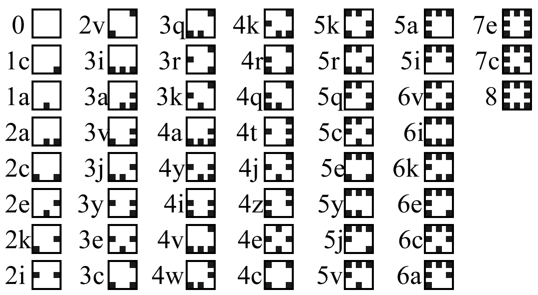


Figure 6. Neighborhood states.

letter, as shown in Figure 6. In the convention, the neighborhood states that allow the central cell to survive is noted in the first part after the letter *s* of the cellular automaton notation and the neighborhood states that allow the central cell to be born is noted in the second part after the letter *b*. For example, the notation *s*2*iv*3*a*8 / *b*4*i*8 means a cell survives for the neighborhood states 2*i*, 2*v*, 3*a*, and 8.

2.3 Search methods

In order to find simulations of AND gates, we examined search methods such as Monte Carlo, Tabu [12], and an evolutionary algorithm [13], as briefly described next.

The Monte Carlo method consists solely of generating random solutions and testing them.

https://doi.org/10.25088/ComplexSystems.17.4.321

Tabu search traverses the solution space by testing mutations of an individual solution. Tabu search generates many mutated solutions and moves to the best solution of those generated. In order to prevent cycling and encourage greater movement through the solution space, a tabu list is maintained of partial or complete solutions. It is forbidden to move to a solution that contains elements of the tabu list, which is updated as the solution traverses the solution space.

Evolutionary algorithms incorporate aspects of natural selection or survival of the fittest. A population of structures is maintained (usually initially generated at random) that evolves according to rules of selection, recombination, mutation, and survival referred to as genetic operators. A shared environment is used to determine the fitness or performance of each individual in the population. The fittest individuals are more likely to be selected for reproduction through recombination and mutation, in order to obtain potentially superior ones.

2.4 Evolving cellular automata

Several good results using the evolution of cellular automaton rules to perform useful tasks have been published. After Packard [14], Mitchell *et al.* [15–18] investigated the use of evolutionary computing to learn the rules of uniform one-dimensional, binary cellular automata. A genetic algorithm [13] produces the entries in the update table used by each cell and candidate solutions are evaluated with regard to their degree of success for the given tasks of density and synchronization.

Sipper [19] presented a related approach, which produces nonuniform solutions. Each cell of a one- or two-dimensional cellular automaton is viewed as a genetic algorithm population member, mating only with its geographical neighbors and receiving an individual fitness. Sipper shows an increase in performance over the work of Mitchell *et al.*, exploiting the potential for spatial heterogeneity in the tasks. Koza *et al.* [20] have also repeated the work of Mitchell *et al.*, using genetic programming [21] to evolve update rules. They report similar results.

The manual search for gliders in cellular automata has been described by Wuensche who used his Z-parameter and entropy [22], Adamatzky *et al.* with a phenomenological search [23], and Eppstein [24]. Sapin *et al.* [25], Ventrella [26], and Lohn *et al.* [27] have considered the emergence of glider-based universality in cellular automata *via* machine learning.

Sapin *et al.* proposed an evolutionary approach in the search for complex glider guns [25, 28–31]. In [32], a glider gun is presented and used to build an AND gate. In this paper, the simulation of the AND gate of Conway was manually reproduced with this glider gun as shown in Figure 7.

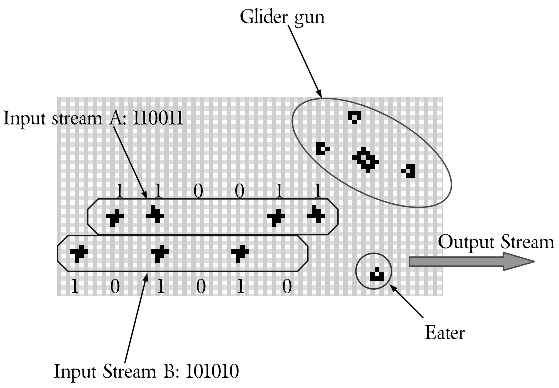


Figure 7. An AND gate manually built from the simulation scheme of an AND gate in the Game of Life.

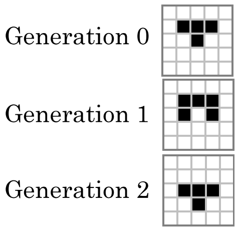


Figure 8. Glider.

In [33], 751 glider guns were found emitting the glider shown in Figure 8. These guns were classified depending on the number of emitted gliders. The period of the gun was automatically determined.

3. Characteristics of the search methods

The next subsection describes the main idea of the search methods used. The search space is determined in the second subsection, and the remaining subsections describe the search methods used to modify the cellular automata transition rules.

3.1 Idea of the search methods

The idea of the search methods is to focus on an existing gun, named G of period p, and try to find a simulation of an AND gate using this gun. Each search method was tried for all of the 751 glider guns in [33].

The synchronization and position of the different components, critical in simulating an AND gate, depend on the period of G. For conciseness and readability, the search for simulations of AND is presented

https://doi.org/10.25088/ComplexSystems.17.4.321

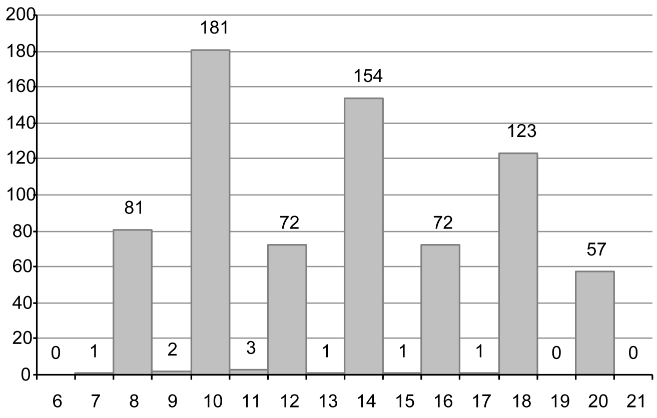


Figure 9. Distribution of the period of the discovered glider gun in [33].

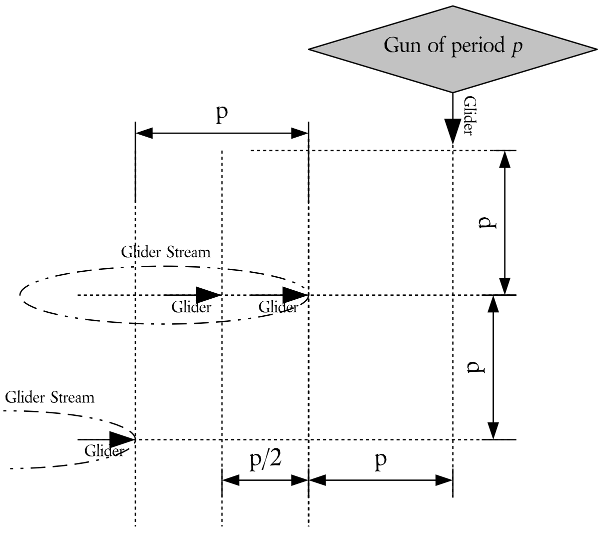


Figure 10. Detail of constructing an AND gate.

only for even period guns as less than three percent of the guns found in [33] have an odd period, as shown in Figure 9.

The schema of an AND gate as shown in Figure 10 is built for all guns and the search method modifies the transition rules of the cellular automata that exhibit the gun in order to find the transition rule that makes simulating the AND gate possible.

After $2p$ generations, the first glider emitted by the gun, named $G1$, and the first glider of the first stream, named $A1$, collide. This collision must result in the destruction of these two gliders before the second glider $G2$ emitted by the gun and the second glider $A2$ of the first stream arrives. So, after $3p - 6$ generations, the gliders $G1$ and $A1$ must have disappeared and the schema of Figure 10 has to evolve to the schema of Figure 11. In this case the AND gate can be simulated.

3.2 Search space

The search space depends on the choice of the glider gun G. The set of cellular automata that exhibit glider gun G is the search space.

This set is determined by the neighborhood states used by G. We use the same process to determine the neighborhood states used by a gun that was developed in [33] to determine the neighborhood states used by a glider. The method is shown in Figure 12 for the glider of Figure 8. This process was applied to the 751 glider guns in [33]. The distribution of the numbers of used neighborhood states of these guns is shown in Figure 13. The minimum and maximum numbers of used neighborhood states are 29 and 83. The average number of used neighborhood states is 44.

Let N be the number of used neighborhood states of the gun G. The existence of the glider gun G does not depend on the value associated with the neighborhood that is not used by G. The total number of neighborhood states is 102, so the number of cellular automata that exhibit G is 2^{102-N}.

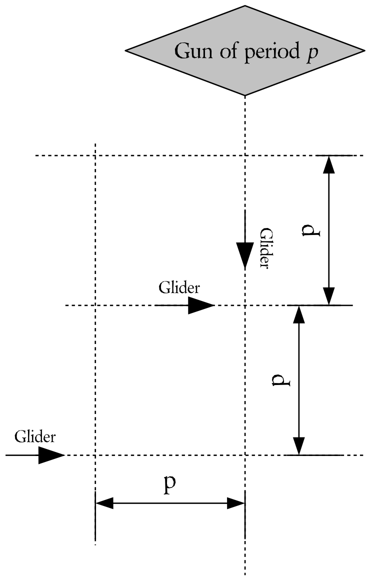
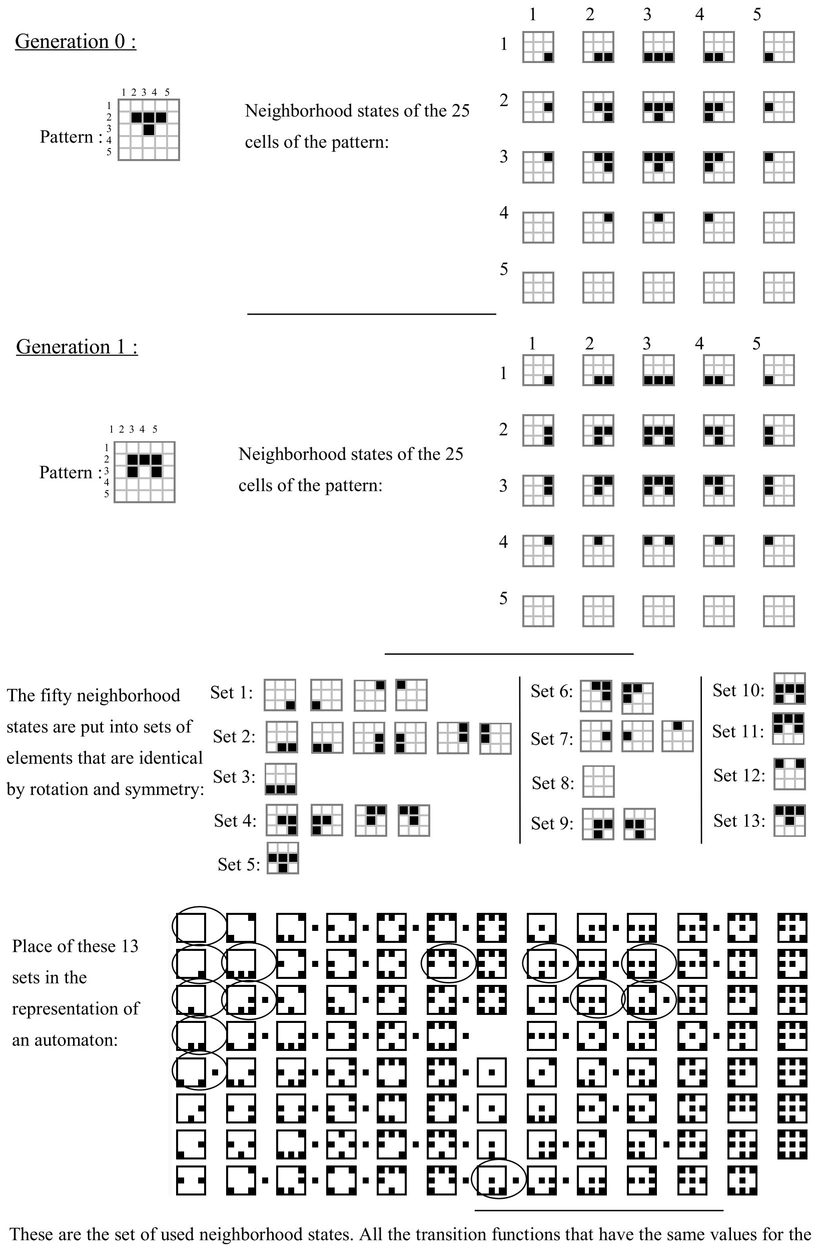


Figure 11. Detail of constructing an AND gate.



Neighborhood states of the 25 cells of the pattern:

Neighborhood states of the 25 cells of the pattern:

The fifty neighborhood states are put into sets of elements that are identical by rotation and symmetry:

Place of these 13 sets in the representation of an automaton:

These are the set of used neighborhood states. All the transition functions that have the same values for the selected neighborhood states exhibit the given glider.

Figure 12. Detail of constructing the set of neighborhood states that are used by a glider.

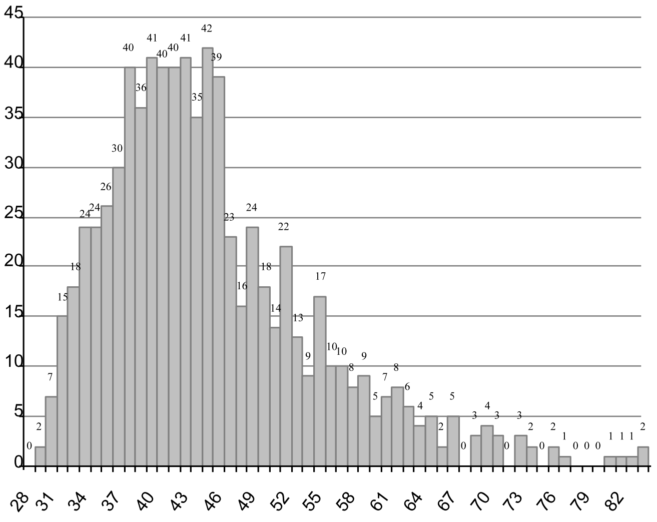


Figure 13. Distribution of the number of neighborhoods the discovered glider guns in [33] used.

The smaller search space has $5.24*10^5$ elements and the bigger search space has $9.45*10^{21}$. The average size of the search space is $1.94*10^{20}$.

3.3 Evolutionary algorithm

- *Initialization.* The search space is the set, described in the previous section, of cellular automata exhibiting the glider gun G. A cellular automaton of this space can be described by a cell-state transition table. An individual is a cellular automaton coded as a bit string of 102 Booleans representing the values of a cell at the next generation for each neighborhood state. The bits of the neighborhood state that are used by the gun G are determined by the evolution of G. The bits of the neighborhood state that are not used by the gun G are initialized at random. The population size is 100 individuals.

- *Fitness function.* The idea of the fitness function is to build the schema of an AND gate as shown in Figure 10 and to check how close the simulation is to the behavior of an AND gate. The schema of Figure 10 evolves during $3p - 6$ generations by the transition rule of the tested cellular automaton in a configuration named F. The fitness function is the number of cells that are in the same state between configuration F and the schema of Figure 11.

- *Genetic operators.* The mutation function simply consists of mutating one bit among the bits corresponding to neighborhood states not used by the glider gun G. The rates 1, 5, and 10 percent are tried, together with three crossover operators.

https://doi.org/10.25088/ComplexSystems.17.4.321

  - *No crossover.* The evolutionary algorithm is tried without crossover.
  - *Central point.* A single point crossover with a locus situated exactly on the middle of the genotype is tried.
  - *Random point.* Another kind of recombination is tried with a single point crossover with a locus randomly situated.

  A linear ranking selection and a binary tournament selection of size 2 are tried.

- *Evolution engine.* An elitist strategy in which the best individual of a population is kept and a nonelitist strategy in which the new population is made of only children are tried.

- *Stopping criterion.* The value of the fitness function and the generation of the best rule are stored. If after 10 new generations the evolutionary algorithm has not found a better rule it will stop.

  With this stopping criterion, an execution of the evolutionary algorithm stops after an average of 24 generations.

3.4 Tabu search

A random cellular automaton A that exhibits the glider gun G is generated, the same fitness function of the evolutionary algorithm measures the performance of this cellular automaton. All the cellular automata obtained by mutating one bit corresponding to neighborhood states not used by the glider gun G, are tested by the fitness function. The best one not in a list L of the last chosen cellular automaton is chosen to become the new cellular automaton A. Sizes of 10, 100, and 1000 are tried for the list L. The algorithm stops when the best cellular automaton, among those obtained by mutation, that is not in list L is not better than the current cellular automaton.

With this stopping criterion, an execution of the algorithm stops after an average of 25, 20, and 19 generations depending on the size of the list L.

3.5 Monte Carlo method

For all guns, one million randomly generated cellular automata are tested. A simulation of an AND gate is found for 13 percent of the glider guns using this method.

4. Results

4.1 Best parameters

In order to determine the best parameters, a simulation of an AND gate was searched for within the 751 guns using the evolutionary algorithm and Tabu search with each of the values of the parameters.

Evolutionary Algorithm

Mutation rate: 1

	Elitist		Non Elitist	
	Tournament	Ranking	Tournament	Ranking
No Crossover	12	9	30	5
Middle Crossover	10	14	31	38
Random Crossover	1	2	2	0

Mutation rate: 5

	Elitist		Non Elitist	
	Tournament	Ranking	Tournament	Ranking
No Crossover	14	10	8	1
Middle Crossover	11	10	10	14
Random Crossover	0	1	2	1

Mutation rate: 10

	Elitist		Non Elitist	
	Tournament	Ranking	Tournament	Ranking
No Crossover	5	7	7	3
Middle Crossover	8	7	12	9
Random Crossover	2	2	0	1

Taboo search

List of Size	10	100	1000
Number	8	12	13

Monte Carlo algorithm: 7

Table 1. Number of executions that find an AND gate simulation under a given combination of parameters or operators.

For each of the values of the parameters, the number of executions which find the simulation of an AND gate are shown in Table 1.

The best parameters, among those tested, are a mutation rate of 1, an elitist strategy, tournament selection, and central crossover. The good results of the central crossover can be explained by the fact that the first 51 neighborhood states determine the birth of cells, while the other 51

https://doi.org/10.25088/ComplexSystems.17.4.321

determine how they survive or die. The elitist strategy that kept the best individual of the population is better than the nonelitist strategy.

4.2 AND gate

With the best parameters, 279 simulations of an AND gate were found using one of the guns found in [33]. 38 percent of these guns provide the simulation of an AND gate.

The simulation of an AND gate is shown in Figure 14, generation after generation with two input streams representing the data streams '11' and '10'. The collision between the first glider emitted by the gun and the first glider of the first input stream begin at generation 5 and finish at generation 12 in which the rest of this collision has disappeared. A second collision begins at generation 15 and finishes at generation 22. The output stream '10', which is the result of the operation 11 *and* 10, is emitted.

5. Synthesis and perspectives

This paper deals with the emergence of computation in complex systems with local interactions. An evolutionary algorithm searching for collision-based computing in cellular automata has been presented. Simulations of AND gates have been inspired by the simulation of an AND gate by the Game of Life and uses a given gun among those presented in [33].

An evolutionary algorithm is used to modify the transition rules of cellular automata in order to find the simulation of an AND gate. Different evolution strategies and different types of selection, crossover, and mutation operators were explored for this evolutionary algorithm. With the best parameters, the evolutionary algorithm finds a simulation of an AND gate for 38 percent of the guns found in [33]. The automatic simulation of an AND gate represents a significant contribution to the area of complex systems that considers computational theory.

All cellular automata that exhibit a simulation of an AND gate may be potential candidates for being universal cellular automata. Further goals are now to develop an automatic system for the simulation of a NAND gate and the demonstration of universal cellular automata. Then, another domain that seems worth exploring is how this approach could be extended to cellular automata with more than two states.

Future work could also evaluate all discovered cellular automata and calculate for each of them some rule-based parameters, for example, Langton's lamda [34]. All cellular automata simulating an AND gate may have similar values for these parameters that could lead to answering the question: Where are the edges of computational universality? Knowing this may lead to a better understanding of the emergence of computation in complex systems with local interactions.

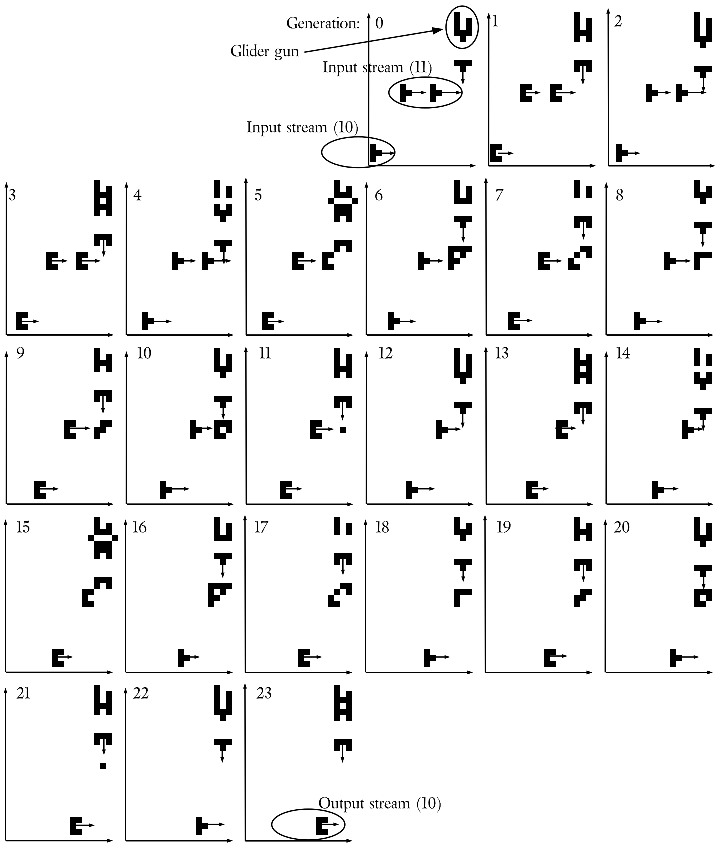


Figure 14. Simulation of an AND gate generation after generation. The transition rule of the cellular automata that exhibit this gate can be denoted as: $s1e2aiv3jreqk4akrjze5kyqv6eca/b2ck3ayqrk4yiwkrtj5yqjvai6vi7c$.

Acknowledgments

The work was supported by the Engineering and Physical Sciences Research Council (EPSRC) of the United Kingdom, Grant EP/E005241/1.

References

[1] S. Wolfram, *A New Kind of Science* (Wolfram Media, Inc., Champaign, IL, 2002).

[2] J. von Neumann, *Theory of Self-Reproducing Automata* (University of Illinois Press, Urbana, IL, 1966).

[3] S. Wolfram, "Universality and Complexity in Cellular Automata," *Physica D*, **10** (1984) 1–35.

[4] E. Banks, *Information and Transmission in Cellular Automata*, Ph.D. Thesis, MIT, 1971.

[5] N. Margolus, "Physics-Like Models of Computation," *Physica D*, **10** (1984) 81–95.

[6] K. Lindgren and M. Nordahl, "Universal Computation in Simple One-Dimensional Cellular Automata," *Complex Systems*, **4** (1990) 299–318.

[7] K. Morita, Y. Tojima, I. Katsunobo, and T. Ogiro, "Universal Computing in Reversible and Number-Conserving Two-Dimensional Cellular Spaces," in *Collision-Based Computing*, edited by A. Adamatzky (Springer-Verlag, 2002).

[8] A. Adamatzky, "Universal Dynamical Computation in Multi-Dimensional Excitable Lattices," *International Journal of Theoretical Physics*, **37** (1998) 3069–3108.

[9] M. Gardner, "The Fantastic Combinations of John Conway's New Solitaire Game 'Life'," *Scientific American*, **223** (1970) 120–123.

[10] E. Berlekamp, J. Conway, and R. Guy, *Winning Ways for Your Mathematical Plays* (Academic Press, New York, 1982).

[11] S. Wolfram and N. Packard, "Two-Dimensional Cellular Automata," *Journal of Statistical Physics*, **38** (1985) 901–946.

[12] F. Glover, "Future Paths for Integer Programming and Links to Artificial Intelligence," *Computers and Operations Research*, **13** (1986) 533–549.

[13] J. Holland, *Adaptation in Natural and Artificial Systems* (University of Michigan, Ann Arbor, MI, 1975).

[14] N. Packard, "Adaptation Toward the Edge of Chaos," in *Dynamic Patterns in Complex Systems*, edited by J. A. S. Kelso, A. J. Mandell, and M. F. Shlesinger (World Scientific, 1988).

[15] M. Mitchell, J. Crutchfield, and P. T. Hraber, "Evolving Cellular Automata to Perform Computations: Mechanisms and Impediments," *Physica D*, **75** (1994) 361–391.

[16] M. Mitchell, P. Hraber, and J. Crutchfield, "Revisiting the Edge of Chaos: Evolving Cellular Automata to Perform Computations," *Complex Systems*, **7** (1993) 89–130.

[17] W. Hordijk, J. Crutchfield, and M. Mitchell, "Mechanisms of Emergent Computation in Cellular Automata," in *Parallel Problem Solving from Nature-V*, edited by A. E. Eiben, T. Bäck, M. Schoenauer, and H.-P. Schwefel (Springer-Verlag, 1998).

[18] R. Das, J. Crutchfield, M. Mitchell, and J. Hanson, "Evolving Globally Synchronized Cellular Automata," *Proceedings of the Sixth International Conference on Genetic Algorithms* (Morgan Kaufmann, San Francisco, CA, 1995).

[19] M. Sipper, "Evolution of Parallel Cellular Machines," in *Annual Reviews of Computational Physics V*, edited by D. Stauffer (World Scientific, 1997).

[20] D. Andre, J. Koza, F. H. Bennet III, and M. Keane, *Genetic Programming III: Darwinian Invention and Problem Solving* (Morgan Kaufmann, San Francisco, CA, 1999).

[21] J. Koza, *Genetic Programming: On the Programming of Computers by Means of Natural Selection* (MIT Press, Cambridge, MA, 1992).

[22] A. Wuensche, Discrete Dynamics Lab (ddlab), www.ddlab.org, 2005.

[23] G. Martinez, A. Adamatzky, and H. McIntosh, "Phenomenology of Glider Collisions in Cellular Automaton Rule 54 and Associated Logical Gates Chaos," *Fractals and Solitons*, **28** (2006) 100–111.

[24] D. Eppstein, http://www.ics.uci.edu/ eppstein/ca/, Apr. 2007.

[25] E. Sapin, O. Bailleux, J. Chabrier, and P. Collet, "A New Universal Automata Discovered by Evolutionary Algorithms," *Gecco2004. Lecture Notes in Computer Science*, **3102** (2004) 175–187.

[26] J. Ventrella, "A Particle Swarm Selects for Evolution of Gliders in Non-Uniform 2d Cellular Automata," in *Artificial Life X: Proceedings of the Tenth International Conference on the Simulation and Synthesis of Living Systems*, edited by L. M. Rocha, L. S. Yaeger, M. A. Bedau, D. Floreano, R. L. Goldstone, and A. Vespignani (MIT Press, 2006).

[27] J. Lohn and J. Reggia, "Automatic Discovery of Self-Replicating Structures in Cellular Automata," *IEEE Transactions on Evolutionary Computation*, **1** (1997) 165–178.

[28] E. Sapin, O. Bailleux, and J. Chabrier, "Research of Complex Forms in the Cellular Automata by Evolutionary Algorithms," *EA03. Lecture Notes in Computer Science*, **2936** (2004) 373–400.

[29] E. Sapin, O. Bailleux, J. Chabrier, and P. Collet, "Demonstration of the Universality of a New Cellular Automaton," *International Journal of Unconventional Computing*, **3** (2007) 79–103.

[30] E. Sapin, "Approche evolutionniste de la recherche d'automates cellulaires universels," *TSI*. To be published.

[31] E. Sapin, O. Bailleux, and J. Chabrier, "Research of Complexity in Cellular Automata through Evolutionary Algorithms," *Complex Systems*, **17**(3) (2007) 231–241.

[32] E. Sapin, O. Bailleux, and J. Chabrier, "Research of a Cellular Automaton Simulating Logic Gates by Evolutionary Algorithms," *EuroGP03. Lecture Notes in Computer Science*, **2610** (2003) 414–423.

[33] E. Sapin, L. Bull, and A. Adamatzky, "Automatic Discovery of Self-Replicating Structures in Cellular Automata," *IEEE Transactions on Evolutionary Computation*, 2007.

[34] C. Langton, "Computation at the Edge of Chaos," *Physica D*, **42** (1990) 12–37.