# Parallelizing Automatic Induction of Langton Parameter with Genetic Programming

SANTIAGO GARCIA CARBAJAL*
DAVID W. CORNE** - ALEJANDRO CONTY*

* University of Oviedo, Campus de Viesques, Gijon, España
** Heriot Watt University, Edinburgh

**Abstract.** – *Many classifications for Cellular Automata have been proposed during time. One of them is based on Langton Parameter, $\lambda$. Depending on the probability of a cell of being active at one time, Cellular Automata are divided into four types. Experimentally, interesting Cellular Aautomata have been shown to have Langton parameter values close to 0.3. It is said that near this value, Artificial Life is possible. We use a Cenetic Programming technique to obtain transition rules with any desired value for $\lambda$. Exploring an environment of the theorethical chaos limit, and measuring the enthropy of the resulting automata, we search for Cellular Automata with interesting behavior.*

## 1. Introduction. Cellular Automata.

Cellular automata were originally conceived by Ulam and von Neumann in the 1940s to provide a formal framework for investigating the behavior of complex, extended systems [2], [3]. They are dynamical systems in which space, time, and the states of the system are discrete. Each cell in a regular lattice changes its state with time according to a rule which is local and deterministic. All cells in the lattice obey the same rule, and their state is determined by the previous states of a surrounding neighborhood of cells [4], [5].

The infinite or finite cellular array (grid) is n-dimensional, where $n = 1, 2, 3$ is used in practice. The identical rule contained in each cell is essentially a finite state machine, usually specified in the form of a rule table (also known as the transition function), with an entry for every possible neighborhood configuration of states. The neighborhood of a cell consists of the surrounding (adjacent) cells. For one-dimensional CAs, a cell is connected to r local neighbours (cells) on either side, where r is a parameter referred to as the radius (thus, each cell has $2r + 1$ neighbours, including itself). The value $a_i^t$ of a site at position i evolves acording to $\oint$.

$$a_i^{(t+1)} = \oint (a_{i-r}^{(t)}, a_{i-r+1}^{(t)}, .., a_{i+r}^{(t)}) \tag{1}$$

The local rule $\oint$ has a range of $r$ sites. Its form determines the behaviour of the cellular automaton.

For two-dimensional CAs, two types of cellular neighborhoods are usually considered:

- von Neumann neighborhood: The von Neumann neighborhood of range $r$ is defined by

$$N^V(x_0, y_0) = \{(x, y) : |x - x_0| + |y - y_0| \leq r\} \tag{2}$$

- Moore neighborhood: The Moore neighborhood of range $r$ is defined by

$$N^M(x_0, y_0) = \{(x, y) : |x - x_0| \leq r, |y - y_0| \leq r\} \tag{3}$$

### 1.1. *Genetic Programming as a way of generating transition rules for Cellular Automata.*

Genetic Programming [6] is directly applicable to the generation of transition rules for Cellular Automata, as the evaluation of a GP tree formed by logical operations in the internal nodes, with variables representing the neighbours of a cell in the terminal ones, gives us the next state as a function of the neighbors' previous one. For Cellular Automata with more than two possible states, the approach is still valid, as we only have to change logical operations for arithmetic ones, and include a wrapper function to obtain as many outputs as desired. For such a transition rule, we can aproximately calculate the value for Lagton Parameter, by simply generating a number of test cases, and counting the number of positive and negative evaluations of the tree. Consequently, we can use the value of $\lambda$ as fitness function ($|P(a) - \lambda|$, really, if we are trying to minimize the valueof the fitness function).

In order to obtain populations of Cellular Automata with interesting behaviour, we use our standar Genetic Programming System. After setting the initial configuration of the GP system, and when the evolution process is finished, we obtain transition rules whose probability of activation for any cell ($\lambda$) is close to the desired one, in most cases.

## 2. Parallel implementation

3D Cellular Automata can be parallelized in a straightforward manner. We just have to split the three dimensional mesh into as many sections as processors, or threads, we want to use. Each one of them will work on that section in order to evolve the transition rule, calculate entropy, or whatever. The schema is directly implemented using OpenMP, and scales correctly, as can be seen in Figure:

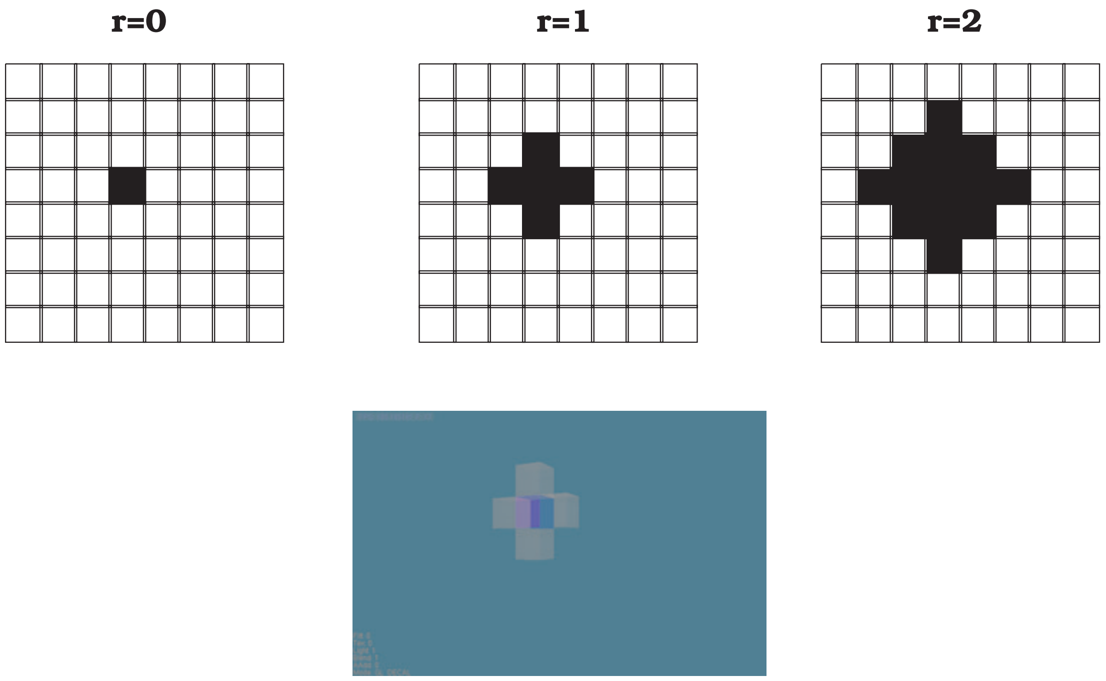

FIGURE 1. – *von Newmann Neighborhood.*

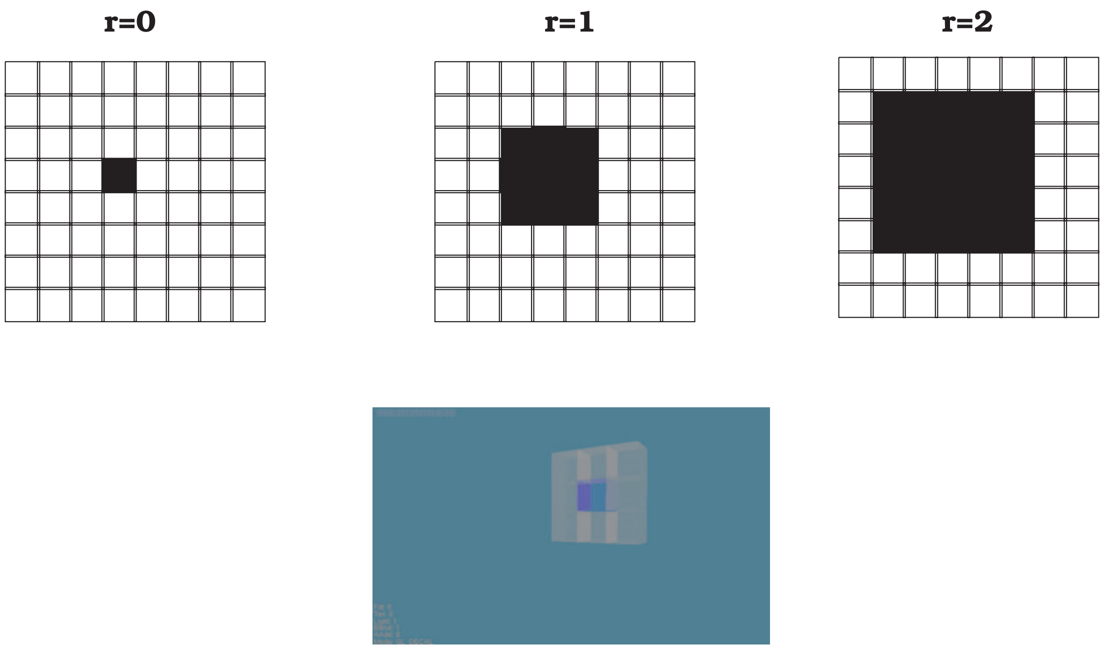

FIGURE 2. – *Moore Neighborhood.*

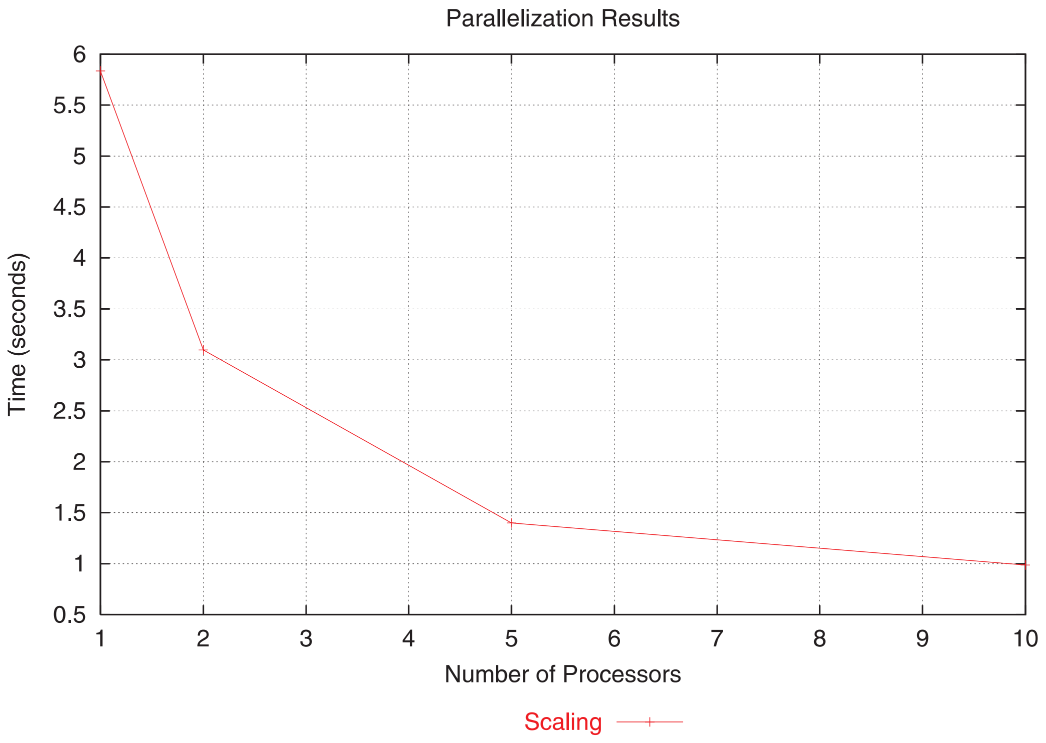

FIGURE 3. – *Parallelization Results.*

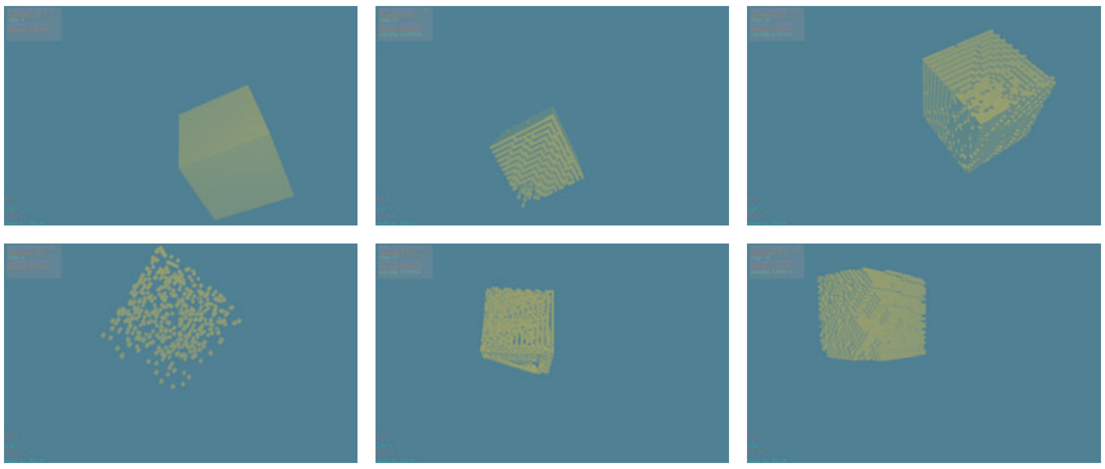

FIGURE 4. – *The tool.*

## 3. Visualization

In this work, we have developed a tool to visualize the behaviour of the 3D Cellular Automata. The goal is to observe if the value of Langton Parameter, $\lambda$, has any influence in the Entropy and Mutual Information measures showed in the Automata, and to isolate complex, cyclic, or self-reproductive patterns.

## 4. Results

The table shows the medium, variance, maximum an minimum values for entropy and Mutual Information at each estimated value of Langton Parameter.

TABLE 1. – Results

| $\lambda$ | Avg | $\delta$ | Max | Min | Avg | $\delta$ | Max | Min |
|---|---|---|---|---|---|---|---|---|
| 0.1 | 0.06086 | 0.0115 | 0.7155 | 0.01546 | 0.4593 | 0.0003549 | 0.8839 | 0.4068 |
| 0.15 | 0.1884 | 0.05447 | 0.8465 | 0.01671 | 0.4617 | 0.003178 | 0.9016 | 0.3887 |
| 0.2 | 0.3160 | 0.1114 | 0.8572 | 0.01734 | 0.4570 | 9.18306e-05 | 0.5821 | 0.4491 |
| 0.25 | 0.2208 | 0.06745 | 0.8200 | 0.01669 | 0.4751 | 0.004670 | 0.9016 | 0.4009 |
| 0.3 | 0.3808 | 0.08601 | 0.8251 | 0.01791 | 0.4884 | 0.007579 | 0.9023 | 0.3941 |
| 0.35 | 0.4553 | 0.09747 | 0.9323 | 0.01735 | 0.4737 | 0.003877 | 0.9016 | 0.3950 |
| 0.4 | 0.3724 | 0.11014 | 0.8607 | 0.01677 | 0.4750 | 0.001954 | 0.8077 | 0.3959 |
| 0.45 | 0.43202 | 0.07929 | 0.9445 | 0.01983 | 0.4867 | 0.007462 | 0.9016 | 0.3978 |
| 0.5 | 0.41656 | 0.06839 | 0.9373 | 0.01825 | 0.4856 | 0.006374 | 0.9016 | 0.3986 |

*Acknowledgements.* This work was carried out under the HPC-EUROPA project (RII3-CT-2003-506079), with the support of the European Community – Research Infrastructure Action under the FP6 "Structuring the European Research Area" Programme).

## References

[1] S.S. SAMPLE, Computer Phys. Sample, **10**, 100 (1900).
[2] J. VON NEUMANN, Theory of Self-Reproducing Automata (1966).
[3] MOSH SIPPE, Machine nature: The Coming Age of Bio-inspired Computing (2002).
[4] S. WOLFRAM, Universality and complexity in cellular automata, *Physica D*, **10**, 1-35, (1984).
[5] T. TOFFOLI and N. MARGOLUS, Cellular Automata Machines (1987).
[6] J.R. KOZA, Genetic Programming: "On the Programming of Computers by Means of Natural Selection" (1992).