EVOLVING CELLULAR AUTOMATA FOR DENSITY CLASSIFICATION TASK

Petre ANGHELESCU
University of Pitesti, Department of Electronics and Computers
petre.anghelescu@upit.ro

Keywords: cellular automata, bio-inspired systems, density classification task

Abstract. *This paper presents a solution for Density Classification Task (DCT) using Cellular Automata (CA). The density preserving property as well as the translation property of fundamental CA rules in 1-D are combined together to obtain the DCT solution. The project has been implemented in software using C# programming language. This work is totally related to the idea of understanding the impact of the inherently local information processing of CA on their ability to perform a coordinated computation at the global level, as mediated by an evolutionary process.*

1. INTRODUCTION

For the past 50 years cellular automata (CA) have established themselves as popular platforms to investigate complex phenomena. Their attractiveness stems in part from their ability to expose highly complex or even chaotic behavior from an initially simple spatial configuration and set of update rules. An important insight that CA may provide is that in contrast to real world systems their dynamical laws are not bound by the classical laws of physics. The updating rule depends only on the state of the component and on the state of its neighbors, thus no component can "observe" the whole system. It is usually easy to model biological and other complex systems with the help of CA.

On the other hand the problems which are intrinsically global in nature like the Density Classification Problem (also known as Density Classification Task (DCT)), first introduced by Packard, is a simple counting problem [1] but, is very difficult for any CA to get its solution, since it has neither the memory to keep a cumulative count nor the spatial range of perception to make a correct assessment. The DCT is therefore attracted a great deal of interest to many CA researchers to test and measure the computing ability of CA.

In the CA two-state version it is desired that the evolved cellular automaton becomes either all 1's or all 0's depending on whatever the initial configuration image was more than half 1's or more than half 0's, respectively. The evolution of such a CA algorithm is thus a means to "classify" the binary strings of 0's and 1's according to their frequency (density) [2].

Roughly speaking, densities of local regions are classified and these regions expand with time. Where there are regions of equal density of 0's and 1's, a checkerboard or white/black boundary signal is propagated indicating that the classification is carried out on a wider scale.

This paper starts with a short review of CA and of relevant related research. This is followed by the surveys on earlier work that presents the DCT, including the known solutions to be analyzed. Section 4 is about the implementation of the proposed algorithm to solve 1-D DCT problem. Finally, section 5 concludes the paper.

2. BASIC CONCEPTS ON CA

CA first introduced by J. v. Neumann [3] and further popularized by S. Wolfram by

publishing a book entitled "A New Kind Of Science"[4] encourage other researchers to work in this area because of its regular, modular, simpler and easily cascadable computing structure. Further, the parallelism property inherently embedded in the CA tool dramatically increases the performance of any application that uses it.

CA are mathematical idealizations of physical systems in terms of discrete space and time, where each cell can assume the value either 0 or 1, and the interactions are only local. Despite of its simple description CA has been extensively used as models for complex systems in different areas. CA models have been applied to fluid dynamics, plasma physics, chemical systems, growth of dendritic crystals, economics, two-directional traffic flow, image processing and pattern recognition, parallel processing, random number generation, and have even been used as a model for the evolution of spiral galaxies. CA has also been applied to several physical problems (modeling of very complicated physical or chemical processes, molecular computing, pattern generators), where local interactions are involved. In fact, CA represents a particular class of dynamical systems that enable to describe the evolution of complex systems with simple rules, without using partial differential equations.

A CA consists of a regular uniform n-dimensional lattice (or array). At each site of the lattice (cell), a physical quantity takes values. This physical quantity is the global state of the CA, and the value of this quantity at each cell is the local state of this cell.

Each cell of the CA is restricted to local neighborhood interactions only, and as a result it is incapable of immediate global communication. The neighborhood of the cell is taken to be the cell itself and some or all of the immediately adjacent cells.

The CA evolves in discrete steps, with the next value of one site determined by its previous value and that of a set of sites called the neighbor sites. The extent of the neighborhood can vary, depending among other factors upon the dimensionality of the cellular automaton under consideration.

The width of the neighbourhood of the cell *j* is the width of the neighbourhood at the j-*th* side of the array. Classical examples for cell neighbourhood are presented in Fig. 1 with 3 cells for one-dimensional cellular automata respective 5 cells for bi-dimensional cellular automata (*Von Neumann Neighbourhood*). In this case, for establishment of the next state of a one cell the direct neighbours are considerate. The so called *Moore Neighbourhood* with 3 cells for one-dimensional cellular automata respective 9 cells for bi-dimensional cellular automata considers both kinds the direct and the diagonal neighbours.

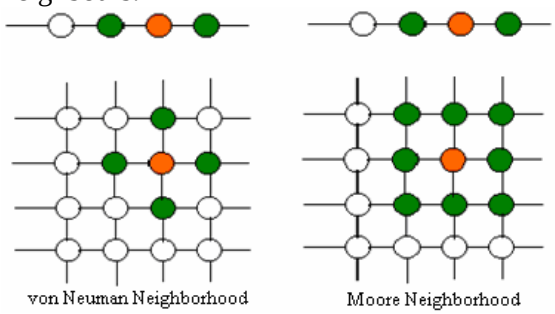

Fig. 1 – CA classical neighborhood

The state of each cell is updated simultaneously at discrete time steps, based on the states in its neighborhood at the preceding time step. The algorithm used to compute the next cell state is referred to as the CA local rule. Usually the same local rule is applied to all cells of the CA.

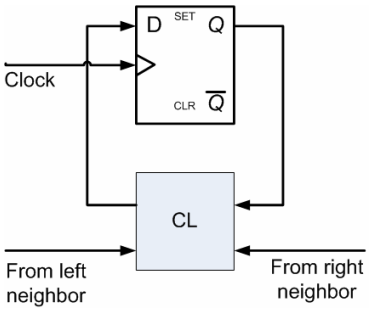

Fig. 2 – Typical CA cell

The cells evolve in discrete time steps according to some deterministic rule that depends only on local neighbours. In effect, each cell consists of a storage element (D flip-flop) and a combinational logic (CL) implemented the next-state functions (see Fig. 2). The combinational logic is called the "rule" of the CA.

The next-state function describing a rule for a three neighborhood CA cell can be expressed as follows:

$$a_i(t+1) = f[a_i(t), a_{i+1}(t), a_{i-1}(t)] \qquad (1)$$

where *i* is the position of an individual cell in one-dimensional array of cells, *t* is the time step, and *f* is the rule of CA.

For the cells a_1 and a_n it can be noted that different boundary conditions can be imposed such as Null, Periodic, Fixed, Adiabatic, Reflexive, Intermediate, etc. In case of Fixed boundary the extreme cells are connected by a pre-assigned fixed logic 0/1 states. If the extreme cells are connected by logic 0 states then it is called Null boundary CA. In case of Periodic boundary the extreme cells are connected to each other and form a cycle. In case of Adiabatic boundary the missing neighbors states (or virtual neighbors) are duplicate of the boundary cell values. In case of Reflexive boundary the value of left and right neighbors are same with respect to the boundary cell. In case of Intermediate boundary the value of the left (right) boundary will be the same as the cell value present at its next to next ((previous to previous) cell.

If the same CA rule determines the "next" bit in each cell of a CA, the CA will be called a Uniform Cellular Automaton; otherwise it will be called a Hybrid Cellular Automaton. In case of 1-D, 3-neighborhood, 2-state the number of all possible uniform CA rules is $2^8 = 256$. These rules are enumerated using Wolfram's naming convention [5] from rule number 0 to rule number 255 and can be represented by a 3-variable Boolean function. Under this perspective these three rules, which will be used in latter sections, are considered fundamental and are shown in Table 1.

Table 1. 3-fundamental 1-D CA rules with 3-neighborhood structure

Rules	7 111	6 110	5 101	4 100	3 011	2 010	1 001	0 000
30	0	0	0	1	1	1	1	0
90	0	1	0	1	1	0	1	0
150	1	0	0	1	0	1	1	0

CA, based on the statistical properties of the three neighborhoods, can be classified into four categories [5]:

Class 1: CA's that evolve to a homogeneous final global state.

Class 2: CAs in which each state lies in some cycle (periodic behavior).

Class 3: CAs that exhibit chaotic or pseudo-random behavior. This model is suitable for pseudo-random pattern generation.

Class 4: CAs having complicated localized and propagating structures. CA of this class are capable of universal computation.

Classes 1 and 2 are more or less self-explanatory. However, the temporally periodic evolution of a spatially disordered configuration ("spatial chaos") falls into class 2. In class 3, "chaotic" is taken to mean "spatio-temporal disorder", in the sense that the number of distinct space-time patches growth exponentially with the linear size of the patch along both the spatial and temporal axes. Class 4 has been the subject of speculation about its possible computational capacity. It has been suggested that CA in this class may be capable of universal computation, that is, of implementing a universal Turing machine. Class 4 was defined by the presence of spatially isolated transients of many shapes evolving for arbitrarily long times.

We can regard these cellular automata as a metaphor of a universe whose physics is reduced to some fundamental simple laws. Regarding the Universe we live in, it seems for the physicists not to have managed until now to discover a set of fundamental simple lows, but it is possible for them to do that taking into account their efforts made in this direction. The cellular automata suggest just the fact that the Universe could remain for us as mysterious as it was until now, even if those universal lows would be discovered [4]. The inventors of the cellular automata, S. Ulam and John von Neumann, wanted to show that the most elementary rules could lead to too complex consequences for someone to guess them. Although the local behaviour of the automatons is perfectly transparent, the overall behaviour it seems that will never be completely understood.

3. SURVEY ON IMAGE DCT

DCT as reported by [1] is a simple computing problem in which the CA starts with an initial configuration and forms a configuration with either all 1's or all 0's depending on which the higher initial density had. For example, if the CA's initial state contains 80% 1's and 20% 0's then there are more 1's than 0's so the CA should become 100% 1's and remain in that state. Like wise, if the initial state contains 40% 1's and 60% 0's then the CA should become all 0's.

It has been proved that no 1-D two-state CA can be constructed, which classifies binary strings according to their densities of 1's and 0's [6]. Further it has been demonstrated [7] that a solution to the density classification problem does exist, defining a different output in comparison to that of Packard and [8] demonstrated that a rule, which resolves the density classification problem, for a one-dimensional elementary CA, has to satisfy two conditions:

1. The density of the initial configuration must be preserved over time.
2. The rules table must exhibit density of 0.5 implying that it should be a balanced rule.

A group of researchers of the Santa Fe institute uses Genetic Algorithm (GA) to evolve CA rules, which are able to solve the problem of the DCT. Results of selected rules can be found in [9, 10].

An alternative, however, is the use of the algorithm developed by [11] and [12] for running a CA rule backwards. In other words, find a CA rule such that the whole of configuration-space is divided into precisely two basins of attraction; one of which consists of all rules with density greater than the critical density *q* and has as its attractor the state of all 1's, and the other consisting of all other configurations and having as its attractor the state of all 0's.

This then raises the possibility that one can now avoid some of the difficulties, both conceptual and computational, of the standard 'forwards' approach. For example, one might be able to more efficiently estimate the performance of a rule not by picking CA configurations at random and following each of them forward to their attractor. The main problem in this approach is the wide variation in quantitative results obtained when sampling different parts of the attractor basin.

In [13] a genetic algorithm evolves 1-D CA in order to perform the classical main task is reported. First time this paper uses higher number of states called multi-states CA and solves the density classification problem. Their result shows that there is a substantial homogeneity between elementary CA and multi state CA.

In [14] a pair of elementary rules, namely the "traffic rule" 184 and the "majority rule" 232, performs the density classification task perfectly.

A solution of 2-D DCT can be found in [15]. Result of this paper shows that, for two-dimensional task, a perfect rule does not exist. However, their experiments show a good performance even when radius of neighborhood is minimum.

An interesting question is to design a general algorithm to solve 1-D and 2-D density classification problem according to an arbitrary critical density q. To the best of my knowledge no earlier work exists that discusses this problem.

4. IMAGE DCT–CA ALGORITHM

The DCT algorithm presented in this paper for 1-D arrives from a two-phase procedure as a slight modification of the classical DCT definition originally introduced by Packard (1988). The initial CA configuration with the application of proper CA rules results in an intermediate CA configuration of 0's and 1's and then the CA picks up the classification decision based on the pre-produced densities.

The algorithm discussed here to solve 1-D DCT problem can be divided into two-phases:

- ✓ A preprocessing phase and
- ✓ A decision phase.

In the preprocessing phase the initial CA configuration of length *n* is evolved by CA rules to reach at a particular type of configuration after which a decision regarding the densities can be made in the decision phase.

In the preprocessing phase we use three fundamental 1-D CA rules. These rules are called 226 for all the cells from x_2 to x_{n-1}, 14 for cell x_1 and 64 for cell x_n in Wolfram convention [5]. These rules have the ability to translate 1's from left to right, from the bcms to the bcmps and preserves the number of 1's and 0's constant throughout the evolution. One such intermediate configuration to be used in the pre-processing phase of the proposed 1-D DCT algorithm is shown in fig. 3.

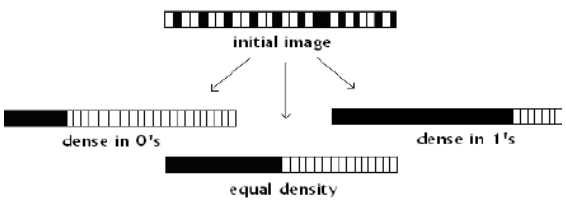

Fig. 3 – Possible intermediate outputs for taking decision in 1-D DCT algorithm

After preprocessing phase is over depending on the length of CA is either even or odd a decision is made in the decision phase.

For an odd length CA only two outputs are possible, which is either dense in 0's or dense in 1's. In this case only looking the value of the middle cell $x_{(n+1)/2}$ is either 0 or 1, it is possible to take the decision of classification.

In case of an even length CA, looking two cells value $x_{n/2}$ and $x_{(n/2)+1}$ situated at the middle and the (middle+1) positions respectively one can take the decision for their densities.

It may be mentioned here that in the decision phase two types of outputs are possible for DCT. In first type, one can print the output as the starting CA is dense in 0's or it is dense in 1's for odd length CA in addition with another possibility to print equal density in case of even length CA. This way of output is required for the DCT problem defined by (Capcarrere et al., 1996).

In the second type, knowing the information that the CA is either dense in 0's or dense in 1's a trivial rule: rule 0 or rule 255 may be applied to get an output of 1-D DCT defined by [1]. Thus in this way it can able to solve both the original DCT defined by [1] as well as the DCT defined by [7].

The proposed two-phase algorithm to solve 1-D DCT problem using the three CA rules (14, 226 and 64) is presented in fig. 4a and fig. 4b.

The preprocessing phase of the algorithm generates the required intermediate image like fig. 3. Starting from the initial configuration of length *n*, in the worst case CA is evolved up to (*n* −1) times. In each evolution process, rule 226 is applied to all the cells except the boundary cells (x_1 and x_n).

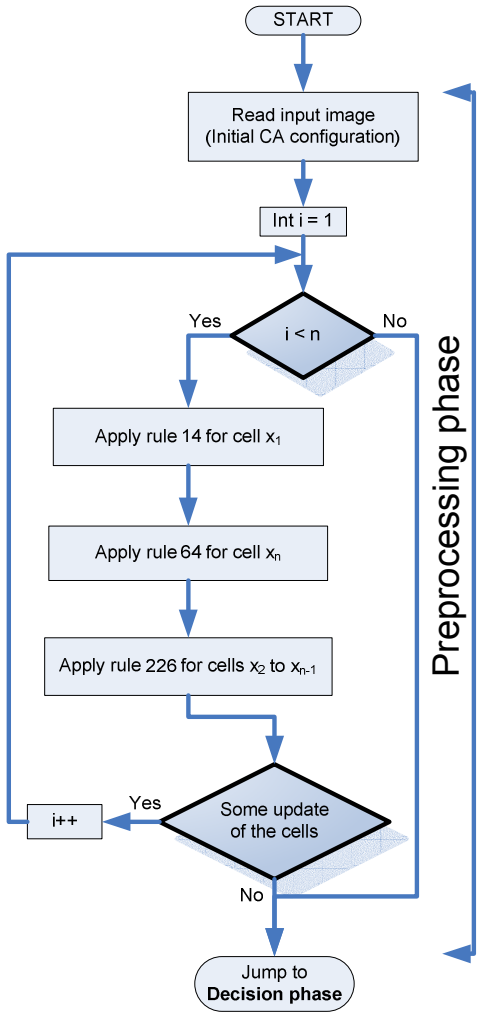

Fig. 4a – Diagram of the DCT preprocessing phase

For an odd length CA only two outputs are possible, which is either dense in 0's or dense in 1's. In this case only reading the middle cell value of the vector $x_1, \ldots, x_n$ one can take the decision of their densities.

For an even length CA, the decision for density classification is made on reading two cell values situated at the middle and the (middle+1) positions of the vector $x_1, \ldots, x_n$.

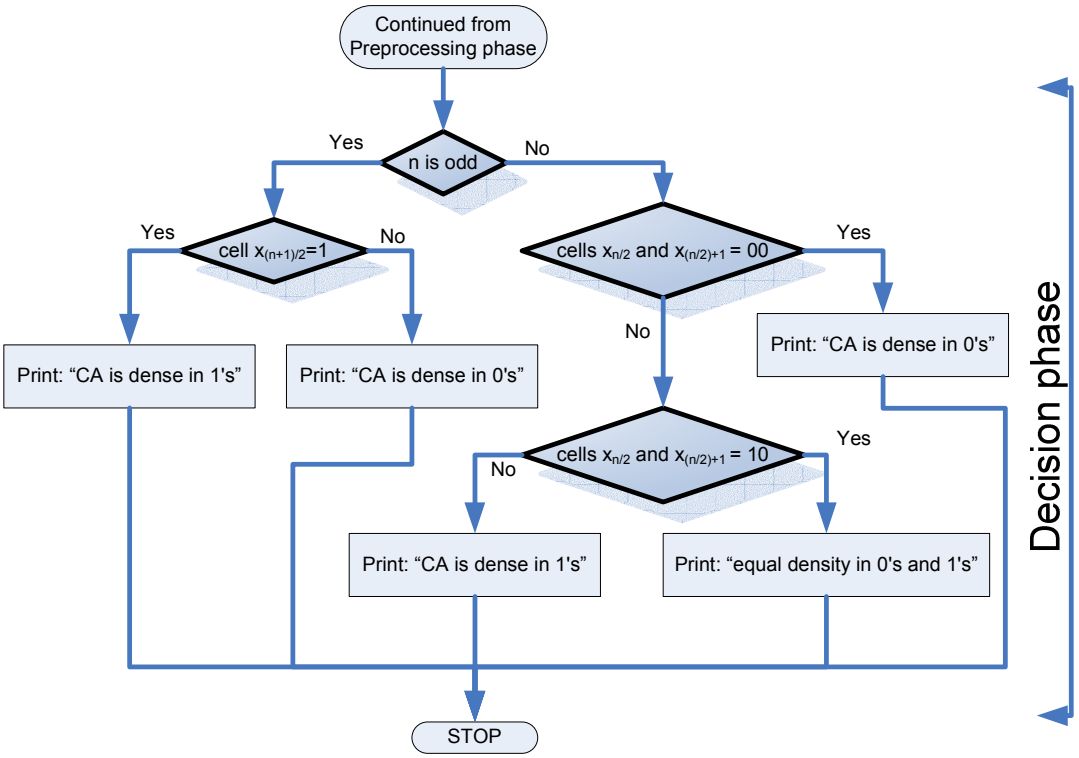

Fig. 4b – Diagram of the DCT decision phase

The complete DCT – CA application including the two phases (pre-processing and decision) was fully implemented in software using C# programming language. In Fig. 5 is presented a relevant instance of a demonstrative task applied to a common input composed from the following bits: 0000101111011.

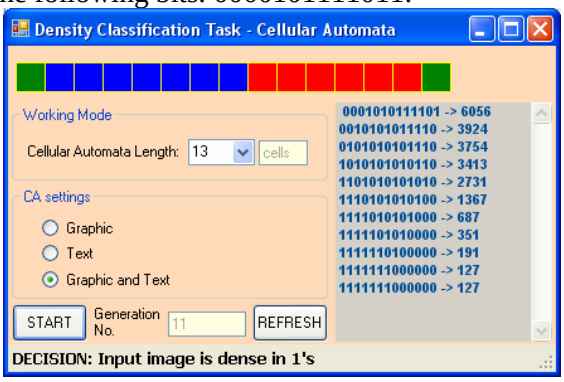

Fig. 5 – DCT – CA algorithm for a random input string of length 13

The following figures (fig. 6, fig. 7 and fig. 8) demonstrate the outputs obtained by the 1-D DCT algorithm by considering three different initial 1-D CA of length 16. Only the intermediate output is shown based on which a decision can be made.

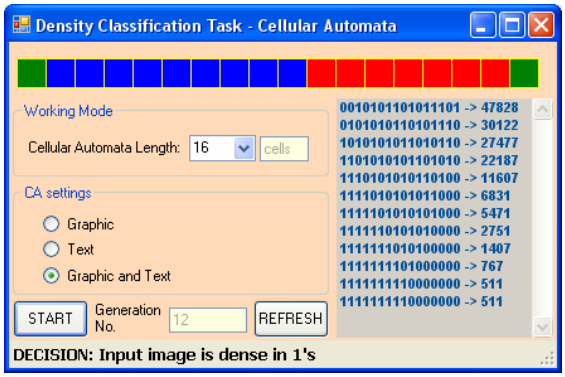

Fig. 6 – DCT – CA algorithm for an input string of length 16: 0010101101011101

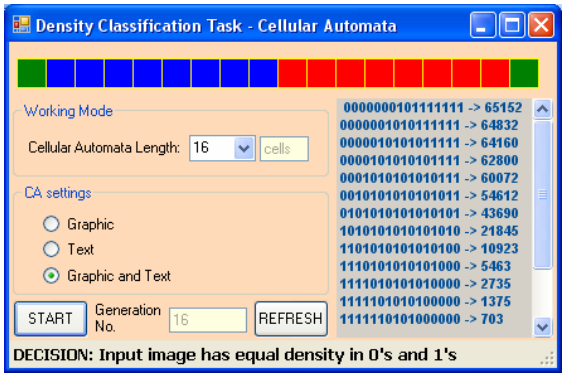

Fig. 7 – DCT – CA algorithm for an input string of length 16: 0000000011111111

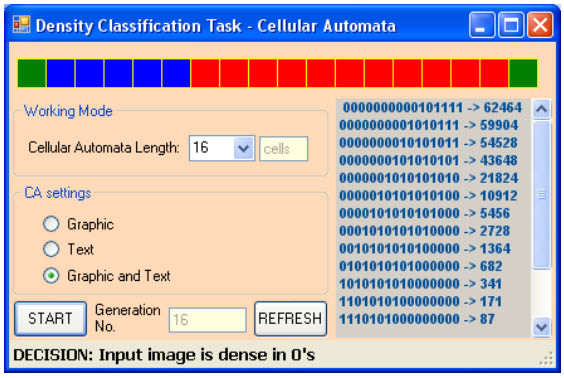

Fig. 8 – DCT – CA algorithm for an input string of length 16: 0000000000011111

The worst case time complexity of this algorithm is clearly O(n) because maximum ($n-1$) sequential evolutions are required in the pre-processing phase to get an intermediate configuration after which another evolution is required in the decision phase.

5. CONCLUSION AND FUTURE RESEARCH DIRECTIONS

This paper highlights some existing solution of DCT reported previously by various authors using conventional CA and given its software implementation using a 1D – CA. The CA uses three rules (14, 226 and 64) to achieve the desired DCT solution.

The above algorithm can be extend to bi-dimensional CA and may be useful to solve other pattern classification problems except DCT. It can be thought of as a 2-D filter that may be useful for separating the noise from some images.

Exploring all these applications, including implementation of 2D–CA DCT is our immediate future efforts.

References

[1]. Packard, N. H., *"Adaptation toward the edge of chaos"*, In: Dynamic Patterns in Complex Systems, World Scientific, pp. 293-301, 1988.

[2]. Schiff, L. J., *"CA – A discrete view of the world"*, John Wiley & Sons, Inc., 2007.

[3]. Neumann, von J., *"The Theory of Self - Reproducing Automata"*, Burks, A.W. (Ed.), Univ. of Illinois Press, London, 1966.

[4]. Wolfram, S., "A new kind of science", Wolfram Media Inc., 2002.

[5]. Wolfram, S., *"Theory and Application of Cellular Automata"*, World Scientific, 1986.

[6]. Land, M., Belew, R. K., *"No Perfect Two-State Cellular Automata for Density Classification Exists"*, Physical Review Letters 74(25), 5148-5150, 1995.

[7]. Capcarrere, M. S., Sipper, M., Tommasini, M., *"Two-state, r=1 Cellular Automaton that Classifies Density"*, Physical Review Letters 77, 4969-4971, 1996.

[8]. Capcarrere, M. S., Sipper, M., *"Necessary conditions for density classification by cellular automata"*, Physical Review E 64, 036113/1-036113/4, 2001.

[9]. Mitchell, M., Crutchfield, J. P., Das, R., *"Evolving cellular automata to perform computations"*, Handbook of Evolutionary Computation, Oxford University Press, 1998.

[10]. Jullie, H., Pollack, J. B., *"Coevolving the ideal trainer: Application to the discovery of Cellular Automata Rules"*, San Francisco, CA, Morgan Kaufmann, 1998.

[11]. Wuensche, A., *"The Ghost in the Machine; Basin of Attraction Fields of Random Boolean Networks, in Artificial Life III"*, ed. C.G. Langton, Santa Fe Institute Studies in the Sciences of Complexity, Addison-Wesley, Reading, MA, 1994.

[12]. Maji, P., Chaudhuri, P. P., *"Non uniform cellular automata based associative memory: Evolutionary design and basins of attraction"*, Science Direct, Information Sciences178, 2315-2336, 2008.

[13]. Gabriele, A. R., *"The Density Classification Problem for Multistates Cellular Automata"* Capcarrere, M. et al. (Eds.): ECAL 2005, LNAI 3630, pp. 443-452, Springer-Verlag Berlin Heidelberg, 2005.

[14]. Fuks, H., *"Solution of the Density Classification Problem with Two Cellular Automata Rules"*, Phys. Rev.: E55, R2081-R2084, Issue 3, 1997.

[15]. Reynaga, R., Amthauer, E., *"Two-dimensional cellular automata of radius one for density classification task"*, Pattern recognition Letters 24, 2849-2856, 2003.

This work was supported by CNCSIS UEFISCSU, project number PN II-RU PD 369/2010