

## Generating Multi State Cellular Automata by using Chua's "Universal Neuron"

Eleonora Bilotta and Gianpiero Di Blasi

*Department of Linguistics, University of Calabria,*

*Arcavacata Rende, 87036, Italy*

*E-mail: bilotta, gdibiasi@unical.it*

*www.unical.it*

Sebastiano Giambó

*Department of Mathematics, University of Messina*

*Messina, 98166, Italy*

*E-mail: giambo@mat520.unime.it*

Pietro Pantano

*Department of Mathematics, University of Calabria,*

*Arcavacata Rende, 87036, Italy*

*E-mail: piepa@unical.it*

*www.unical.it*

In 1999 Chua demonstrated that it is possible to obtain boolean Cellular Automata (CA) by using Cellular Neural Networks (CNNs), implemented as chips, which definitely reduce the time of simulation for discrete dynamical systems such as CA and allow for a better understanding of complexity. In this work we demonstrate that Chua's Universal Neuron which simulates boolean CA can be generalized for multistate CA. This new approach allows to investigate the nature of the CA rule space, in relationship with the Universal Neuron parameter space, establishing relationships between discrete and continuous dynamical systems and analysing the nature of local and global rules in affecting the behaviour of these systems. The method which we have developed is fruitful since we have found a lot of CA which can be considered complex rules of class IV, in the Wolfram classification. Furthermore we have used the idea of genetical computation in considering the values of the Universal Neuron parameter space as a sort of genotype which determines and produces variations in the CA behaviour. On this basis, we have implemented a genetic algorithm which fits very well with the searching for CA complex rules. The nature of the relationship between continuous and discrete systems is a very important topic in the Science of Complexity which helps us to clarify the

temporal dimension of many biological phenomena and processes.

*Keywords:* Style file; L<sup>A</sup>T<sub>E</sub>X; Proceedings; World Scientific Publishing.

## 1. Introduction

Introduced by von Neumann in the late 1950's<sup>1</sup> Cellular Automata (CA) are one of the paradigms of the Science of Complexity, in the program aimed at investigating the "logic of life" and recreating the structure of life in a synthetic artificial world.<sup>2-4</sup> In fact, the automatic treatment of the self-reproducing process has become a mathematical problem; the "logic of life" or rather, how to extract mathematical algorithmic structures from biological phenomena, structures which can be useful to understand the biology of the phenomena and their possible reinvention in digital worlds. CA are used in many domains of contemporary science from Physics to Biology, from Geology to Social sciences since they represent simple programs which can simulate and explain the emergence of a great quantity of behaviours from simple to complex to chaotic. For a systematic review of this topic see [5].

The nature of CA is discrete since the system is composed of cells that change their state according to the states of their neighbours. Nevertheless, the great amount of different behaviours, which CA manifest, can be considered as some typically continuous phenomena, such as wave propagation, which are usually known as solitons.<sup>6</sup> The behaviour of multistate CA (Cellular Automata whose states can assume values different from 0 and 1, especially complex multistate CA) shows surprising analogies with the behaviour of non-linear waves in continuous systems. Furthermore, if we assume that the cell state of a CA is real number, it is possible to use these systems as possible models of partial derivative equations.<sup>5</sup> In the same time, the study of CA behaviour can shed light on wave propagation phenomenon or create a tractable simulation environment of complex processes or consider CA as approximate models of continuous phenomena non tractable with classical methods. In 1999 Chua demonstrated that it is possible to obtain boolean Cellular Automata (CA) by using Cellular Neural Networks (CNNs), implemented as chips which definitely diminishes the time of simulation for discrete dynamical systems as CA and allows for a better understanding of complexity. Introduced by Chua in 1988, CNNs have been used first as pattern recognition systems. They are a collection of cells, connected to other cells in a neighbourhood, where each cell is a small non-linear Chua's circuit. From a technological point of view, CNNs

currently represent a method for creating a cell array of meaningful size for image processing and other applications. In fact it is possible to arrange more than 4 million of transistors into a  $128 \times 128$  CNN chip on 1 square centimeter area of silicon.<sup>7</sup> This chip is called the CNN Universal Chip. Their speed in the computational process is 500 times higher than that of other computational systems. CNNs can be used also to study reaction-diffusion processes and non-linear wave propagation phenomena.<sup>8</sup>

Despite their apparent simple criteria of evolution, CA can display rich and complex patterns, whose organization is completely unpredictable. Wolfram<sup>9</sup> classified Cellular automata rules qualitatively according to their asymptotic behaviour: class I (homogeneity); class II (periodicity); class III (chaos); class IV (complexity). Complex rules are important since they perform computation, transmission, storage and modification of information, like biological systems and life. Furthermore, universal computation requires memory and communication over long distance in space and time. Therefore complex computation requires long transient and space-time correlation lengths, typical of complex rules of class IV. Langton found that complex CA rules can be found in this restricted region, called the Edge of Chaos and that the complex dynamics of systems in the vicinity of a phase transition rest on a fundamental capacity for producing information. In some related works<sup>10,11</sup> we have used genetic algorithms for searching complex multistate CA rules. We have found a lot of self-reproducing systems in 2D CA,<sup>12,13</sup> which can be considered as proto-organisms for structure replication that provide insight into analogous processes in the biological world.

In this work, starting from results obtained by Chua<sup>14,15</sup> we demonstrate that multistate CA can be obtained starting from a universal neuron, subject to a process of mutation.

The work is organized as follows. In the second section we introduce formal aspects of CA, while in section 3 we illustrate the method that Chua has used for simulating Boolean CA. In section 4 we generalize this method and apply it to multistate CA. Finally, section 5 contains some examples of generation of complex rules by using this method. Conclusion on the discrete-continuous organization finishes this work.

## 2. CA formal aspects

The environment considered is a one-dimensional CA, which can be thought as the following tuple:

$$A = (d, S, N, f) \quad (1)$$

where  $d$  is a positive integer that indicates the CA dimension (one, two, three or more),  $S$  a finite set of  $k$  states,  $N = (x_1, \dots, x_n)$  is a neighbourhood vector of  $n$  different elements of  $Z^d$ ,  $f$  is a local rule defined as:

$$f : S^n \rightarrow S \quad (2)$$

In our case  $d = 1$ , and the neighborhood identifies the cells with a local interaction ray  $r$ , so (2) associates to the  $(2r + 1)$  elements of  $S$  another element of  $S$ , that is

$$(\dots s_i \dots) \mapsto s_i \quad (3)$$

A rule that discriminates all possible cases is expressed in exhaustive form, and considers all  $k^{(2r+1)}$  possible cases. In Elementary Cellular Automata (ECA),  $S = \{0, 1\}$ , for each cell  $i$   $N$  is composed of the cell itself and of its right and left neighbors. An example of how the local rule (2) works is represented in the following table:

$$\begin{bmatrix} 111 & 110 & 101 & 100 & 011 & 010 & 001 & 000 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \quad (4)$$

A visual example of table (4) is presented in Figure 1, Table 4 is also called



Fig. 1. Visual representation of table 4.

CA rule table. A rule table can be represented in a synthetic form as a sequence of 8 bits in the form reported in (5).

$$l = (10000001) \quad (5)$$

Each ECA has a local rule which can be represented by using a string of 8 bits and is univocally represented by a number between 0 and 255, which is the decimal corresponding to the string 1.

### 2.1. The Universal Neuron

Following Chua,<sup>14</sup> it is possible to represent the rule table as a cube whose vertices are white or black (Figure 2). A white or black sign is assigned to each vertex of the boolean cube according to whether the corresponding value of the bit in the string is white or black. A vector  $u_i$  with  $i = 0, 1, \dots, 7$ ,

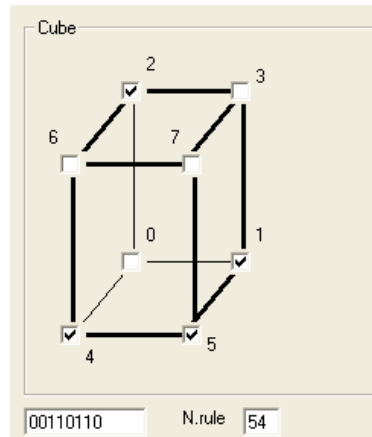


Fig. 2. The string that identifies the ECA rule 54 is visualized in the bottom on the left side of the figure. At each vertex of the boolean cube we have put a black (=0) or white(=1) dot according to the string values. Vertex 0 corresponds to the last position in the string; Vertex 7 corresponds to the first position in the string.

corresponds to each vertex of the cube. The components of the vectors are:

$$\begin{cases} u_0 \equiv (-1, -1, -1), & u_1 \equiv (-1, -1, 1), & u_2 \equiv (-1, 1, -1), & u_3 \equiv (-1, 1, 1) \\ u_4 \equiv (1, -1, -1), & u_5 \equiv (1, -1, 1), & u_6 \equiv (1, 1, -1), & u_7 \equiv (1, 1, 1) \end{cases} \quad (6)$$

The projection of these vectors on another vector  $b \equiv (b_1, b_2, b_3)$ , will detect dots marked in white and black, according to the color of the preceding vertex. Figure 3 shows these projections. Following Chua, we indicate by  $\sigma_i$  the scalar product of  $b$  and  $u_i$ :

$$\sigma_i = b \cdot u_i \quad (7)$$

It is important to note that it individuates in a precise way the neighborhood configuration in the rule table. Let us build a discriminating function (*off-set level*) in the following way:

$$w(\sigma) = z_2 \pm [|z_1 \pm |z_0 + \sigma||] \quad (8)$$

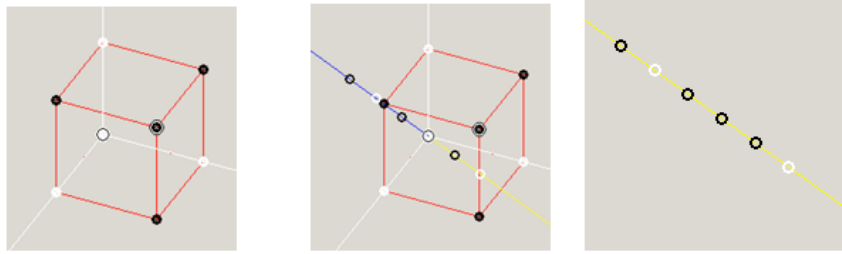


Fig. 3. The vertex of the boolean cube are projected on the vector  $b$ .

where  $z_0, z_1, z_2$  are integers. Consequently, the discriminating function depends on 8 parameters  $z_2, z_1, z_0, b_1, b_2, b_3, v$ , and two integers which can assume the values  $\pm 1$ . After having fixed the set of 8 parameters, a value of the discriminating function equal to  $w(\sigma_i)$  will consequently correspond to each local rule  $i = 0, 1, \dots, 7$ .

Let us consider the following dynamical system:

$$\dot{x} = g(x) + w(\sigma_i) \quad (9)$$

where

$$g(x_i) = -x_i + |x_i + 1| - |x_i - 1| \quad (10)$$

is called *driving function*. For each value of  $\sigma_i$ , (9) individuates 8 different dynamical systems. If we consider as initial data of (9)  $x(0) = 0$ , according to the sign of  $w(\sigma)$ , (9) will have a unique attractor, positive if  $w(\sigma) > 0$ , and negative in the opposite case. In Figure 4 some attractors generated by the system (9) for different values of the discriminating function are presented. From what we have discussed above, the following fundamental theorem stems:

- The state  $x(t)$  of a dynamical system like (9), with initial condition  $x(0) = 0$ , converges monotonically toward a positive attractor  $Q+$  if  $w > 0$  and toward a negative attractor  $Q-$  if  $w < 0$ .

Let us define the following value  $y_i$  as output of the dynamical system (9)

$$y_i = \text{sgn}w(\sigma_i) \quad (11)$$

We have built a machine based on the dynamical system (9) which takes as input a neighbourhood  $u_i$  and gives as output  $y_i$ . In this way we can generate rule table like table (4) or (5). This machine is ruled by the set

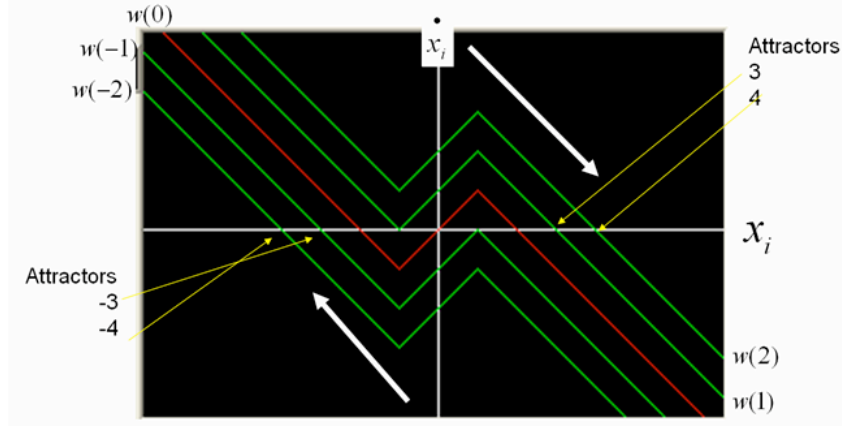


Fig. 4. In this figure, some attractors generated by the system (9) for different values of the discriminating function are represented.

of 8 parameters  $z_2, z_1, z_0, b_1, b_2, b_3, c_1, c_2$ , where  $c_1$  and  $c_2$  are two integers which can assume the values  $\pm 1$  (See Figure 5):

$$y_i = \text{sgn} \{ z_2 + c_2 [|z_1 + c_1 |z_0 + b \cdot u_i|] \} \quad (12)$$

Since dynamical systems like (9) are easily implemented on chips as CNN,

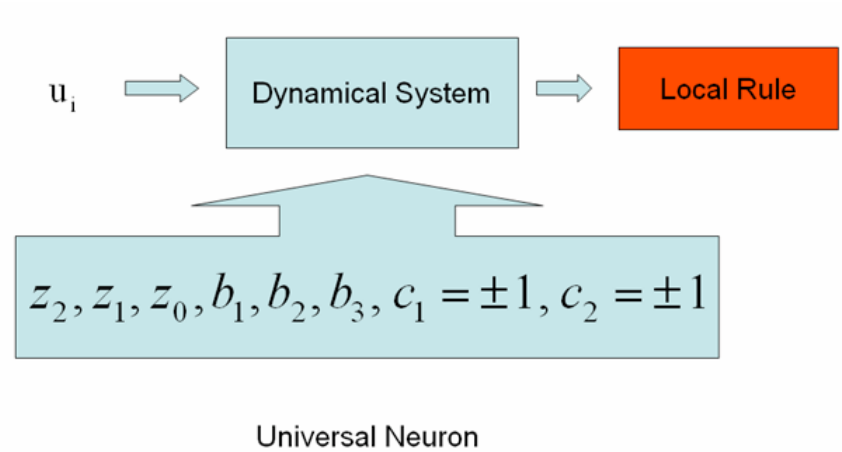


Fig. 5. The universal neuron computes the data of the neighbourhood and gives as output the corresponding local rule.

it is possible to create analogous systems that simulate the CA behaviour and definitely reduce the time of simulation for discrete dynamical systems such as CA.<sup>14</sup> Associating parameters to the universal neuron, it is possible to generate the whole set of the ECA rules. In Table 1 some rules and the corresponding universal neuron values are reported. It is important to

Table 1. In this Table, the values of the universal neuron for some ECA are reported .

N. Rule	$z_2$	$c_2$	$z_1$	$c_1$	$z_0$	$b_1$	$b_2$	$b_3$
0	1	-1	1	1	2	-1	1	0
1	3	-1	1	1	3	1	1	1
2	2	-1	1	-1	-1	1	1	4
3	2	-1	0	1	3	1	1	0
5	2	-1	3	-1	1	2	0	2
6	4	-1	2	1	3	3	1	1
9	2	-1	0	1	3	3	1	-1
10	3	-1	1	1	2	3	0	1
12	3	-1	2	1	2	3	1	0
13	2	-1	0	1	3	2	-1	1
14	2	-1	0	-1	2	3	1	1
15	2	-1	1	-1	3	3	2	0
16	2	-1	0	-1	3	2	1	4
17	2	-1	4	-1	1	0	3	3
18	1	-1	2	-1	1	2	1	4
20	1	-1	0	-1	2	3	3	2
21	3	-1	0	-1	3	1	1	2
22	1	-1	1	-1	2	1	2	2
24	2	-1	1	1	0	3	1	2
26	3	-1	1	-1	1	4	2	3
32	2	-1	0	1	1	1	4	1
34	3	-1	1	1	2	0	4	1
36	1	-1	0	1	0	-1	1	2

underline that the parameter space of the universal neuron is bigger than the ECA rule space and that to one ECA rule can correspond more than one universal neuron, that is to say more dynamical systems like that presented in (9).

### 3. Generating multistate CA

The method described above can simply be generalized to the multistate CA. We define multistate cellular automata systems in which the state of the CA cell can assume integer values from 0 to  $k - 1$ , expressed as follows:

$$S = \{0, 1, \dots, k - 1\}. \quad (13)$$

In this case, we cannot use the boolean cube, but we can map the neighbourhood values on specific points of the same cube. The length of the string that represents the rule table will be equal to  $k^{(2r+1)}$ . On the cube it will be possible to detect a corresponding number of points. In Figure 6 two examples of the application of this generalization, with  $k = 3$  (on the left) and  $k = 4$  (on the right) are showed. The vector  $u_i$  with  $i = 0, 1, 2, \dots, (k^{(2r+1)} - 1)$

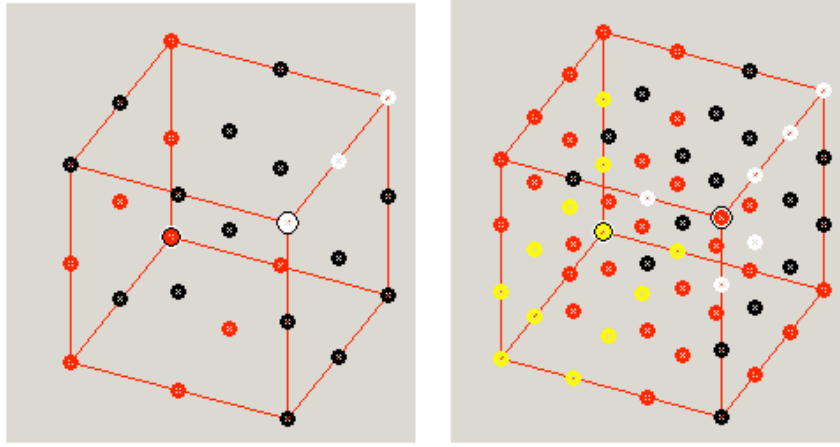


Fig. 6. In this Figure, on the left it is shown a representation of the neighbourhood values for  $k = 3$  (the string is: 011112122011112121011112222). On the right a representation of the neighbourhood values for  $k = 4$  ( the rule is: 2011012212232233001211221223223301121122122322320112112222232333)

has the following components:

$$u = (p, q, r) \quad (14)$$

where  $p \in S, q \in S, r \in S$ . So that

$$u_0 = (0, 0, 0), u_1 = (0, 0, 1), u_2 = (0, 0, 2), \dots, u_{k^{(2r+1)}-1} = (k-1, k-1, k-1)$$

$$w(\sigma) = z_2 + c_2 [|z_1 + c_1 |z_0 + \sigma||] \quad (15)$$

Also in this case the discriminating function depends on 8 parameters  $z_2, z_1, z_0, b_1, b_2, b_3, c_2$  and  $c_1$ , where  $c_2$  and  $c_1$  are two integers which can assume the values  $\pm 1$ . Furthermore, it is possible to introduce a dynamical system like (9), with the same driving function (10), whose attractors will be fixed points  $Q_i$  and they are as many as the neighbours present in the local rule. They will be equal to  $k^{(2r+1)}$ . In the same way,

10

- Given a neighbourhood  $u(p, q, r)$ , the state  $x(t)$  of each dynamical system, with initial conditions  $x(0) = 0$ , converges monotonically toward an attractor  $Q_{qp}$ .

In order to calculate the output, we define

$$\bar{y}_{pqr} = (k - 1) \frac{Q_{qp} - \min}{\max - \min} \quad (16)$$

where max and min are the maximum and minimum values of the attractors. The output  $y_{pqr}$  which corresponds to the neighbourhood  $u(p, q, r)$  will be then defined as the integer  $y_{pqr}$  nearer to  $\bar{y}_{pqr}$ . In order to calculate the output, we define

$$y_{pqr} = \text{Cint}(\bar{y}_{pqr}) \quad (17)$$

In Figure 7, different rules obtained by the same neuron, with different values of  $k$ , are visualized, considering the universal neuron that generates the rule 110 with  $k=2$ . As it is possible to see, the rules present some emergent structures such as regular domains and gliders. The behaviour of the obtained rules presents many differences as well. This approach is very useful

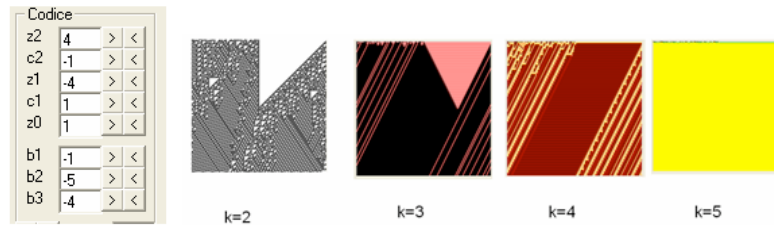


Fig. 7. In this image, different rules obtained by the same neuron, modifying the value of  $k$  are visualized.

in searching for multistate CA rules. Furthermore, the method of searching CA complex rules by using dynamical systems allows the generation of chips endowed with these mechanisms and the use of parallel analogical systems with many advantages with respect to traditional methods.

#### 4. Genetical approach

The method of the universal neuron allows for a higher level of searching for complex rules and the analysis of their corresponding genetic traits. For example, it was possible to investigate the relationship between the

change of the universal neuron parameters and the change in the complex rules patterns. Figure 8 shows how CA patterns change, varying parameter  $b_3$  (that is to say changing the orientation of the vector on which the points of the cube are projected). As it is possible to see, the rules present behaviours which go from complex to ordered organizations. In the follow-

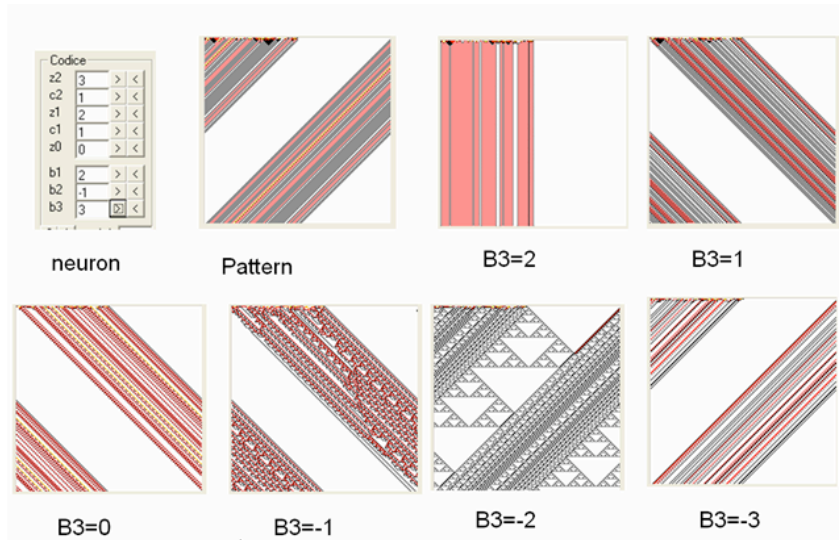


Fig. 8. In this image a collection of patterns that change in relation to the variations of parameter  $b_3$ .

ing Table 2, there are the related collection of rules presented in Figure 8 and on the right the values of the universal neuron, with the variation of parameter  $b_3$ . From our experiments, the variations of the parameters  $b_1$

Table 2. In this table the related collection of rules presented in Figure 8 and on the right the values of the universal neuron, with the variation of parameter  $b_3$  are reported.

2211321132213221211022102211321121002110211022101101110021002110	>	3	1	2	1	0	2	-1	3
2211222132213222211022102211222111002100211022101001100011002100	>	3	1	2	1	0	2	-1	2
2211222132223322111021112211222110001100111021110011000110001100	>	3	1	2	1	0	2	-1	1
222222222223333000011112222222200000000000011112222111100000000	>	3	1	2	1	0	2	-1	0
0012012212222231000000100120122210210010000001322222122102100	>	3	1	2	1	0	2	-1	-1
1001101100120112210011011001101122102110210011013221321122102110	>	3	1	2	1	0	2	-1	-2
2101100110011012210021002101100122102210210021003221321022102210	>	3	1	2	1	0	2	-1	-3

and  $b_2$ , even with negative values, are very interesting for the presence of complex structures in the patterns. The variations of the parameter  $z$  is instead less relevant. This new approach allows to investigate the nature of the CA rule space, in relationship with the Universal Neuron parameter space, establishing relationships between discrete and continuous dynamical systems and analysing the nature of local and global rules in affecting the behaviour of these systems.

### References

1. von Neumann, J. (1966). *Theory of Self-Reproducing Automata*. University of Illinois Press, Illinois. Edited and completed by A. W. Burks.
2. Langton, C.G. (1984). Self-Reproduction in Cellular Automata. *Physica D*, 10, pp. 135-144.
3. Langton, C.G. (1986). Studying artificial life with cellular automata. *Physica D*, 22, pp. 120-149.
4. Langton, C.G. (1988). Computation at the edge of chaos, *Physica D*, 42, pp. 12-37.
5. Wolfram, S. (2002). *A New Kind of Science*. LLC, Canada.
6. Aizawa Y., Nishikawa I., Kaneko K. (1990) Soliton Turbulence in one-dimensional Cellular Automata, *Physica D*, 45, pp. 307-327
7. Linan G., Rodriguez-Vazquez A., Espejo S. and Dominguez-Castro R. (2002) "ACE16K: A 128 x 128 focal plane analog processor with digital I/O," *Proc. 7th IEEE Int. Workshop on Cellular Neural Networks and their Applications*, ed. Tetzlaff, R. (World Scientific, Singapore), pp. 132-139.
8. Chua L.O. (1998). *CNN: A paradigm for complexity*, World Scientific.
9. Wolfram S. (1984). Universality and Complexity in Cellular Automata, *Physica D*, 10, pp. 1-35.
10. Bilotta E., Lafusa A. and Pantano P. (2003). Searching for complex CA rules with GA's. *Complexity*, 8 (3), pp. 56-67
11. Bilotta E., Lafusa A. and Pantano P. (2003). Life-like self-reproducers. *Complexity*, 9 (1), pp. 38-55.
12. Bilotta E. and Pantano P. (2005). Emergent patterning phenomena in 2D Cellular automata. *Artif Life*, 11 (3), pp. 339-362.
13. Bilotta E. and Pantano P. (2006). Structural and Functional Growth in Self-Reproducing Cellular Automata. *Complexity*, 11 (6), pp. 12-29.
14. Chua L.O., Yoon S. and Dogaru R. (2002). A nonlinear dynamics perspective of Wolfram's new kind of science. Part I: Threshold of complexity. *Int. J. Bifurcation and Chaos*, 12, pp. 2655-2766.
15. Chua L.O., Sbitnev V.I. and Yoon S. (2004). "A nonlinear dynamics perspective of Wolfram's new kind of science. Part III: Predicting the unpredictable," *Int. J. Bifurcation and Chaos*, 14, pp. 3689-3820.