Is Self-replication an Embedded Characteristic of Artificial/Living Matter?

Eleonora Bilotta, Antonio Lafusa and Pietro Pantano

Centro Interdipartimentale della Comunicazione, Universitá della Calabria
Cubo 17, 87036 Arcavacata di Rende (CS) Italy
bilotta@unical.it, alafusa@tiscalinet.it, piepa@unical.it

Abstract

This paper introduces a method through which, using genetic algorithms on two dimensional cellular automata, we obtain emergent phenomena of self-replication. Three indices of complexity, based on input entropy have been developed and used as fitness functions in the evolutionary experiments. The genetic algorithm, realized by a special design of the genome, is efficient and the research in the CA rules space has given appreciable results, both for the quantity and for the quality of the emergent phenomena. We found that each of these indices is strictly connected to the complexity of the rules and to the self-reproducers behavior contained in them. We noticed that self-reproduction is a widespread process also in artificial life simulations. Almost all the evolved rules manifest self-reproducers, as if this process were an embedded characteristic of artificial/living matter. The self-reproducers, different in shape, function and behavior, reveal an algorithmic logic in self-replication, which follows different but synchronized rhythms, evidencing variation, increasing structural complexity and some of them general constructive capacity.

Introduction

One of the most important properties of living beings is reproduction that is the capacity of organisms to create copies of themselves, through highly specialized processes of duplication or multiplication. In the biological domain, different reproduction mechanisms exist. Reproduction allows the multiplication of the number of individuals of one species, starting from examples of the same species. The reproduction of a living organism, however small or simple, involves a programme able to build a new entity, which itself contains the same programme and transmits it in its turn (Maynard Smith and Szathmáry 1995). The problem of reproduction of life-like forms into other digital media is one of the main goals and challenges of contemporary research (Bedau et al. 2000). Almost all the applications that deal with the simulation and synthesis of living systems are linked to the seminal work of John von Neumann (1966), who proposed a machine able to reproduce itself. Since then, systems in which space, time and states are discrete, called Cellular Automata (CA), have been mainly used to study self-reproduction in the realm of artificial life. The research of self-reproducing structures can be divided into four categories. The first deals with the implementation of universal constructors, based on the approach of von Neumann's self-reproducing automaton, traceable to a series of studies made in the 50's and 60's (von Neumann 1966; Codd 1968; Vitányi, 1973). Subsequently, research into a minimum system capable of non-trivial reproduction took place, thanks to a series of studies started by Langton (1984), which stimulated other similar works (Byl 1989; Sipper 1994 and 1998; Morita and Imai 1996). During the 90's, many studies provided other computing capabilities of the self-reproducers, (Perrier, Sipper and Zahnd 1996; Chou and Reggia 1998). From the 90's to today, many researchers have been investigating the self-reproducers' emergence and evolution (Lohn and Reggia 1995; Chou and Reggia 1997; Sayama 1998).

In von Neumann's endeavor, the basic idea of such works is that a self-reproduction machine could have the characteristic of computational universality, that is the ability to operate as a universal Turing machine and the ability to construct any kind of configuration in the cellular space, starting from a given description. In particular, self reproduction is a special case of universal construction. (Langton 1984). But von Neumann's analysis also shows a series of different problems involved in the self-reproduction process: the method through which self-reproduction is obtained, the reproducer's strength to mutation, the possibility of variation, heredity, the notion of complication of the system, the increasing in complexity of the system, and the essence of the concept of self-reproduction, related to the general constructive capacity of the automaton. Only recent studies have focused on understanding the complexity of the process of self-reproduction in this direction (McMullin 2000). Moreover, almost all self-reproduction systems based on CA are hand-designed, not changeable, subject to environmental perturbations and only some of them consider evolution (for example, Sayama 2000).

This paper deals with the development of a new method for obtaining self-reproduction in the artificial domain, trying to individuate some of the basic mechanisms of self-reproducer systems. We began by considering a set of fundamental questions: Is it possible that the logic of computing that von Neumann looked for, is the essence of life and that these very informational and computing capacities originate life also in the artificial domain? Is self-reproduction an innate characteristic of living matter, which can also be re-proposed in the process of artificial synthesis? To what extent does synthetic life have to be complex to show itself? A second group of problems concerns such questions as: Which are the mechanisms that lead to self-reproduction? Is it possible to identify some of the basic mechanisms of self-reproducer systems? Is it possible to think that gliders, in evolving, lead to self-reproduction? Is it possible to structure a fitness function that in some way evolves self-reproducers? Is there a relationship between complexity or levels of complexity and self-reproduction?

We used Genetic Algorithms to evolve two-dimensional CA in order to obtain self-reproducing systems. We had to limit our attention to a subset of the CA rules space, which we defined as k-totalistic rules. We assumed that self-reproducers can be found inside the complex rules of class IV (Wolfram, 1984). Within such rules, we found different types of self-reproducers. Moreover, we developed three fitness functions that roughly correspond to the three complexity indices functions, which we have used to perform different evolutionary runs. In our opinion, each of these indices is strictly connected to the complexity of the rules and to the self-reproducer's behavior contained in them. We believe that different types of self-reproducers exist that show a great variety of shapes, functions and behavior.

This work has a second paragraph that deals with a CA and GA overview. In particular, the design of the k-totalistic rule, used for representing the genome and to start the evolutionary process for two-dimensional CA, is described. The indices of complexity are given in the third paragraph. Three experiments, each one with a different fitness function and thus with a different index of complexity, are reported in the fourth paragraph, together with the results and some examples of the self-reproducers obtained. In the fifth paragraph, some self-reproducers are described, according to the shape, function and behavior which characterize them. The conclusions try to extrapolate a coherent view of these complex problems.

Cellular Automata and Genetic Algorithms overview

The environment considered is a two-dimensional CA. A CA can be thought as a tuple:

$$A = (d, S, N, f) \tag{1}$$

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

$$f : S^n \longrightarrow S. \tag{2}$$

In our case $d = 2$, and the neighborhood identifies the cells with a local interaction radius r. So (2) to the $n = (2r+1)^2$ elements of S associates another element of S, i.e.

$$\begin{pmatrix} \cdots & \cdots & \cdots \\ \cdots & s_{ij} & \cdots \\ \cdots & \cdots & \cdots \end{pmatrix} \longrightarrow s_{ij}, \tag{3}$$

with $s_{ij} \in S \ \forall i, j$.

An exhaustive rule considers all $k^{(2r+1)^2}$ possible cases. In a previous work (Bilotta, Lafusa and Pantano, submitted), we used a GA based on input-entropy to find the complex rules of class IV (Wolfram, 1984) for one-dimensional CA. With the increase of k and r, the exhaustive rule becomes non-treatable and it is necessary to use a new simple form to be used as the genome, to start the evolutionary process. Here we define a rule (called hereafter the k-totalistic rule), considering rules that don't distinguish the position of neighbors in the surrounding area, but just consider how many cells are in a given state. Let $h_s(t)$ be the number of cells of the neighborhood that are in the state s at time t. We denote with V the set of all possible configurations of the neighborhood, whose elements can be represented by a numerical string $(h_0 h_1 ... h_{k-1})$. h_i values are not arbitrary, but they must satisfy the following constraints:

$$\begin{array}{l} h_0 + h_1 + ... + h_{k-1} = (2r+1)^2 \\ \text{and} \\ h_i \geq 0, \ for \ i = 0, 1, ..., k-1. \end{array} \tag{4}$$

By definition a totalistic rule T is an application that associates with any configuration $v \in V$ an S element:

$$T : V \longrightarrow S \tag{5}$$

In particular, the application (5) can be represented explicitly as a table that associates with the first $k-1$ totals (for the constraints (4), the last of which depends on the others) a number included between 0 and $k-1$. For example, according to (4), the string (240), for a CA $k = 4$ and $r = 1$, indicates that 2 elements of the neighborhood, composed of 9 cells, are in the state 0; 4 elements are in the state 1; 0 in the state 2, and 3 in the state 3. To the string (240) will correspond one element $q \in S$. In this example, 220 local rules of this kind will exist. In this way, the constraints (4) allow a substantial lowering of the number of possible configurations. To estimate this value, we consider the problem as being the same as that of placing $(2r+1)^2$ undistinguishable objects in k containers. From the elements of combinatorial analysis, such a number is given by:

$$N_T = \begin{pmatrix} k+(2r+1)^2-1 \\ (2r+1)^2 \end{pmatrix} = \frac{(k+(2r+1)^2-1)!}{(2r+1)^2!(k-1)!} \quad (6)$$

The table of rules, that allows the association of an element of S with any configuration of the neighborhood, can be represented as a numerical sequence, of N_T length:

$$l = (s_1, s_2, ..., s_{N_T}) \quad (7)$$

with $s_i \in S$. To make a configuration $(h_0 h_1 ... h_{k-1}) \in V$ correspond to an s_i value of the string (7) and vice versa, we fix the convention that s_i corresponds to the i^{th} configuration in lexicographical order. To deal with the rule algorithmically, it is convenient to characterize it as a decisional tree (Knuth 1997), whose intermediate knots are represented by the values of h_j, whose arcs stand for the alternatives and whose leaves are the s_i of the string (7). In this case, a configuration $(h_0 h_1 ... h_{k-1}) \in V$ is a path along the oriented tree. For $k = 2$, the k-totalistic rules become the well-known totalistic rules for binary CA. Using the k-totalistic rule, given by (5), the rule becomes treatable even for higher values of k and r. With the k-totalistic rules used as a genome, we adapted a relatively standard method for designing the GA (Holland 1975; Mitchell 1996). The important design parameters were: initial generation of rules casually chosen, crossover and mutation operators, generational replacement with elitism, updating of the fitness scores. We used the variance of the input-entropy as fitness function. Such a function is an index of the degree of order-chaos that we applied to the k-totalistic rules, following the method devised by Wuensche (Wuensche 1999) for exhaustive rules. For a step of simulation, the input-entropy ε^t is defined as:

$$\varepsilon^t = -\sum_{i=1}^{N_T} \left(\frac{Q_i^t}{n} \log \left(\frac{Q_i^t}{n} \right) \right) \quad (8)$$

where n is the number of cells in the grid of the CA, and Q_i is the frequency of consultation of the i^{th}local rule. From our experiments, we verified that the input-entropy on k-totalistic rules works in a similar way as it does for exhaustive rules in one-dimensional CA. In ordered systems, it goes immediately to a minimum constant value; in chaotic systems, it stays constantly on high values, with very little variations. On the contrary, in complex systems, (8) varies remarkably over time. The initial population of rules in the GA are generated using the string (7). Crossover and mutation operators generate the rules of the next generations, between the rules of the elite and successive mutations. The GA parameters will be given later.

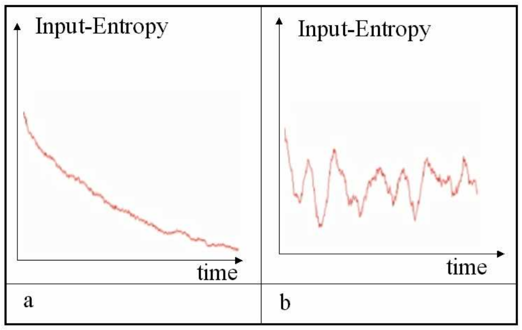

Figure 1: Input-entropy curves. 1a shows the tendency of the curve after 350 steps of simulation, for an ordered rule; 1b, another tendency of the curve after 350 steps of simulation for a rule that presents oscillatory phases.

Indices of complexity which indicate the presence of self-reproducers

We started up many evolutionary runs using as a fitness function the variance of input-entropy, given in (8. Results were, at first, not very satisfactory. In fact, designing a fitness function to evaluate self-replication is a very difficult task, since this is a highly dynamic complex process. Is it possible to evaluate complexity by means of some indices, which indicate the presence of self-reproducers? In evaluating the fitness of each rule, we observed two classes of curves for the rules, which scored high fitness values:
a) an input-entropy tendency, slowly decreasing, from an initial chaotic phase to an ordered phase (see figure 1a). These systems show a transitory that extinguishes after a very long time.
b) an oscillatory tendency, which continuously alternates different phases (figure 1b).

Furthermore, we observed that for systems in which self-reproducing structures emerge, the tendency of the input-entropy is of the type 1b. To select just the systems of the type given in figure 1b, we used the strategy of directly detecting an oscillatory tendency, defining a second index of complexity to be used as a fitness function. For the values of ε^t we calculate:
I) the absolute minimum $\varepsilon_{\min}$:

$$\varepsilon_{\min} = \min\ \varepsilon^t$$

II) the absolute maximum $\varepsilon_{\max}$, calculated starting from the minimum point $t_{\min}$:

$$\varepsilon_{\max} = \max\ \varepsilon^t;\ t > t_{\min}$$

III) the absolute minimum $\varepsilon'_{\min}$, calculated starting from $t_{\max}$:

$$\varepsilon'_{\min} = \min\ \varepsilon^t;\ t > t_{\max}$$

We define the index I_1 as the sum of the width of the two peaks between the minimum and the maximum of

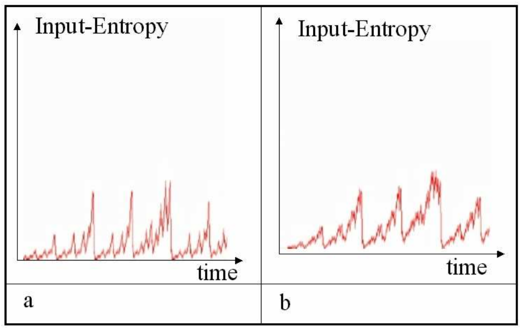

Figure 2: Different input-entropy curves. 2a shows the tendency of the input-entropy curve for a rule with high fitness after 350 steps of simulation; 2b, the tendency of the input-entropy curve for a rule with high fitness after 350 steps.

the curves, from $t_{\min}$ to $t_{\max}$ and from $t_{\max}$ to $t'_{\min}$

$$I_1 = (\varepsilon_{\max} - \varepsilon_{\min}) + (\varepsilon_{\max} - \varepsilon'_{\min}) \quad (9)$$

The higher the input-entropy peaks are, the greater I_1 will be. We moreover observed that some rules, which present self-reproducers, obtain medium fitness values. More precisely, these systems are characterized by an oscillatory tendency of the input-entropy, but with a moderate width. To modify the I_1 index, so that the oscillations around values lower than ε^t would be detected, we divided the I_1 by the values of the ε^t mean, obtaining a measure independent of the scale of the peaks. So, we define I_2 as:

$$I_2 = I_1/M \quad (10)$$

where M is the mean of ε^t values, calculated during a simulation. Figure 2 shows the typical tendency of the input-entropy, for rules with a high value of the index I_2. In the rule which produces the curve in Figure 2a, one or two self-reproducers are present. In the rule which produces the curve in Figure 2b, we found many self-reproducers.

Experiments

Three experiments, each with a different index of complexity, as above mentioned, were performed. Given a k and r, the genetic parameters that might be chosen are listed in Table 1. Crossover works on a point of the genome, randomly chosen.

In calculating the fitness function, in order to eliminate the high frequencies of entropies, we do not estimate the entropy of every time step, but we take a mean of the entropies on a time window of w steps, on which we evaluate the variance. Furthermore, we determine the fitness, starting from F time steps, to allow the CA to adhere to its typical behavior (Wuensche 1999), allowing the system to exit from the initial chaotic configuration. To avoid rules with good fitness in one generation being discarded in the next, we update the fitness of a rule only if that of the next generation is higher; otherwise, we maintain the previous value.

P	number of rules for each generation
G	Number of generations
E	Elite
M	Mutation
w	Time window
F	Initial steps
n	Number of CA sites
T	Steps of the simulation

Table 1: Parameters of the genetic algorithm.

We have also used the lambda parameter, as devised by Langton (Langton 1990), in order to ascertain if there is a relation between the lambda values and the emergence of self-reproducers in the evolved CA.

Experiment 1. Complexity index: input entropy

The parameters we chose for the CA are listed in Table 2. The experimental results for such an evolutionary run are given in Figure 3. In Figure 3a, it is possible to note the curves of the best, the mean and the elite fitness for each generation. The best fitness is low for nearly 10 generations, and then goes up suddenly.

States number	$k = 4$
Neighborhood radius	$r = 1$
Generations	$G = 20$
Populations	$P = 40$
Ignored initial steps	$F = 50$
Steps of simulation	$T = 350$
Elite (%)	$E = 50.0$
Mutation (%)	$M = 2.0$
CA sites	size $x = 40$; size $y = 40$ ($n = 160$)

Table 2: Parameters for experiment 1.

Figure 3b shows how the rules of the last generation are distributed in the lambda parameter.

In order to ascertain the differences among the evolved rule of the last generation, we determined the Hamming distance, calculated with regard to the rule which obtained the best fitness in the last generation, for this evolutionary run. This distance evaluates the differences in the genome of each rule. The maximum value which resulted is 177, realized by the 34^{th} rule. The minimum value obtained is 4, achieved by the 2^{nd} and 17^{th} rules. This evolutionary run produced a number of complex rules, many of which have self-reproducers. The rule

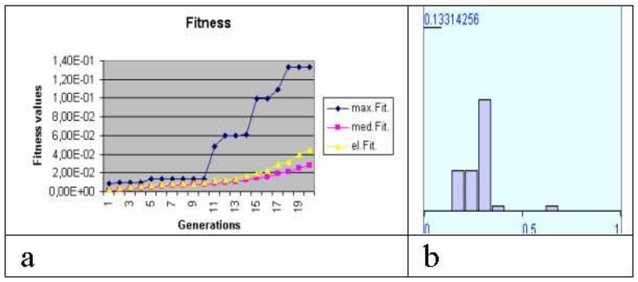

Figure 3: This figure illustrates, in a, the curves of the best, the mean and the elite fitness for each generation. b shows how the rules of the last generation are distributed in the lambda parameter in experiment 1.

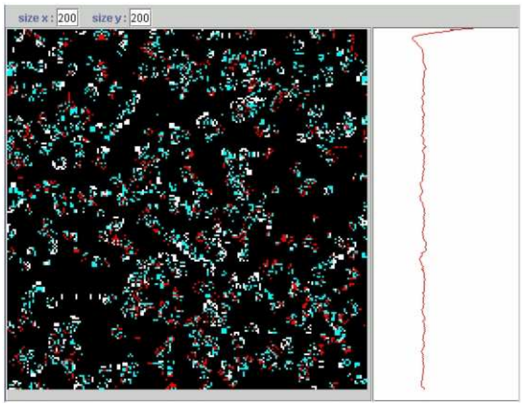

Figure 4: A pattern generated with the best fitness rule of the last generation after 200 time steps, starting from an initial casual configuration. On the right is the curve of the input-entropy. Starting from an initial very high value, it rapidly goes down, to stop, successively, on a stationary value regarding which it will have weak oscillations. Many self-reproducers can be observed.

with the best fitness has many self-reproducers (see Figure 4) and clearly belongs to Wolfram's fourth class. For example, the structure given in Figure 5,

$$\begin{pmatrix} 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 & 2 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 \end{pmatrix}$$

evolves in two copies of itself, after two time steps. It evolves further into more complex patterns, which show scaling invariance. Concisely, we can say that this run has produced a lot of complex CA rules, which are clustered in the lambda parameter with values which go from 0.15 to 0.43.

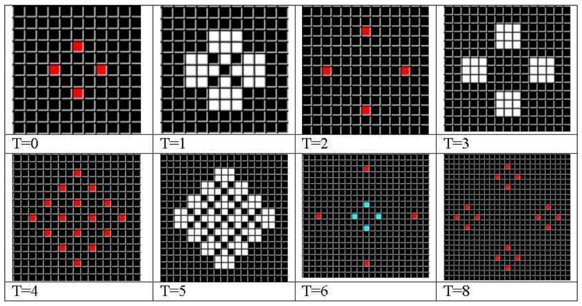

Figure 5: Process of self-reproduction which manifests scaling invariance.

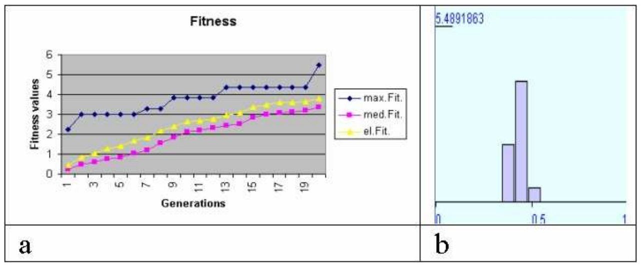

Figure 6: This figure illustrates, in a, the curves of the best, the mean and the elite fitness for each generation. b shows how the rules of the last generation are distributed in the lambda parameter for experiment 2.

Experiment 2. Complexity index: I_1

The parameters used for the CA are the same as those of experiment 1, varying only the fitness function. The experimental results for this evolutionary run are given in Figure 6. In this experiment, the curves for the maximum, mean and elite fitness for each generation have grown remarkably, with a quantitative increase in their values. Figure 6b shows how the rules of the last generation are distributed in the lambda parameter. The Hamming distance presents a greater dispersion with regard to the previous run, with a maximum value of 122 for the 20th rule, and a minimum value of 31 for the 18th rule. In this run, the rules of the last generation are clustered in the values of the lambda parameter nearer to 0.5, going from 0.36 to 0.57. Using I_1 as a fitness function, it is possible to see that the rules with higher fitness have an input-entropy curve tendency of the type highlighted in figure 1b. Moreover, the percentage of systems in which there are self-reproducing structures increases remarkably with regard to those obtained using the variance of input-entropy. Those CA with self-reproducing structures constitute the greater part of the rules of the last generation. The best fitness rule of the last generation presents many gliders, as shown in Figure 7. It is worth noting that the glider in Figure 7a becomes a complex self-reproducer if we use this structure as input data in

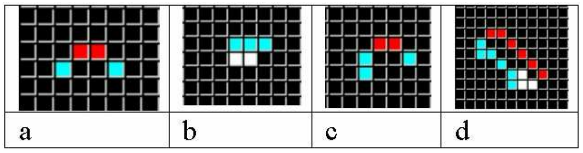

Figure 7: Examples of gliders from the best fitness rule of the last generation of experiment 2

other rules of this evolutive run. This could mean that evolution links together glider, reproduction and complexity by means of a specialized function which evolved from a simple carrying on behavior to self-reproducing behavior or viceversa, and that reproduction is a way to carry on information in the environment.

Experiment 3. Complexity index: I_2

For this evolutionary run too, we varied only the fitness function. The experimental results for such an evolutionary run (8) show how the curves for the maximum,

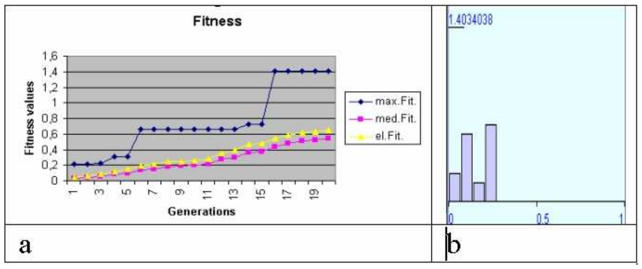

Figure 8: This figure illustrates, in a, the curves of the best, the mean and the elite fitness for each generation. In b, the rules of the last generation are distributed in the lambda parameter for experiment 3.

mean and elite fitness for each generation have grown. Figure 8b shows how the rules of the last generation are distributed in the lambda parameter. The Hamming distance presents a different organization with regard to the previous run, with a maximum value of 67 for the 16th rule, and a minimum value of 4 for the 12th rule. In this run, the distribution of the last generation rules in the lambda parameter highlights a different clustering, going from 0.05 to 0.30. Even having a fitness value inferior both to the first and to the second experiment, the rules of the last generation of this experiment present a large number of self-reproducers. In some rules, particularly those in which I_2 is high, larger structures emerge after a certain time, formed by self-reproducers, which, in their turn, are self-reproducing structures. Also for this evolutionary run, the rules of the last generation present a lot of replicators, from which we choose those presented in Figure 9.

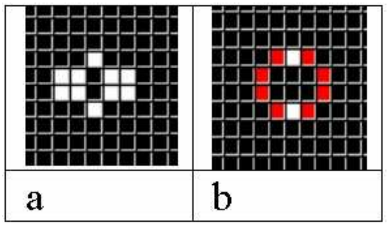

Figure 9: Two examples of self-reproducing structures found in the rules of the last generation of experiment 3

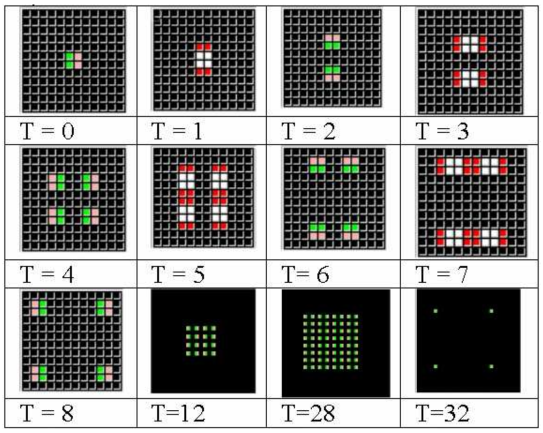

Figure 10: Evolution of a simple self-reproduction structure in different time steps.

Self-reproducers shapes, functions and behavior

If after a short transient period, we start simulation of an evolved rule, self-replicating patterns emerge. The behavior of the self-reproducer takes place at micro and macro-levels of the simulation time steps. The typical outline of the micro-level deals with self-replication. Self-reproducers grow over time, showing scaling invariance, change their position in the CA space, rotate and change their orientation, and change their spatial coordinates, undergoing annihilation when replication is no longer possible. But they never disappear, instead beginning a new loop of replication. We call this kind of behavior simple self-replication, especially when it happens in the cellular automata space in isolation. Many of the structures we found exhibit this kind of simple self-replication. In Figure 10, we illustrate an example of how, starting from self-reproducer data, this process occurs. At time $T = 0$ (initial state) there is a single structure formed by states (44, 66). At step $T = 1$ the structure is transformed and a new structure, formed by the states (22, 00, 00, 22), appears, which will follow its own path with its own behavior law. At step $T = 2$, the initial self-reproducer repeats itself, generating two copies in different positions. At step $T = 3$,

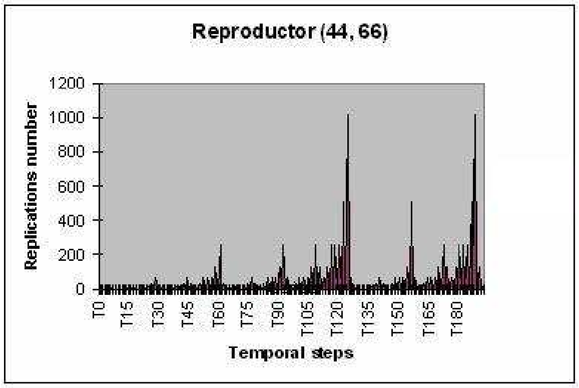

Figure 11: Graphic of the self-reproduction process of the structure formed by the states (44, 66)

two copies of the configuration of step $T = 1$ are generated, in different positions. At step $T = 4$, four copies of the self-reproducer (44,66) are obtained. At step $T = 5$, four copies of the self-reproducer (22, 00, 00, 22) are obtained. And so on. Let us now consider the self-reproducer (44, 66). If we make such a reproducer evolve for a relevant number of steps (190), we can see that it describes in time the curve shown in Figure 11. As the graphic shows, the increasing of the self-reproducer follows a constant and repeated tendency over time, with changes in the replication scale. Such a replication occurs at different time scales (different time steps of the simulation) and with different size scales (number of different replications). This curve led us to understand that the process of replication follows a specific law, since it is possible to note how some recurring patterns behave as if they were musical rhythms which repeat themselves as refrains, along a growing and changing scale. To explain this analogy, we can say that the curve seems to be a musical canon which recurs on different scales, with groups of 8, slightly different notes, as are listed in Table 3. The curve presented in Figure 11 subsumes a

2	4	4	4	8	16	8	4
8	16	16	16	32	64	16	4
8	16	16	16	32	64	32	16
32	64	64	64	128	256	32	4
8	16	16	16	32	64	32	16
32	64	64	64	128	256	64	16
32	64	64	64	128	256	128	64
128	256	256	256	512	1024	64	4

Table 3: Numerical representation of the self- reproduction process.

specific reproduction algorithm, that is the number of copies which the structure is able to reproduce, at any time step, for a given time. Such an algorithm could

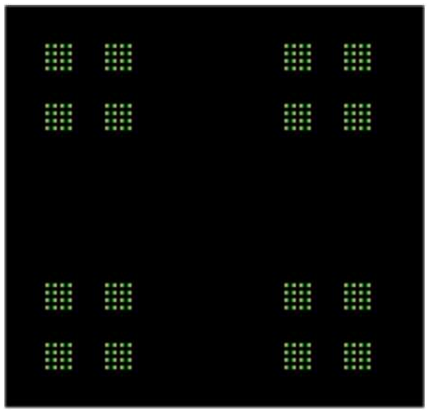

Figure 12: Patterns which represent scaling invariance in the reproduction process of the structure (44, 66).

be the logical part of reproduction as von Neumann and Langton meant it. The self-reproducers move, occupying ever-increasing portions of space to allow other copies of themselves to appear. Such a portion of space seems to be proportional to the number of copies that appear in a given time. The reproducer also presents two basic movements in the CA space, one in the cardinal directions N-S, E-W, and the other in a rotatory sense.

Moreover, the copying progression presents the interesting characteristic of scaling invariance, as can be observed in Figure 12. Each new configuration, emerging by replication, maintains the shape of the reproducer, with a different size. These geometric figures are similar, since they have the same shape, or new emerging shape, given by the sum of the parts which compose the new structure. Together with the reproducer (44, 66) another self-reproducer exists, (with the following data 22, 00, 00, 22) which seems to be connected to the reproducer by which it is in some way manifested. The datum, if given on its own as input, is able to self-reproduce, manifesting also the reproducer (44, 66). It looks as if the two reproducers are in some way related, even though they have a different structure, both sharing a reproductive processes in which they are involved. In the graphics of Figures 13 and 14 respectively, the time development of the reproducer (22, 00, 00, 22) and the process of the two reproducers represented together, are shown. From the graphic in Figure 13, it can be seen how the reproduction rhythm for (22, 00, 00, 22) is slower, it is therefore more scattered in the time flow and less dense in the number of reproductions (we have excluded, for the time being, in order to better understand the phenomenon, the steps of simulation in which the copies interact through some functional junctions amongst themselves, that surely highlight other processes usually explained as overpopulation, from which other phases of annihilation follow).

Instead, the copies reproduced by the structure (44, 66) never come into contact with one another. The same categories as have been noticed for the reproducing structure (44, 66) are valid for (22, 00, 00, 22). There are changes in scale and invariance of shape while the progression follows a rhythm of evolution with increasing-decreasing phases. The rhythm of this reproducer, formed only by two beats or notes, is the following: 1-2; 4-8; 4-8; 4-8; 16-32; 4-8; 16-32; 16-32; 64-128; 4-8; 16-32; 16-32, and so on. From the graphic in Figure 14,

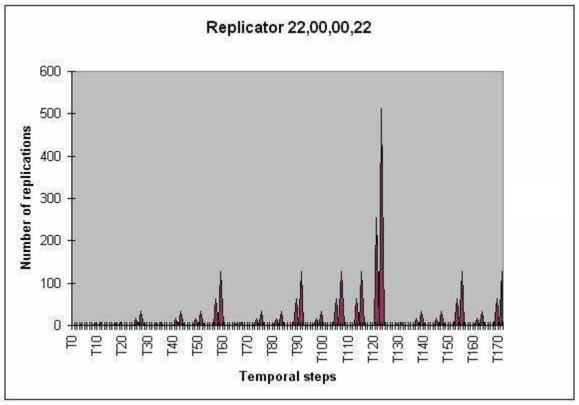

Figure 13: Graphic of the self-reproduction process of the structure formed by the states (22, 00, 00, 22).

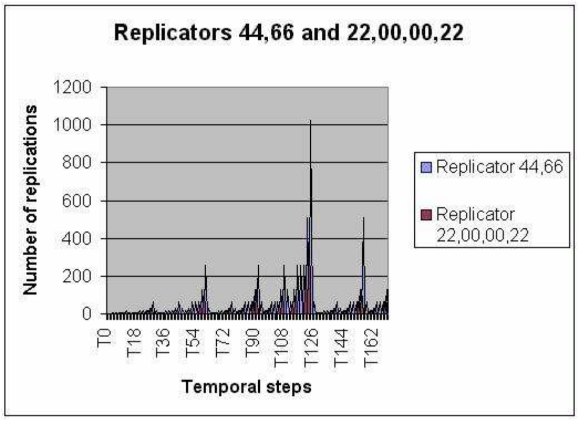

Figure 14: Process of reproduction of the two reproducers considered to be unitary.

it can be seen how the reproducer (44, 66) is quantitatively preponderant and seems to be in the foreground with regard to the other reproducer, that seems to be in the background.

The most interesting phenomenon, (which could be von Neumann 's idea about the general constructive capacity of the automaton) for the reproducer (44, 66) is the possibility of reproducing structures realized ad hoc. Any macro-structure, generated in this way, and formed by the arbitrary combination of simple structures made by the reproducer (44, 66), is, in its turn, a self-reproducing structure (see Figure 15). This property was noticed for different self-reproducers. In order

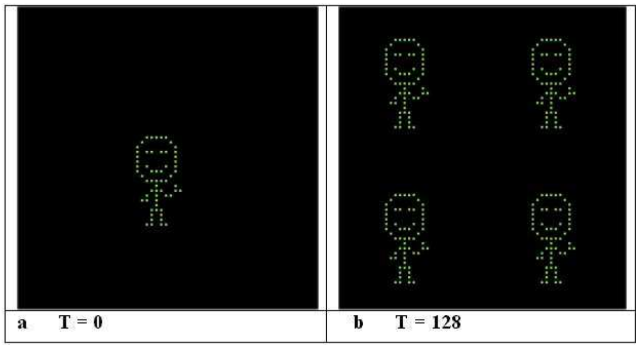

Figure 15: Self-reproduction of an arbitrary configuration (14a). After 128 steps, four copies of the homunculus are obtained (14b).

that any component of whatever structure we want to replicate can interact correctly, it is necessary that each element have the same orientation, or that the elements are turned around by 180° with respect to each other and that each element is at a distance of 2 cells, or at a distance of 2 cells plus a multiple of four. The mechanism of reproduction of these components put together is similar to a simple reproducer's behavior.

The structure (22) is one of the many replicators contained in the best fitness rule of the last generation, evolved by the experiment 3. (22) is the simplest one. As can be seen in Figure 16, the structure replicates itself at $T = 4$ and $T = 8$, and produces in $T = 1$, $T = 2$ and $T = 3$ other structures which, in turn, reproduce themselves in $T = 5$, $T = 6$ and $T = 7$. In the same rule, as shown in Figure 17, using as initial data (22, 22, 55, 55), it is possible to obtain a reproduction process which occurs in almost every reproducer which this rule contains, like a Sierpinski Triangle, with the typical geometrical features that this phenomenon manifests in one dimensional CA.

Other kinds of reproducer are different as regards structure and behavior. We realized different evolutionary runs for different values of k and r, and we found that self-replication does not depend on k and r values. In all evolved rules, self-reproducers are always present, and they seem to stand for the basic but widespread potential of artificial matter to reproduce and to evolve. In the following, we present a complex phenomenon of reproduction for a $k = 4$, $r = 2$ CA rule. The simulation begins with an initial casual state, and, after many time steps, the self-replicating structure shown in Figure 18 can be observed. The self-replicating structure creates four different directions of the replication process,

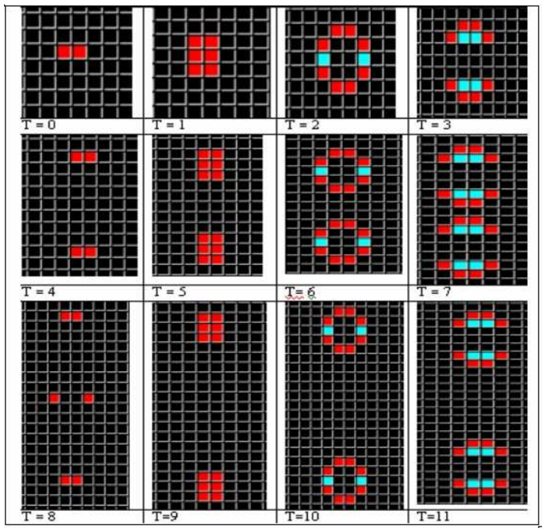

Figure 16: Self-reproduction with a simple behavior.

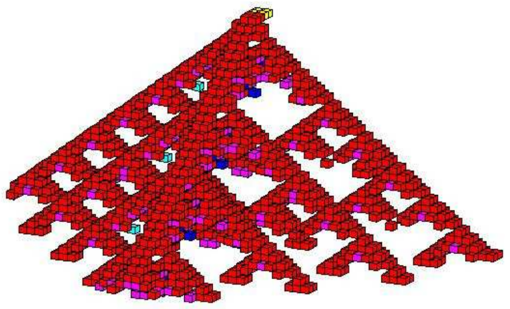

Figure 17: 3d spatial-temporal pattern from the data (22, 22, 55, 55) which has a Sierpinski Triangle configuration.

each generating many copies and giving life to a population of other entities, which in their turn begin to reproduce. Moreover, during the explosion of the population, many gliders are produced, so it seems that, globally, the macro-structure behaves as a glider gun.

Conclusions

Making use of two-dimensional Cellular Automata and Genetic Algorithms, it is possible to evolve complex rules that contain self-reproducers, which emerge spontaneously during the evolutionary process. The GA realized is efficient and research in the space of k-totalistic rules has given appreciable results, both for the quantity and for the quality of the phenomena which have emerged. The three types of fitness functions, related to the three indices of complexity that were developed, manifest types of reproducers which are different in shape, function and behavior. The self-reproducers

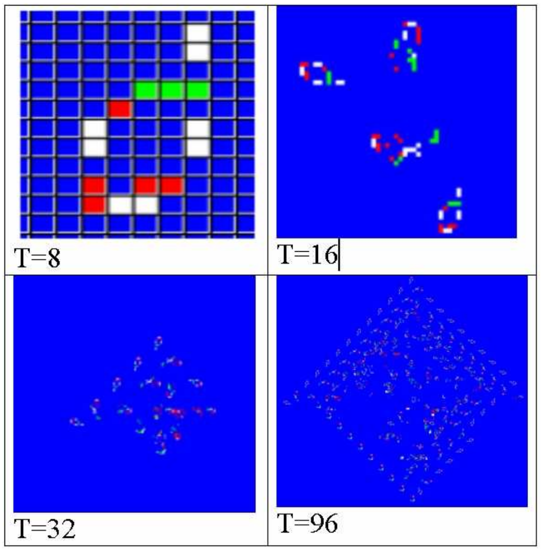

Figure 18: Complex phenomena of self-reproduction. The structure evolves creating a global self-constructing macro-structure. The last configuration seems similar to the initial structure, but, in the middle of the pattern, many other processes take place.

emerge from the primeval soup, are very stable, emerge again after collisions, and are able to self-inspect and reconstruct the information so that they can reproduce again. They are dominant over a long time period and exhibit changes in scale and invariance in shape. These kinds of structure reveal an algorithmic logic in reproduction, which follows different times, rhythms and phenomenology.

The duplicated information, increasing the total quantity of the system at a microscopic level, creates global changes at a macroscopic level, which modify the quality of the system: the shape and function of the self-reproducer organize themselves on different time, quantitative and qualitative scales. An increase of quantity will occur that will certainly be functional with regard to a change of quality or function, as occurs in the biological world where an additional production of DNA will be the basis for further programs that will concern it. Duplication in itself does not produce significant novelties, nor does it cause any increase in complexity, but it does provide the primary matter which allows the occurrence of such an event.

We have noticed that, as opposed to what is commonly believed, self-reproduction is a widespread process: almost all the evolved rules, even of the first generations, manifest self-reproducers, as if this process were an embedded characteristic of living matter, which can also be re-proposed in the artificial synthesis processes. The phenomenon of a structures replication can be seen as the creation of a redundancy of the system; that is the possibility of spreading some characteristics that they posses within more complicated systems, manifesting equally complicated behavior and so creating new organizations of living/artificial matter. We think that replication is the combination of many phenomena put together. Because we are operating in a context which is dynamic, and interactions with other data, information tends to multiply until a single structure has different dynamics and goes beyond what, at a first glance, it might seen to be. Replication phenomena may be similar, slightly different, different, and so on, in various degrees of quantitative and qualitative differences, but in some way harmonized or synchronized, and included in more general phenomena or on different scales. The mathematical language through which these phenomena are described is given by the specific algorithm of replication. These embedded characteristics can be considered as an evolved modality that living matter uses in order to reproduce itself. We have rediscovered it in artificial systems.

References

Bedau M. A., McCaskill J. S., Packard N. H., Rasmussen S., Adami C., Green D. G., Harvey I., Ikegami T., Kaneko K., and Ray T. S. 2000. Open problems in artificial life. Artificial Life 6: 363-376.

Bilotta E., and Pantano P. 2001. Observations on Complex Multi-state CAs. In J. Kelemen and P. Sosik eds. Advances in Artificial Life: Proceedings of the 6th European Conference on Artificial Life, 226-235. Springer-Verlag, Berlin.

Bilotta E., Lafusa A., and Pantano P. Searching for complex CA rules with GAs. submitted.

Byl J. 1989. Self-Reproduction in small cellular automata. Physica D 34: 295-299

Chou H. H., and Reggia J. A 1997. Emergence of self-replicating structures in a cellular automata space. Physica D 110(3-4): 252-276.

Chou H. H., and Reggia J. A 1998. Problem solving during artificial selection of self-replicating loops Physica D 115:293-312.

Codd E. F 1968. Cellular Automata. Academic Press, New York.

J. Holland. 1975 Adaptation In Natural and Artificial Systems. The University of Michigan Press, Ann Arbour.

Knuth D.E. 1997. The Art of Computer Programming, Vol. I. Addison-Wesley, New York.

Morita, K., and Imai K. 1996 Self-reproduction in a reversible cellular space. Theoretical Computer Science, 168:337-366.

Morita, K., and Imai K. 1997 Simple self-reproducing cellular automaton with shape-encoding mechanism. In Langton C.G., and Shimohara K., eds, Artificial Life V: Proceedings of the Fifth International Workshop on the Synthesis and Simulation of Living Systems, 450-457 Cambridge, MA,. MIT Press.

Langton C. G. 1984 Self-reproduction in cellular automata. Physica D 10:.135-144.

Langton C. G. 1990 Computation at the edge of chaos. Physica 42D 12-37.

Lohn J. D., and Reggia J. A. 1995 Discovery of self-replicating structures using a genetic algorithm, Proceedings of IEEE International Conference on Evolutionary Computation, 678-683.

Maynard Smith J., and Szathm*á*ry E. 1995 The Major Transitions in Evolution. W.H. Freeman, Oxford.

McMullin B. 2000 John von Neumann and the Evolutionary Growth of Complexity: Looking Backwards, Looking Forwards. In Bedau M. A.,McCaskill J. S., Packard N. H., and Rasmussen S. eds. Artificial Life VII: Proceedings of the Seventh International Conference, 467-476. MIT Press.

Mitchell M. 1996 An introduction to Genetic Algorithms MIT Press. Cambridge, Massachusetts. London, England.

Perrier J. Y., Sipper M., and Zahnd J 1996 Toward a viable, self-reproducing universal computer. Physica D, 97:335-352

Sayama H. 1998 Introduction of Structural Dissolution into Langton's Self-reproducing Loop. In Adami C., Belew R. K., Kitano H., and Taylor C.E. eds. Artificial Life VI: Proceedings of the Sixth International Conference on Artificial Life, 114-122 Los Angeles, California, MIT Press.

Sayama H. 2000 Self-Replicating Worms that Increase Structural Complexity through Gene Transmission. In Bedau M. A., McCaskill J. S., Packard N. H., and Rasmussen S. eds. Artificial Life VII: Proceedings of the Seventh International Conference, 467-476. MIT Press.

Sipper M. 1994 Non-Uniform Cellular Automata: Evolution in Rule Space and Formation of Complex Structures Artificial Life IV, In R. A. Brooks and P. Maes eds. Proceedings of the 4th International Workshop on the Synthesis and Simulation of Living Systems, 394-399.

Sipper M. 1998 Fifty years of research on self-replication: An overview. Artificial Life, 4(3): 237-257.

Vit*á*nyi P. M. B. 1973 Sexually reproducing cellular automata Mathematical Biosciences 18: 23-54.

von Neumann J. 1966 Theory of Self-Reproducing Automata. University of Illinois Press, Illinois. Edited and completed by A.W.Burks.

Wolfram S. 1984 Universality and Complexity in Cellular Automata Physica 10D, 1-35.

Wuensche A. 1999 Classifying cellular automata automatically; finding gliders, filtering, and relating space-time patterns, attractor basins, and the Z parameter.

Complexity, 4(3): 47-66.