Introducing Discrete Dynamic Systems in Algebra Teaching Process

PAOLA SZEKIETA, ALICIA TINNIRELLO, EDUARDO GAGO
Computer and Basic Sciences Laboratory
Universidad Tecnológica Nacional – Facultad Regional Rosario
Zeballos 1341, Rosario, Santa Fe
ARGENTINA
paolasz@gmail.com, amtinni@gmail.com, eagago@gmail.com

Abstract: - The aim of this work is to present a university classroom experience where the concepts of discrete dynamic systems are introduced in Algebra and Analytical Geometry subject with the purpose of using simulations where matrices are involved, in this case cellular automaton models are used as a dynamical system with discrete values in space, time and state. In this experience, computer scientists and mathematicians work together to carry out interdisciplinary projects which present discrete data management to first-year engineering students. Taking into account the fact that cellular automata have been used in different disciplines successfully, the authors of this paper consider introducing its concepts and applications in engineering teaching process.

Key-Words: - Discrete dynamic systems, cellular automata, computational mathematics

1 Introduction

This innovative activity is carried out by the Computer Laboratory of Basic Science of our University in order to introduce the mathematical developments to discrete variable events. The starting point is not whether content or processes have priority in the learning process, but making sure that learning becomes meaningful and functional.

Computer tools currently available are used to develop students' skills in the design of mathematical modeling with one discrete variable by teaching the fundamental basics of cellular automata (CA) in order to show the existing relations with symbolic calculus and the applications which these have with system resolution and modeling.

These applications are wide, ranging from microscopic simulations of Physics and Biology to macroscopic simulations of social and geological processes (Our translation) [1].

CAs are among the simplest mathematical representations of dynamical system that consist of more than a few – typically nonlinearly – interacting parts [2].

As such CAs are extremely useful idealizations of the dynamical behavior of many real systems, including physical fluids, molecular dynamical systems, natural ecologies, military command and control networks , economy fire spreading, epidemiology and many others [3] [4]. Because of their underlying simplicity CAs are also powerful conceptual engines with which to study general pattern formation [2].

CAs consists of a regular array of identically programmed units called cells or sites that interact with their neighbors' subjects to a finite set of rules prescribed by local transitions. All sites make a regular lattice and they evolve in discrete time steps as each site assumes a new value based on the values of some local neighborhood of sites and a finite number of previous time steps [5].

As M. Resnick suggests, the performance of this model is governed, not by a centralized authority but by the local interaction among decentralized components [6].

2 Concepts of Cellular Automata

According to Wolfram: CA are examples of mathematical systems constructed from many identical components, each simple, but together capable of complex behavior [7]. Some basic characteristics as regards the structure which the CA has are described. They represent a discrete system where the space, the time and the states of the system are all discrete and have the following properties: Space is represented by a regular lattice in one, two, or three dimensions; each site, or cell in the array of the CA can be in one of a finite number of states [8].

2.1 Neighborhood

The neighborhood of a lattice site consists of the site itself and its nearest neighbor sites, called *neighbors*. Two kinds of neighborhoods are commonly defined for a rectangular lattice:

A Von Neumann neighborhood consists of the site and the four nearest neighbors, situated above, below, right and left as shown in Fig.1 below.

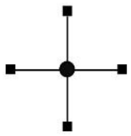

Fig.1: Von Neumann neighborhood

A Moore neighborhood consists of the site and the eight nearest neighbors as shown in Fig.2 below.

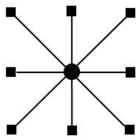

Fig.2: Moore neighborhood

2.2 Lattice Boundaries. Periodic Boundaries

The nearest neighbors of sites along the sites of a lattice are determined differently for various boundary conditions. The way these conditions are defined will impact directly on the automata behavior. The periodic boundaries which are used in the modeling of the CA used in the target activity are defined. To illustrate this criterion, the corresponding Moore neighborhoods are shown below for each site in the following simple lattice:

1 2 3
4 5 6
7 8 9

This type of boundaries is defined when the neighbors of the sites on the borders of the lattice are set in the following way:

9 7 8   7 8 9   8 9 7
3 **1** 2   1 **2** 3   2 **3** 1
6 5 4   4 5 6   5 6 4

3 1 2   1 2 3   2 3 1
6 **4** 5   4 **5** 6   5 **6** 4
9 7 8   7 8 9   8 9 7

6 4 5   4 5 6   5 6 4
9 **7** 8   7 **8** 9   8 **9** 7
3 1 2   1 2 3   2 3 1

The nearest neighbor left of a site on the left border is the site in the same row on the right border. In the same way, the neighbors on the right of the cells on the right border are analyzed. The nearest neighbor above site on the top border is the site in the same column on the bottom border. In the same way, the neighbors on the bottom order are analyzed.

2.3 Evolution Rule

Another basic component worth mentioning is the Evolution Rule which defines the state of each cell according to the immediate previous state of the neighborhood. This evolution is determined by a mathematical function which captures the influence of the neighborhood over the target cell.

2.4 Virtual Clock

The virtual clock is a clock which will generate simultaneous ticks to every cell indicating that the evolution rule must be applied to modify or maintain the state of the cell. This component fulfills the parallelism condition, i.e. all the cell area updated at the same time [9].

3 Game of Life algorithm

The Game of Life, which was created by the British mathematician J. H. Conway in 1970s, is the most famous CA. More computer time has been spent on running this game than on any other calculation and it was the first program executed by the *Connection Machine*, the world's first parallel computer. According to Gaylord & Wellin: it is the forerunner of so-called artificial life (or a-life) systems which are of great interest today, not only for their biological implications, but for the development of so-called intelligent agents for computers [5].

This automaton is a game of cero players, which implies that its evolution is determined by its initial set-up and there is no need of any further data entry. The game unfolds over a bidimensional grid as the game board. Each position on the board is called *cell* and it has 8 neighbor cells which are the nearest to each of them, including the diagonal ones (Moore neighborhood). The cells have two states, *living* or *dead*, which are represented by the numbers 1 and 0 respectively. The number and arrangement of living cells on the board evolve along the discrete time units. All cell states are taken into account to calculate their state in the following time. All cells are updated simultaneously. The transitions depend on the number of neighbor cell which are alive. A dead cell with exactly 3 living neighbors will be born in the next turn. If a living cell has 2 or 3 neighbor living cells, the following turn it will still be living. In any other case, it will die or remain dead due to *loneliness* or *overpopulation*.

The game set out will continue until 2 identical consecutive states are obtained, or rather, until a certain number of predetermined transitions are reached.

To start the modeling of this game, an initial cell array over the board is set at the time $t = 0$, represented by the following grid:

$$\begin{matrix} 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{matrix}$$

Fig.3 below shows each element on the board and its neighbors considering a Moore neighborhood with periodic boundaries.

$$\begin{pmatrix} \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 1 \end{pmatrix} & \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 1 & 0 \end{pmatrix} & \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix} & \begin{pmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \\ \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 1 & 1 \end{pmatrix} & \begin{pmatrix} 0 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 1 & 0 \end{pmatrix} & \begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{pmatrix} & \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{pmatrix} \\ \begin{pmatrix} 0 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 1 & 0 \end{pmatrix} & \begin{pmatrix} 1 & 1 & 0 \\ 1 & 1 & 0 \\ 1 & 0 & 0 \end{pmatrix} & \begin{pmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} & \begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{pmatrix} \\ \begin{pmatrix} 0 & 1 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{pmatrix} & \begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} & \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} & \begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix} \end{pmatrix}$$

Fig.3: Moore neighborhood on initial grid

It is possible to determine the number of living neighbors in each of the initial positions, counting the numbers of living neighbors which are around each cell:

$$\begin{matrix} 4 & 3 & 2 & 2 \\ 4 & 3 & 3 & 3 \\ 4 & 4 & 2 & 3 \\ 3 & 3 & 2 & 3 \end{matrix}$$

Comparing the state of each cell of the game board in time t and the number of living neighbors, the following state in time $t + 1$ can be set up.

Fig.4 below shows the transition from initial time $t = 0$ to $t = 1$. To visualize some examples, if we consider de state of de second element on first board row, it has 3 living neighbors so this cell will be born at t=1 but first cell on third row of the board will be dead at next turn due to overpopulation.

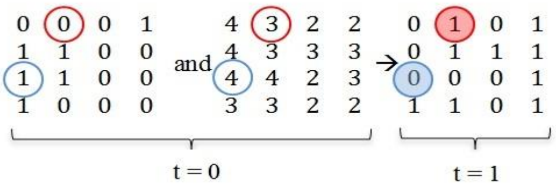

Fig.4: Evolution from $t = 0$ to $t = 1$

Fig.5 shows evolution from $t = 1$ to $t = 2$. On the last row of the game board two cases of surviving rules are highlighted.

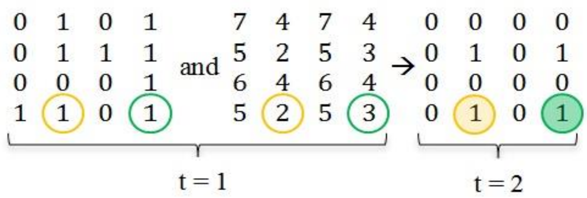

Fig.5: Evolution from $t = 1$ to $t = 2$

At the time instance $t = 3$, a board with all dead cells is obtained as shown in Fig.6 below. The following turn, time $t = 4$, the same result will be obtained, so the game will finish by obtaining two consecutive similar states.

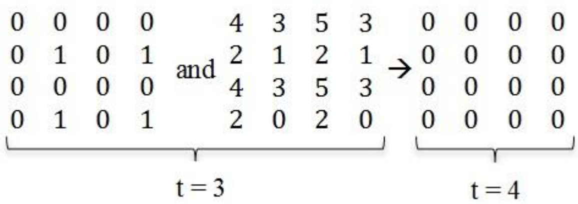

Fig.6: Evolution from $t = 3$ to $t = 4$

4 Laboratory Project: Defining and developing a computational simulation model

In the context of meaningful learning, the students' activities must be oriented in a school system based on research and development of appropriate strategies for connecting and integrating the computational mathematics and the basic technologies and applied in Engineering to promote the multidisciplinary approach to the curriculum content corresponding to the plans of study, aiming to train professionals capable of solving complex models with the use of technologies.

The existence of simulation tools transformed the programming environments toward more collaborative spaces, with the updated listings of increasingly complex systems but with broad application in the various areas that comprise the engineering, it is possible to design methodological strategies that integrate the knowledge of the compartmentalized disciplines.

The developments that have experienced the mathematical software and the affinity that the students have to be linked with the technologies, imposes on the university teachers makes the effort to transform the teaching-learning process in the process of learning investigating [10].

The present experience shows the representation of the Game of Life using specific software (MATHEMATICA, Wolfram Research). To do this, the board and the way neighborhood for each cell is obtained as well as the transition rules should be set up.

The board is represented by a square matrix of order 4 St, and each of its entries is the state of a particular cell at a given time t.

The following step is to define the function which returns the number of neighbors alive of each cell of the board. To model this automaton, the Moore neighborhood is considered which is made up by the 8 neighbors around the position which is often identified with a cardinal point according to the position of the central cell: north, northeast, east, southeast, south, southwest, west and northwest. Fig.7 below shows this Moore neighborhood.

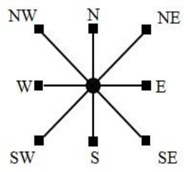

Fig.7: Moore neighborhood

It is possible to obtain a matrix that shows a particular neighbor by performing elementary operations on S_t.

For example, to obtain a matrix N_t whose elements n_{ij} represent north neighbor of each site s_{ij} in S_t at a certain time t, it is necessary to move down every row on S_t by properly interchanging them. Ec. (1) shows N_o that is north neighbors of each site s_{ij} at time $t = 0$.

$$\begin{pmatrix} 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix} \xrightarrow{\substack{f1=f4 \\ f2=f1 \\ f3=f2 \\ f4=f3}} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \end{pmatrix} = N_0 \quad (1)$$

To find the matrix NE_o whose elements represent the neighbor in the Northeast position of each s_{ij} in S_o, first move the rows (f) downwards as shown in (1) above and then, on this resulting matrix, interchange columns (c) to the left. See (2) below.

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \end{pmatrix} \xrightarrow{\substack{c1=c2 \\ c2=c3 \\ c3=c4 \\ c4=c1}} \begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \end{pmatrix} = NE_0 \quad (2)$$

Similarly, it is possible to obtain matrices that show a particular neighbor for each position in the state space and therefore to know the number of living neighbors of s_{ij} adding these 8 matrices of neighbor positions.

$$\underbrace{\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \end{pmatrix}}_{N_0} + \underbrace{\begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \end{pmatrix}}_{NE_0} + \underbrace{\begin{pmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix}}_{E_0} + \underbrace{\begin{pmatrix} 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}}_{SE_0} +$$

$$+ \underbrace{\begin{pmatrix} 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}}_{S_0} + \underbrace{\begin{pmatrix} 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix}}_{SW_0} + \underbrace{\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}}_{W_0} + \underbrace{\begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \end{pmatrix}}_{NW_0} =$$

$$= \begin{pmatrix} 4 & 3 & 2 & 2 \\ 4 & 3 & 3 & 3 \\ 4 & 4 & 2 & 3 \\ 3 & 3 & 2 & 3 \end{pmatrix} = V_0$$

Fig.8: Number of living neighbors' matrix at $t = 0$

Fig.8 above shows how to obtain number of living neighbors matrix at time $t = 0$.

Analyzing values of homologous elements on S_t and V_t, the next state into which s_{ij} will evolve can be obtained.

The evolution rule is function to the state of a cell s_{ij} and the number of living neighbor which it has.

Rule [st_{ij}: cell state s_{ij}, vt_{ij}: number of living neighbors s_{ij}] = st+l_{ij}.

A living site with two living nearest neighbor sites remains alive: $Rule[1,2] = 1$.

Any site (no matter if living or dead) with three living nearest neighbor sites stays alive or is born: $Rule[\_,3] = 1$.

All other cases, one cell either remains dead or die: $Rule[\_,\_] = 0$

The following matrices: S_o; S_1; S_2; S_3; S_4 show the consecutive states which the game reaches at each instance t. Matrices can be represented graphically with a black site for living cells and a white one for dead cells. Fig.9 shows both representations.

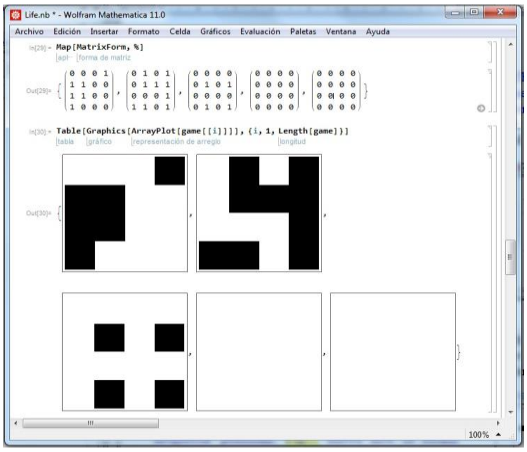

Fig.9: Game consecutive states

These rules are simple enough for anyone to understand, yet the Game of Life leads to an endless number of different patterns, and to significant complexity [11]. It is interesting to observe different these patterns or *life forms*. Students investigated these patterns and modified initial set-up to watch diverse evolution processes of the game. Laboratory work classes are an integral part of any educational program and their purpose is bringing the students closer to real situations of the area of studies.

Some of these cases are shown below Fig.10, Fig.11 and Fig.12.

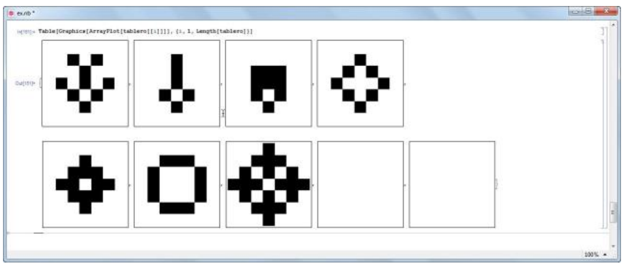

Fig.10: Cross pattern

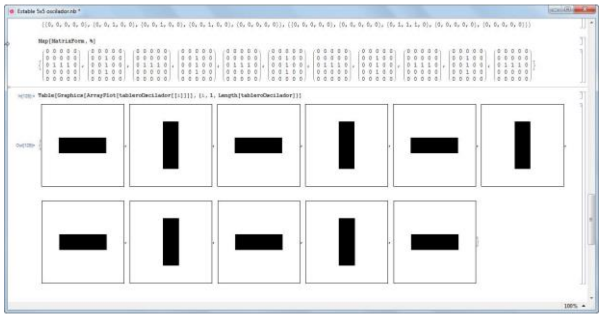

Fig.11: Oscillator

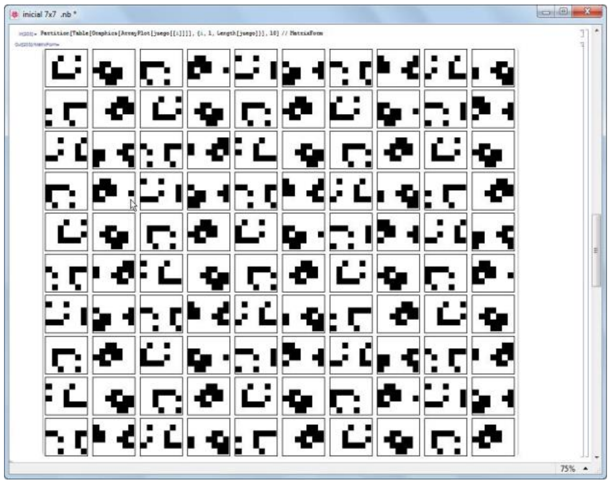

Fig.12: Spaceship pattern

5 Engineering application: Pollutant diffusion in a liquid form

CA application is the pollutant diffusion in a liquid form, for example the water flows which are presented in the projects of environmental remedial. In fact, this tool can be useful to guide and support the processes applied to purify contaminated water. It is not easy to obtain a numeric solution from a mathematical expression for such a phenomenon. However, there are ways to approach the problem by mathematical modeling of fluid transport: some of these methods will be studied in higher courses of engineering careers, CAs often provide a simpler tool that preserves the essence of the process by which complex natural patterns emerge [12].

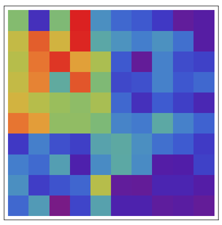

Fig.13: Initial concentrations of pollutant

Fig.13 shows a model of a liquid form with high pollutant concentrations on the high left corner of the analyzed section.

Fig.14: CA Simulated Pollutant Diffusion

Fig.14 above shows the evolution of the pollutant diffusion throughout time until all section under analysis is covered and a constant concentration is reached:

Students are required to interpret the software graphic output of pollutant diffusion and to compare it with other software outputs which can result from changing the initial configurations of the pollutant concentrations. This mathematical model is applied to understand the distribution of pollutants by formulating a 2D diffusion.

4 Conclusion

The CAs have been used in different disciplines successfully. Currently, attention is raised towards the development of models which can carry out complex tasks such as cryptography, image processing and turbulence analysis, among others.

This is the reason why we consider important to introduce CA concepts and applications in engineering teaching, taking as a starting point the study of mathematical models with discrete variable systems.

By introducing experiences as the ones described in the present paper for Algebra and Analytical Geometry, we search for a change of perspective which can see algorithms as a mathematical key activity and computer science as complementary knowledge to run those algorithms and manage their outputs.

Strategy design is highlighted as the outset of the study of mathematical models to solve discrete variable problems integrating the computational mathematics with the technological areas of the engineering curriculum while incorporating new learning styles.

References:

[1] G. Merino, *Use of a Cellular Automaton to Create a Diffusion Model of Pollutants in a Soil-Water System*. Journal of Mathematics: Theory and Applications, Vol.18, Nro.1, 2011, pp. 63-76.

[2] A. Ilachinski, *Cellular Automata: A Discrete Universe*, World Scientific, 2001, pp. 175-185.

[3] J. Quartieri, N. Mastorakis, G. Iannone, C. Guarnacci, *A Cellular Automata Model for Fire Spreading Prediction*, Proceedings of 3rd WSEAS International Conference on Urban Planning and Transportation, 2010, pp. 173-179.

[4] M. Dascalu, G. Stefan, A. Zafiu, A. Plavitu, *Applications of Multilevel Cellular Automata in Epidemiology*, Proceedings of the 13th WSEAS international conference on Automatic control, modelling & simulation, 2011, pp. 439-444,.

[5] R. Gaylor, P. Wellin, *Computer Simulations with Mathematica: explorations in complex physical and biological systems*. Springer-Verlag, 1995.

[6] M. Resnick, *Turtles, Termites, and Traffic Jams*, MIT Press, 1994.

[7] S. Wolfram, *Cellular automata as Model of Complexity*. Nature, Vol.311, 1984, pp. 419-424.

[8] R. Gaylor, K. Nishidate, *Modeling Nature: Cellular Automata Simulations with Mathematica*, Springer-Verlag, 1996.

[9] J. Muñoz, *Autómatas Celulares y Física Digital*, Memorias del Primer Congreso Colombiano de Neuro Computación. Academia Colombiana de Ciencias Exactas, Físicas y Naturales, Bogotá, 1996.

[10] A. Tinnirello, E. Gago, M. Dádamo, M. Valentini, *Design, Simulation and Analysis of a Fluid Flow System through Multiphysics Platform*, Proceedings of 7th International Conference of Education, Research and Innovation, 2014, pp. 5847-5855.

[11] T. Ostoma, M. Trushyk, *Cellular Automata: Theory and Physics. A New Paradigm for the Unification of Physics*, Cornell University Library, 1999.

[12] S. Wolfram, *Computer Software in Science and Mathematics*, Scientific American, Vol.251, 1984.