Research of Complexity in Cellular Automata through Evolutionary Algorithms

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

Olivier Bailleux
Jacqueline Chabrier
Universite de Bourgogne, 9 avenue A. Savary, B.P. 47870,
Dijon, 21078 Cedex, France

This paper presents an evolutionary approach to searching for cellular automata that accept gliders. The proposed technique is based on a specific fitness function taking into account spatial evolution, the number of living cells, and the presence of gliders. The results show that the genetic algorithm is a promising tool for finding cellular automata with specific behaviors, and then could prove to be decisive for identifying new automata that support universal computation.

1. Introduction

Cellular automata are discrete systems in which a population of cells evolve from generation to generation on the basis of local transition rules. They can simulate simplified forms of life [1, 2] or physical systems with discrete time and space and local interactions [3–5].

Wolfram showed that one-dimensional cellular automata can present a large spectrum of dynamic behaviors. In [6] he introduces a classification of cellular automata, comparing their behavior with that of some continuous dynamic systems. He specifies four classes of cellular automata on the basis of qualitative criteria.

For all initial configurations, class 1 automata evolve after a finite time to a homogeneous state where each cell has the same value. Class 2 automata generate simple structures where some stable or periodic forms survive. The class 3 automata's evolution leads, for most initial states, to chaotic forms. All other automata belong to class 4. According to Wolfram, automata of the class 4 are good candidates for universal computation.

The only binary automaton currently identified as supporting universal computation is Life, which is, besides, in class 4. Its ability to simulate a Turing machine is proved in [7] in a constructive way, using

Complex Systems, **17** (2007) 231–241;

https://doi.org/10.25088/ComplexSystems.17.3.231

gliders (i.e., periodical patterns which, when evolving alone, are reproduced identically by shifting in space) to carry information and to realize logical gates through collisions. The identification of new automata capable of generating gliders is consequently a possible lead in the search for new automata that support universal computation.

In the spirit of the work described in [8–10] about one-dimensional automata, this paper presents an evolutionary algorithm discovering new rules capable of spontaneously generating gliders. This algorithm uses a fitness function detecting the birth of gliders in a primordial soup.

Section 2 describes the framework, including the representation of the transition rules. Section 3 details selection, crossover, mutation operators, and the fitness function that allowed us to obtain experimental results presented in section 4. In section 5, we present a synthesis of the results and several research perspectives.

2. Framework

In this study we only look into cellular automata with the following specifications.

- Cells have two possible values: 0 or 1.
- They evolve in a two-dimensional matrix, called the *universe*.
- Transition rules only take into account the eight direct neighbors of a cell for the current generation, so as to determine its states for the next generation.

We call the *context* of a cell the state of the cell and its eight neighbors. A cell thus can have 512 different contexts. A transition rule is defined as a boolean function that maps each of the 512 possible contexts to the value which will be taken by the concerned cell at the next generation. Therefore the underlying space of automata includes 2^{512} rules.

Let us recall that there is a more limited space, known as Bays space [11–14], where a transition rule is specified by a *EbEh/FbFh* quadruplet. A cell survives to the next generation if and only if its number of neighbors for the current generation is included between *Eb* and *Eh*. A cell is born during the next generation if and only if its number of neighbors at the current generation is included between *Fb* and *Fh*. Bays space only includes 1296 rules, which simplifies the exhaustive study of corresponding automata. However, some rules presenting complex behaviors such as the one presented in [15], where a cell survives if and only if it has one or three living neighbors, do not belong to Bays space.

3. Genetic evolution

This section describes the use of an evolutionary algorithm for the search of new rules capable of generating gliders (cf. Figure 1). Because con-

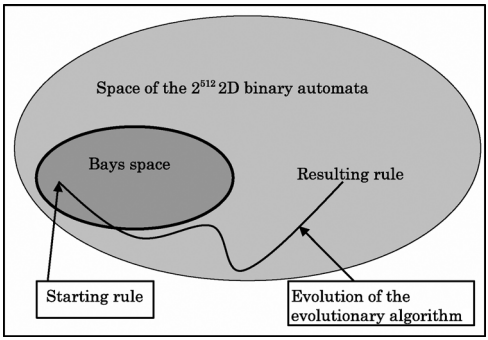

Figure 1. Representation of the evolutionary process allowing the creation of a new rule.

fusion may exist between the evolutionary algorithm and cellular automaton generations, we use the word "transition" for the generations of an automaton.

3.1 Encoding

Rules have been encoded by 512-bit strings. The value of the bit related to each of the 512 possible contexts is the value taken by the considered cell at the time of the next transition of the cellular automaton.

For example, Figure 2 represents the rule 35/33 of Bays space. The rectangle pointed to by the arrow represents a context that leads to the birth of a cell at the following generation, which is shown as a point on the right of the context's representation. If the bit related to this context

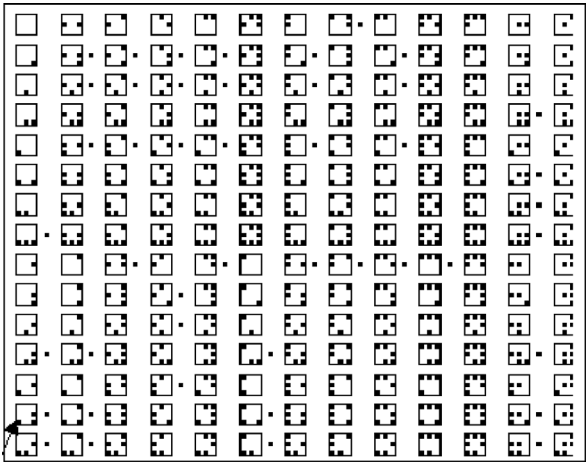

Figure 2. Relevant section of representation of the rule 35/33.

https://doi.org/10.25088/ComplexSystems.17.3.231

in the string undergoes a mutation, this context will no longer provoke the presence of a living cell.

3.2 Initialization and selection

The evolutionary algorithm manages a population of 50 rules. The initial population comes from a rule R of the Bays space. It includes 10 occurrences of R, 10 variants of R obtained through one mutation, 10 rules obtained through two mutations of R, 10 rules obtained through three mutations of R, and 10 rules obtained through four mutations of R. A ranking selection operator [16] is used, with conservation of the 20 best rules. At each generation, the population stemming from the selection stage is completed with 20 rules obtained through mutation and 10 obtained through crossover.

3.3 Crossover

The 20 best rules are dispatched randomly into 10 couples from which stem 10 new rules that are added to the population. These new rules are generated by a simple crossover operator at a median point.

3.4 Mutation

A mutation consists of modifying a randomly chosen bit, with the same weight for each of the 512 bits of a rule.

We noted that only one mutation per rule for each generation was not sufficient to observe a convergence toward automata capable of producing gliders. Therefore we chose a more aggressive mutation strategy, aimed at maintaining some diversity in the population, inspired by the research work of Lee and Takagi in [5] and Sefrioui and P'eriaux in [19]. For each couple used for crossover, the Hamming distance between both rules determines the number of mutations. If this distance is greater than or equal to 5 then only one mutation is applied to each rule. Otherwise five mutations are applied to each rule.

3.5 Fitness function

The fitness function is based on the evolution, during 300 transitions, of a "primordial soup," randomly generated in a square of $40*40$ centered in a $200*200$ space. For each transition, the dimensions of the smallest rectangle containing all the living cells, which is called the *including rectangle*, are measured. In the presence of gliders this rectangle grows larger over the transitions.

During evolution of the automata, the algorithm counts the number n_1 of times the area of this rectangle increases and the number n_2 of times it decreases. For L transitions of the automata we define:

$$S_1 = \frac{n_1 - n_2 + L}{L}. \tag{1}$$

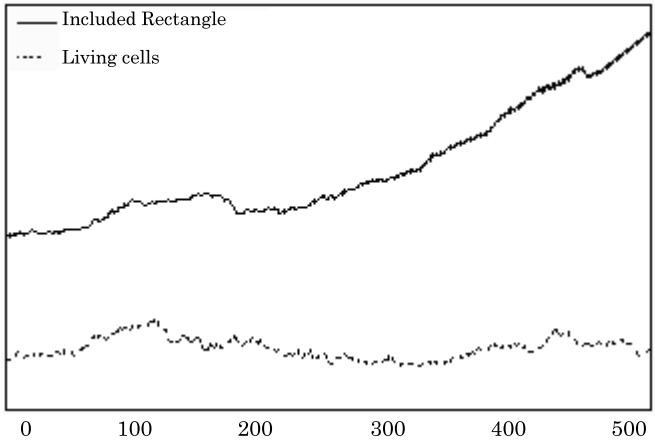

Figure 3. Generation-by-generation average of the surface of the including rectangle and the number of living cells during 500 generations for the evolution of Life.

This score represents the ability of the automata to spread through the space.

At each transition, the total number of living cells is also taken into account. This number, in Life, tends to remain stable, as verified with Figure 3 where the total number of cells and the surface of the including rectangle are represented for Life. Given that Life is the only automaton known as complex and supporting gliders, we chose to take into account this characteristic in our fitness function.

This propensity to have a stable number of cells is estimated by a second s_2 score. During the evolution of the automata, the algorithm counts the number m_1 of times the total number of cells decreases and the number m_2 of times it increases. For an automaton having evolved for L transitions, S_2 is defined by:

$$S_2 = \frac{m_1 - m_2 + L}{L}. \tag{2}$$

With a first fitness function $S_1 * S_2$, we observed a convergence toward rules with good S_1 and S_2 scores but they do not accept gliders. In these rules, the increase of the including rectangle is due to a global move of the living cells in a privileged direction. In order to avoid such behavior, we add a coefficient to the fitness function, reflecting a move of the center of gravity of the living cells. This coefficient is defined by $c_1 = 1 + dg/sz$, where dg is the euclidian distance between the center of gravity and the middle of the square universe of size sz.

Moreover, we add another coefficient c_2 in order to promote rules that allow the spontaneous apparition of gliders and periodic patterns.

https://doi.org/10.25088/ComplexSystems.17.3.231

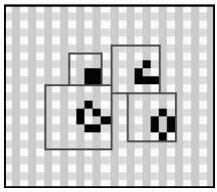

Figure 4. Group of isolated cells.

The presence of gliders and periodic patterns has been the subject of the following test, inspired by Bays' test in [11]: after the evolution of the automaton, from a primordial soup, each group of connected cells (see Figure 4) is isolated in an empty space and evolves during 20 transitions. For each transition, the original pattern is searched in the test universe. The following three cases can happen.

- The initial pattern has reappeared at its first location (it is then considered to be periodic).
- It has reappeared at another location (it is then considered to be a glider).
- It has not reappeared (it is then considered to be evolving).

Let g denote the number of appearances of gliders and p denote the number of appearances of periodic patterns. The coefficient c_2 is then defined as:

$$c_2 = 1 + v + \frac{s}{100}. \tag{3}$$

The fitness function becomes:

$$F = \frac{S_1 * S_2 * c_2}{c_1}. \tag{4}$$

The number of gliders plays a part in the new fitness function, but also the number of periodical patterns, so as to promote the automata that, even if not having gliders spontaneously emerge, accept nevertheless periodical patterns. These automata are privileged because we infer that, if they accept periodical patterns, these automata are able to evolve into rules that support gliders, thanks to the evolutionary algorithm.

4. Results

4.1 Notation

For conciseness and readability, the following convention has been adopted for presenting the rules. A rule R' stemming from the evolution of an initial rule R will be noted:

$$(EbEh/FbFh)\{m_k\}_{0<k<n} \tag{5}$$

256	128	64
32	16	8
4	2	1

Table 1. The associated weight of the neighbors of a cell used to evaluate the rank of the context of this cell.

where (*EbEh*/*FbFh*) is the representation of R in Bays space, n the number of contexts in which R and R' differ, and the set mk corresponds to the ranks of these contexts. The rank of a context corresponds to the place of the related bit in the string of the rule. It is calculated by giving the weight, shown in Table 1, to the living cells.

For instance, the rank of the context pointed at by the arrow in Figure 2 is 13, corresponding to 1+4+8.

4.2 New complex rules

The following three rules, that accept gliders, were obtained from our evolutionary algorithm. These three rules can be found in [19] in mcl format [20].

The rule (12/33) {210, 156, 432, 312, 211, 242, 188, 179, 181, 433, 369, 372, 313, 311, 72, 73, 226, 452, 141, 142, 172, 163, 165, 166, 353, 356, 300, 227, 453, 173, 391}, denoted R_1, presents 31 mutations and was obtained after 59 generations. It allows many stable and periodic patterns to emerge. Some examples are presented in Figure 5.

Several gliders also appear, such as the one presented in Figure 6, that moves 12 squares horizontally while four vertically. This glider also leaves a trace behind. Such a pattern is more exactly called a "puffer" [1] (i.e., stable patterns subsisting after the passage of a periodically displacing pattern). This pattern, going through 12 squares during 40 transitions, then has a speed of 3/10.

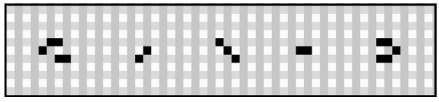

Figure 5. Stable patterns and periodic patterns of R_1.

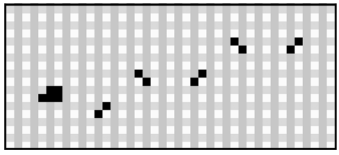

Figure 6. Puffer of R_1.

https://doi.org/10.25088/ComplexSystems.17.3.231

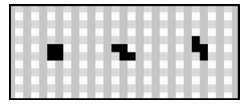

Figure 7. Some stable and periodic patterns for the R_2 rule.

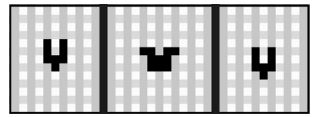

Figure 8. Evolution of a glider during one period for the R_2 rule.

Another interesting result was obtained by letting the rule 35/33 evolve for 350 generations, ending up with the rule (35/33) {49, 83, 124, 187, 381, 164, 204, 450, 102, 488, 271, 399}, denoted R_2. This rule offers a large sample of stable and periodic patterns (some of them are presented in Figure 7).

A glider, which is able to move horizontally and from top to bottom following its initial direction, is represented in Figure 8. It has a period of 2 and a speed of 1/2. We should note that this glider exists in the rule 35/33 [17], but the mutations have strongly augmented its appearance potentiality.

Another interesting example is the rule 22/33 {144, 80, 24, 17, 18, 20, 48, 272, 154, 432, 72}, denoted R_3, obtained after 30 generations. Two gliders, with a period of 11 and 4, are presented in Figures 9 and 10 respectively. Many periodic patterns, such as those presented in Figure 11, also exist in this rule.

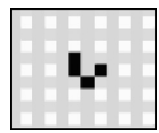

Figure 9. Glider of R_3 with a period 11, π angle, at a speed of 1/11.

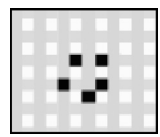

Figure 10. Glider of R_3 with a period of 4, $-\pi/2$ angle, at a speed of 1/4.

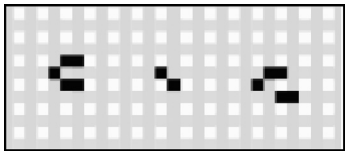

Figure 11. Stable and periodic patterns of R_3.

5. Related Works

In [8, 9], Das, Mitchell, and Crutchfield used genetic algorithms to evolve cellular automata to perform computational tasks that require global information processing. They studied one-dimensional binary-state cellular automata for two computational tasks: density classification and synchronization. For the density classification task, the goal is to find a cellular automaton that decides whether or not the initial configuration contains a majority of ones (i.e., has a high density). [15] shows an embedded-particle framework capturing the main information processing mechanisms of the emergent computation presented in [8] and [9].

In [21], genetic programming is also applied to evolve cellular automata for simple random number generation.

6. Synthesis and perspectives

In this paper, we proposed an evolutionary approach in the research of cellular automata that accept gliders, based on a classical scheme of genetic algorithm with binary representation of the transition rules. We used a specific fitness function taking into account the spatial evolution, the number of living cells, and also the presence of gliders.

Starting from initial populations of rules without the apparition of gliders, we have noticed the emergence of new rules that accept gliders. This constitutes a first experimental contribution to the research of new cellular automata that support universal computation. Beyond this punctual result, the evolutionary approach proves to be very promising for the research of two-dimensional complex automata presenting specific behaviors.

However, in spite of the precautions taken in the fitness function, the mutation and crossover operators used permit the emergence of nonisotropic rules, and in particular, unidirectional gliders.

We plan to adapt the mutation operator, in order to keep the isotropy of rules. The genetic algorithm would then be able to find isotropic, complex two-dimensional automata with two states that accept gliders.

https://doi.org/10.25088/ComplexSystems.17.3.231

We also plan to expand the research, starting from an initial population consisting of any rules (i.e., not only from Bays space), for example, by using randomly generated rules.

Later on, our aim is the research of new transition rules that allow the realization of glider guns, logical operators, and simulation of a Turing machine. One research direction may be a co-evolution of patterns and rules.

References

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

[2] M. Gardner, "On Cellular Automata, Self-Reproduction, the Garden of Eden, and the Game of Life," *Scientific American*, **224** (1971) 112–118.

[3] C. Dytham and B. Shorrocks "Selection, Patches and Genetic Variation: A Cellular Automata Modeling Drosophila Populations," *Evolutionary Ecology*, **6** (1992) 342–351.

[4] I. R. Epstein, "Spiral Waves in Chemistry and Biology," *Science*, **252** (1991) 67.

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

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

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

[8] R. Das, M. Mitchell, and J. P. Crutchfield, *Evolving Cellular Automata with Genetic Algorithms: A Review of Recent Work*, Proceedings of the First International Conference on Evolutionary Computation and Its Applications (EvCA'96), held in 1996, edited by Russian Academy of Sciences.

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

[10] W. Hordijk, J. P. Crutchfield, and M. Mitchell, *Evolving Cellular Automata with Genetic Algorithms: A Review of Recent Work*, Proceedings of the First International Conference on Evolutionary Computation and Its Applications (EvCA'96), held in 1996, edited by Russian Academy of Sciences.

[11] C. Bays, "Candidates for the Game of Life in Three Dimensions," *Complex Systems*, **1** (1987) 373–400.

[12] C. Bays, "A Note on the Discovery of a New Game of Three-dimensional Life," *Complex Systems*, **2** (1988) 255–258.

[13] C. Bays, "The Discovery of a New Glider for the Game of Three-dimensional Life," *Complex Systems*, **4** (1990) 599–602.

[14] C. Bays, "A New Candidate Rule for the Game of Three-dimensional Life," *Complex Systems*, **6** (1992) 433–441.

[15] J. C. Heudin, "A New Candidate Rule for the Game of Two-dimensional Life," *Complex Systems*, **10** (1996) 367–381.

[16] L. Davis, *The Genetic Algorithm HandBook* (Van Nostrand Reinhold, New York, 1991).

[17] M. A. Lee and H. Takagi, "Dynamic Control of Genetic Algorithms Success using Fuzzy Logic Techniques," *Proceedings of the Fifth International Conference on Genetic Algorithms*, held in 1993, edited by S. Forrest.

[18] M. Sefrioui and J. P'eriaux, "Fast Convergence Thanks to Diversity," in *Proceedings of the Fifth Annual Conference on Evolutionary Programming*, held in 1996.

[19] E. Sapin, http://uncomp.uwe.ac.uk/sapin/complexsystems/rules.zip.

[20] M. Wojtowicz, http://psoup.math.wisc.edu/mcell/index.html.

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

https://doi.org/10.25088/ComplexSystems.17.3.231