Adaptive Strategies Applied to Evolutionary Search for 2D DCT Cellular Automata Rules

Gina M. B. de Oliveira Luiz G. A. Martins Enrique Fynn
Artificial Intelligence Laboratory, School of Computation, Federal University of Uberlandia
Av. João Naves de Ávila, 2121, bloco 1B, Campus Santa Mônica
CEP:38400-096, Uberlândia/MG, Brazil
+55 (34) 3239-4144
gina@facom.ufu.br gustavo@facom.ufu.br enriquefynn@comp.ufu.br

ABSTRACT
Cellular automata (CA) are able to perform complex computations through local interactions. The investigation of how CA computations are carried out can be made by the usage of CA rules to solve specific tasks. The well-known problem called density classification task (DCT) is investigated, with focus on its two-dimensional version. Evolutionary algorithms have been widely used in the search for DCT rules. A sample of lattices with Gaussian distribution is commonly used to evaluate rule quality. However, uniform lattices are easier to classify, allowing an initial selective pressure needed to start the convergence. A comparative evaluation of three adaptive strategies is presented here: they start using easy lattices to classify and as effective rules are being obtained the difficult level is progressively increased toward the target evaluation. Several experiments were performed to evaluate the strategies efficiency and new rules were found, which outperform the best ones published.

Categories and Subject Descriptors
I.2.m [**Artificial Intelligence**]: Miscellaneous.

General Terms
Algorithms, Performance, Experimentation.

Keywords
Cellular Automata, Evolutionary Algorithms, Density Classification Task.

1. INTRODUCTION
Cellular Automata have the potential to embody models of complex systems and to act as abstract machines that perform sophisticated computations with high degree of efficiency and robustness. The understanding of how CA computations are carried out is still vague. The most-common approach investigates specific tasks to be solved by CA rules. An efficient strategy is the usage of evolutionary algorithms to search over CA rule space for a transition rule able to exhibit a desired computational behavior. The most studied computational task is the Density Classification Task (DCT) [1]-[14][16][18]. Considering DCT, the goal is to find a CA rule transition that can classify the density of 1s in the initial lattice. This task has been historically studied in one-dimensional CA space, and recently the two-dimensional space was explored [8][11][12][14][18]. In the present work, we explore DCT in 2D space using Moore neighborhood, resulting in rules with 512 bits and a search space of high cardinality (2^{512}).

Considering DCT, the most difficult instances to be classified are the lattices with a distribution of 0s and 1s around 50%. Therefore, lattice samples with Gaussian distribution are commonly used to evaluate the rule efficacy to solve DCT. Mitchell and colleagues has identified that an evolutionary search of CA rules to solve DCT returns a better convergence if they were evaluated using lattice samples with a uniform distribution [6]. Uniform lattices are easier to classify and they provoke an initial selective pressure in the first generations shooting search convergence. Several subsequent works in DCT problem also applied this strategy to obtain a better convergence [1][3][8]-[10]. However, it was later verified that the gap between the easier evaluation performed during the evolution (based on uniform distribution) and the hardier evaluation carried out in the end of run (based on Gaussian distribution) represents an drawback to evolution of high efficacy rules [4]. Different adaptive strategies were used in recent works to deal with such gap, including coevolutionary approaches [3][4][12][18].

A high efficacy rule was published for 2D DCT in [18]. This CA rule was evolved using 21×21 lattices through a sophisticated two-level hierarchical evolutionary environment. Besides the two-tier mechanism, the achieved search success was attributed to the usage of a parameter named *BWLR-symmetry* (associated to Black/White and Left/Right equivalent transformations [17]) and to the application of a simple kind of cooperative coevolution that optimizes and adapts the fitness evaluation during the search. Later on, other high efficacy rule was published for 2D DCT in [12], evolved using 12×12 lattices. A much simpler evolutionary algorithm was used in this second work; it was also obtained using *BWLR-symmetry* information and a different type of adaptive evaluation, which was previously employed in 1D DCT search [3]. A comparative analysis of these both 2D DCT rules has highlighted a strong dependence of rules efficacy with the parity of the lattice size [12][9]. The 21×21 evolved rules are better in all the odd-size lattices analyzed while the 12×12 evolved rules surpass the previous ones in all even lattice sizes tested. Besides, 21×21 evolved rules presented a more severe

GECCO'11, July 12–16, 2011, Dublin, Ireland.

oscillatory behavior between even and odd size lattice than the 12×12 evolved one.

Since high efficacy 2D DCT rules were obtained using different adaptive evaluation approaches, we decided to investigate such strategies. The main goal of the present work is to clarify the role of different approaches guiding the evolution of two-dimensional cellular automata. The present work helps to understand the dependencies between approaches published before. Such kind of comprehension is relevant not only to DCT task itself, but also for researches using evolutionary methods to search for CA transition rules to solve other computational tasks or more specific behavior. For example, the basic framework proposed by Mitchell and colleagues investigating DCT [6] have been used in the search for synchronization task rules [12][18] and transition rules for CA-based scheduling environments [15]. In a more general context, this study can also be used by other Evolutionary Computation (EC) researchers who need to apply sampling strategies to do an efficient stochastic fitness evaluation.

Three adaptive evaluation approaches are investigated here. In the strategy named *A1*, the CA lattices used in rule evaluation are initially generated using a uniform distribution and as good rules are being obtained, an increasing portion of lattices is generated using Gaussian distribution. In strategy *A2*, initial lattice sizes are smaller than the target size and, as efficient rules are being obtained, this size is progressively incremented. For strategy *A3*, lattices are generated with Gaussian distribution being those with a distribution of 1s around 50% are discarded (inside a range 50% ± *d*) and the value of *d* is being decremented as efficient rules occur. Different experiments were carried out in the present work, the best ones using *BWLR-symmetry* parameter to guide GA search. In general, the most efficient strategies were *A1* and *A3*, returning the best rules for even and odd lattice sizes, respectively. Besides, it was possible to find better rules than the previously published ones.

2. CELLULAR AUTOMATA AND DENSITY CLASSIFICATION TASK

Cellular Automata (CA) are discrete dynamic systems composed by a large number of simple components with local connectivity. Basically, a cellular automaton consists of the cellular space and the transition rule. Cellular space is a regular lattice of N cells, each one with an identical pattern of local connections to other cells, and subjected to some boundary conditions. These cells are arranged in an n-dimensional space and the most studied arrangements are one-dimensional (1D) and two-dimensional (2D).

The simplest CA is the 1D binary structure [17] formed by an array of cells (the lattice), where each cell a_i can assume states 0 or 1. Cells interact locally in a discrete time t, usually in a parallel and synchronous way. The transition rule establishes how the states will change along time based on the current states of each cell and their immediate neighbors. For 1D CA, the neighborhood size m is usually written as $m=2R+1$, where R is the radius. The state of the cell a_i at time $t+1$ depends only on the states of itself and its neighbors at time t and it is determined by the transition rule τ:

$$a_i^{(t+1)} = \tau[a_{i-R}^{(t)},...,a_i^{(t)},...,a_{i+R}^{(t)}] \quad (1)$$

Considering 2D CA, the neighborhoods usually used are von Neumann and Moore. Von Neumann neighborhood is formed by 5 cells: the centre one and its four adjacent cells (east, west, north and south). Moore neighborhood is formed by 9 cells: the same five cells of Von Neumann neighborhood and the diagonal cells (NE, NW, SE and SW). Therefore, in Moore neighborhood there are 512 (2^9) different arrangements, each one can be represented by a 3x3 matrix. These arrangements can be linearly represented by sequence of its matrices rows ($a_{11}\ a_{12}\ a_{13} - a_{21}\ a_{22}\ a_{23} - a_{31}\ a_{32}\ a_{33}$) and the output bits of the rule can be lexographically ordered from 000-000-000 to 111-111-111. The usage of 2D CA with Moore neighborhoods is the focus of this work.

Although being very simple to implement, cellular automata are able to perform complex computations [8]. The computational power of CA has been investigated with emphasis in the study of 1D CA able to perform specific computational tasks [7]. The most widely studied CA task is known as the density classification task (DCT) [6]. In this task, the goal is to find a binary CA that can classify the density of 1s in the initial lattice, such that: if it has more 1s than 0s, the CA should converge to a null configuration of 1s; otherwise, it should converge to a null configuration of 0s.

The most studied configuration for DCT is defined by one-dimensional lattice consisted of 149 cells and CA rules with neighborhood of radius 3 [3][4][6][18]. Although the majority of published works investigates its original 1D DCT version, some studies are performed to solve this problem in two-dimensional rule space [8][11][12][14][18]. Figure 1 shows some steps of a temporal evolution of a 2D lattice using a rule with Moore neighborhood. This rule successfully solves DCT for an initial lattice with more 1's than 0's.

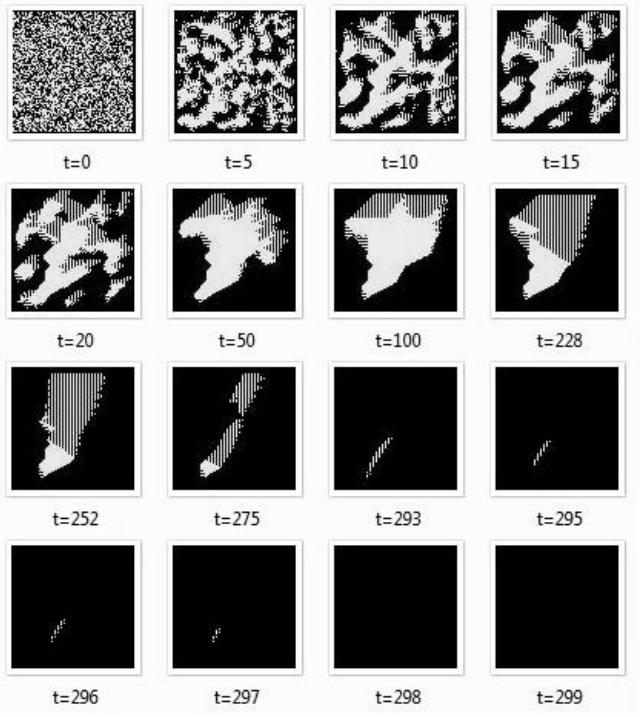

Figure 1. Snapshots of a rule evolution with Moore neighborhood solving DCT in 2D space.

DCT is a nontrivial task for a small-radius CA in any dimension, since they rely only on local interactions. On the other hand, this task is trivial for a system with a central controller or a central storage [7]. Performing this task well for a fixed lattice size requires more powerful computation than can be performed by a single cell or any linear combination of cells. Since the 1s can be distributed throughout the CA lattice, the rule transition must transfer information over large distances. A kind of global coordination is required to communicate cells separated by large distances and that cannot communicate directly. It was proven to be impossible to solve the DCT perfectly, by any one-dimensional cellular automaton with finite radius and periodic boundary conditions [5].

Although perfect solutions can be given to alternative formulations of the task [2], the best possible rule for the original formulation of Density Classification Task remains unknown. The trend to look for better and better DCT rules has led to better and better algorithms, more and more fine-tuned to the problem [1][3][4][6][8]-[14][16][18].

3. EVOLUTIONARY SEARCH FOR DCT RULES

Since a computational task is defined, it is not easy to find a CA rule that performs it. Manual programming is difficult and costly, and exhaustive search of the rule space becomes impossible, due to its high cardinality. A practical alternative has been the employment of EC methods [1][3][4][6][8]-[14][16][18]. Packard (1988) was the first to publish results using a genetic algorithm (GA) as a tool to find CA rules with a desirable computational behavior [13]. In this work, the genotype of the automaton was given by its transition rule and the phenotype by its ability to perform the required task. Crossover among two CA was defined by the creation of two new transition rules, out of segments of two other ones; mutation was achieved by the random flip of an output bit of the rule transitions. Other evolutionary computation techniques were used to find such kind of CA rules. Genetic programming was also used as a search procedure for CA rules to perform the DCT [1]. Furthermore, for the same task, an important radius-3 rule was obtained by a coevolutionary approach [4].

Another approach related to the present work is the usage of a heuristic based on some CA static parameters to guide the evolutionary search toward efficient DCT rules [3][9]. This approach uses parameter bands where good rules are more likely to occur. Once this information is available, it is used in an active way, as an auxiliary metric to guide the processes underlying the GA search. This approach has been previously used to find DCT and synchronization rules [3][12][9]. Different static parameters have been evaluated in the evolution of CA rules [3][16]; in the experiments described here *BWLR-symmetry* was used.

Any CA rule can have three dynamical equivalent rules obtained using three possible transformations: complementary, reflection and complementary-plus-reflection (or black-white, left-right and joint-black-white-and-left-right transformations) [17]. Previous works have shown that the symmetry of the bits related to the complementary transformation is related to the success of a rule to perform as specific task. In [3] it was observed that in the best DCT rules known until that moment the bits related to the complementary symmetry has an asymmetrical pattern. This observation was used in the evolutionary experiments reported in [3] to obtain a rule as good as the best rule known at that moment (JP rule discovered in [4]). *BWLR-symmetry* parameter is defined as the number of state transitions of a rule that respect the joint black-white and left-right transformations. In the case of good DCT rules, the amount of *BWLR-symmetry* should be maximal, or equal to 1 (in a range from 0 to 1). In Wolz and de Oliveira's work [18] the symmetry of the bits related to the complementary-plus-reflection transformation has revealed as the most significant to the success of a DCT CA rule. In fact, the value of this symmetry in the best rules published in [18] for 1D and 2D spaces are equal to 1. That is, these bits are totally symmetrical, turning that each one of these rules does not have a different complementary-plus-reflection equivalent rule as the same binary code is obtained when the transformation is applied. This information was used as a heuristic to guide the evolutionary search in [18] to the region of the best possible rules for DCT (both in 1D and 2D rule spaces). Heuristic was implemented as a repairing infeasible solutions procedure and it has caused the discovery of almost all good rules having the *BWLR-symmetry* equal to 1.

The specification of DCT problem using 2D rules with Moore neighborhood was first proposed by Morales and colleagues [8], which found a reasonable rule for this task using a GA-based environment. Their best rule has an efficacy of about 69% [8]. This rule was evolved and evaluated using the same lattice size: 21×21. Later on, this search was improved by Oliveira and Siqueira using a parameter-based heuristic and it was possible to find better rules [11]. The best rule found has 70.62% of efficacy in 12x12 lattices. 2D DCT was also investigated by Reynaga and Amthauer [14]. Subsequently, Wolz and de Oliveira could found high efficacy rules using a two-tier environment [18]. They could found several rules with efficacy above 82% in 21×21 lattices and the best one have an efficacy of 82.23%. Experiments was performed in [12][9] by Oliveira and colleagues to evaluate the rules published in previous works, verifying their efficacy in different lattice sizes, both odd and even-sizes. Although DCT was not commonly studied using even-size lattices due to the existence of configurations in which the task is not well-defined (exactly 50% of 1s), it is expected that any rule evolved for DCT has also a good performance in the even lattices in which the majority can be decided. The results obtained in [12][9] showed that the rule published in [18], which was evaluated in 21×21 lattices size, returns the best performance in odd-size lattices. However, rules evaluated in odd-size lattices present a severe oscillating behavior between odd and even lattices. For even-size lattices, the best efficacy was obtained in [12][9] by a rule evolved in 12×12 lattice size. It has an efficacy of 82.78% when tested in 100,000 lattices generated using a Gaussian distribution. This rule also presented an oscillating behavior related to the parity of the lattice size; but it is lesser than that observed for odd size evolved rules. As a consequence, the average of the efficacy obtained for this 12×12 evolved rule [12][9] in all tested lattice sizes (4×4 to 21×21) is greater than the average obtained by the best 21×21 evolved rule in [18]: 81.86% against 73.59%.

4. ADAPTIVE STRATEGIES FOR FITNESS EVALUATION

Rule fitness related to DCT is defined by the percentage of initial configurations (ICs) of lattices that density is successfully decided by the rule. It was established in [6] that the initial lattices used to evaluate the individuals should be generated by a uniform

distribution for a better convergence, since it allows the selective pressure to drive the evolutionary search. However, the best rule found is evaluated in a sample of random lattices with a Gaussian distribution in the end of each GA run. When IC bits are randomly generated, the peak of the distribution of 1s will be near 50% would make density decision hard. Adaptive approaches were used in CA rules evaluation to reduce the gap between this both distributions [3][12][9][18]. They start using easy lattices to classify and, as effective rules are being obtained, the difficult level is progressively increased toward the target evaluation.

Based on works [12][9] and [18], three adaptive evaluation strategies are comparatively investigated here. They are related to the way in which IC samples to be used in each generation evaluation are generated. Strategy *A1* starts from uniform distributed lattices and, as the rules becoming better, it is increase a portion of the lattice using Gaussian distribution. Strategy *A2* randomly generates lattices using Gaussian distribution with a small initial size range (4×4 or 5×5), and later on smoothly shifted towards the target size (20×20 or 21×21). Strategy *A3* randomly generates lattices using Gaussian distribution, but removing all ICs in the density range [0.50±*d*] for a given $d \in [0.02, 0.07]$, which is decreased at later stages of the search. *A1* was employed in [3] and [12][9] to find respectively 1D and 2D DCT rules. *A2* and *A3* were jointly employed in [18] to find both 1D and 2D DCT rules. The target evaluation is characterized by the lattices sample created with Gaussian distribution in the goal size (20×20 for even sizes and 21×21 for odd sizes); except when strategy *A3* is applied, because it must be excluded all ICs in the density range [0.48, 0.52] (for $d = 0.02$).

The dynamics of creating IC samples to perform rule evaluation depends on the adaptive approach used. The evaluation in the first generation starts from ICs different from the target: (*i*) *A1* generates a uniformly distributed sample of ICs with the target size (100% of lattices are created with uniform distribution); (*ii*) *A2* generates a Gaussian distributed sample of ICs with the size smaller than the target one (4×4 or 5×5 for even or odd-lattices, respectively); and (*iii*) *A3* generates a Gaussian distributed sample of ICs with the target size and *d* starts equal to 0.07, that is, lattices with density between 0.43 and 0.57 are excluded. As generations pass by and evolved rules achieve a pre-specified efficacy bound E_B, the creation of ICs is slightly modified towards of the target evaluation: (*i*) *A1* changes 5% of the current uniform distributed ICs to the ones created with Gaussian distribution; (*ii*) *A2* increases in 2 the current lattice size of the Gaussian distributed sample of ICs (e.g. from W x W to $W+2$ x $W+2$); and (*iii*) *A3* reduces the current value of d ($0.5 \pm d$) in 0.01. In the last generations, if the evolutionary search had found consecutive good rules (with efficacy above the limit E_B), ICs will be created according to the target evaluation.

5. EXPERIMENTS

This section presents experiments performed using the adaptive strategies *A1, A2* and *A3*. A very simple GA was implemented based in [12][9]. The individuals are 2D binary rules using Moore neighborhood. GA evolves a population (T_p) of 100 rules during 100 generations (N_g). Each individual evaluation was obtained out of testing the rule efficacy in 100 initial configurations (N_{IC}). Two-dimensional IC sample used in rule evaluation is created in each generation according with the strategy adopted (*A1*, *A2*, *A3*). Elitism [6] was used at a rate of 20%, parent selection for the one-point crossover was made directly from the elite and mutation was applied after crossover at a rate of 2% per bit. The efficacy of the GA run was measured by testing the efficacy of the best rule found in the classification of 10^5 Gaussian distributed ICs. Initially, 100 GA runs was executed for each adaptive approach alone. After some exploratory runs to set the best value for efficacy bound (E_B), we adopted $E_B = 85\%$ for all the approaches. A basic experiment was also performed in which all the ICs sample are generated with a uniform distribution as in [8]. This experiment was used as a reference to estimate the actual contribution of each adaptive strategy to improve the evolutionary search. Table 1 presents the efficacy of the best rule found in each run categorized within specified interval. It also presents the average of the efficacy considering the best rules in each experiment (100 runs); the confidence interval (95%) associated to the best rule found; and the efficacy of the best rule found per experiment. It is possible to observe that in both sizes (20×20 and 21×21) the adaptive strategy that returned higher efficacy rules was *A1*: it could find rules with efficacy between 70% and 75% in more than 40% of the runs while *A3* was able to find about 1% of the rules in this range and *A2* did not find any rule in this range. As the work [18] joint applied *A2* and *A3* to obtain good rules, we also performed an experiment to evaluate this composition. However, for this composition we used $E_B = 60\%$. As one can see in Table 1, the joint application of both strategies had returned better values than their application alone in the evaluation process; however, their results are still significant worst than the application of *A1* alone. Comparing the adaptive experiments with the basic search, we can see that the unique actual improvement was obtained using *A1*. The usage of other adaptive strategies returned lower values than the simple adoption of uniform IC during individual evaluation.

Table 1. 2D DCT rules obtained using different evaluation approaches.

Efficacy	Lattices	Adaptive Strategies				
		Basic	***A1***	***A2***	***A3***	***A2,A3***
< 60	**20×20**	28	12	99	98	96
	21×21	18	15	98	97	94
60 ~ 65	**20×20**	1	1	0	0	0
	21×21	1	2	0	0	0
65 ~ 70	**20×20**	48	38	1	1	0
	21×21	62	41	1	2	3
70 ~ 75	**20×20**	23	49	0	1	4
	21×21	19	42	1	1	3
Average (%)	**20×20**	**66.47**	**67.93**	**51.35**	**51.61**	**52.06**
	21×21	**66.25**	**67.21**	**51.55**	**51.72**	**52.32**
C Int 95%	**20×20**	**±1.28**	**±1.24**	**±0.25**	**±0.49**	**±0.79**
	21×21	**±1.26**	**±1.34**	**±0.42**	**±0.54**	**±0.55**
Best Rule	**20×20**	**73.33**	**73.84**	**69.13**	**71.15**	**72.11**
	21×21	**72.15**	**72.86**	**71.33**	**71.19**	**71.90**

In previous work using 1D DCT [6], the usage of uniform ICs make the problem easier to solve and GA employed by them

could start their convergence in the first generation toward effective rules to DCT. By our results, we can conclude that strategies *A2* and *A3* alone cannot deal with this problem and they still generate difficult ICs in the initial generations and in almost all runs the search could not start the convergence to good rules. On the other hand, we know that the general idea of these strategies was successfully applied in [18]: the evolutionary environment described in such reference was able to find high effectives rules both in 1D and 2D DCT. Therefore, it seems that these adaptive strategies have received some kind of push in the first generations to be able to start GA convergence. A heuristic based on *BWLR-symmetry* parameter were used to guide GA search. In fact, the best 21×21 rule was obtained in [18] using this parameter and the best 20×20 rule was obtained in [12][9] also using this information, although they have been incorporated in different ways in both previous works.

Therefore, we incorporated *BWLR-symmetry* parameter in the evolutionary environment, in the same way described in [12][9], applying the same adaptive approaches analyzed previously. Each experiment was consisted of 100 GA runs and the same parameters used in experiments of Table 1 were employed, except for the usage of a weighted fitness function to compose the original fitness (the number of ICs classified correctly) and the value returned by *BWLR-symmetry* parameter. Their results are presented in Table 2.

Table 2. 2D DCT rules obtained using *BWLR-symmetry* and different evaluation approaches.

Efficacy	Lattices	Adaptive Strategies				
		Basic	***A1***	***A2***	***A3***	***A2,A3***
< 65	**20×20**	0	0	2	1	0
	21×21	0	0	4	0	2
65 ~ 70	**20×20**	15	2	22	6	7
	21×21	3	3	18	3	7
70 ~ 75	**20×20**	84	77	68	57	65
	21×21	74	78	72	59	65
75 ~ 80	**20×20**	1	21	8	36	28
	21×21	23	19	6	38	26
Average (%)	**20×20**	**71.55**	**73.61**	**71.58**	**73.98**	**73.76**
	21×21	**73.45**	**73.69**	**71.69**	**74.22**	**73.51**
C Int 95%	**20×20**	**± 0.33**	**± 0.37**	**± 0.41**	**± 0.32**	**± 0.40**
	21×21	**± 0.39**	**± 0.35**	**± 0.39**	**± 0.30**	**± 0.56**
Best Rule	**20×20**	**76.12**	**78.71**	**76.79**	**78.16**	**78.11**
	21×21	**78.27**	**78.18**	**77.10**	**79.70**	**78.44**

These experiments showed significant improvements when compared with the respective experiments in Table 1. Therefore, as pointed in previous works, *BWLR*-based heuristic indeed promotes a consistent improvement on evolutionary search for DCT rules. However, the most notable improvement was obtained by experiments that use *A3*. We believe that the strength guidance of *BWLR*-based heuristic toward regions of search space, where high efficacy rules are more likely to occur, aim adaptive strategy *A3* to start GA convergence. On the other hand, *A2* returned the worst results when applied jointly with *BWLR*-based heuristic, even compared with basic search.

The experiment using strategy *A1* found the best rule for 20×20 lattices while the experiment using strategy *A3* found the best one for 21×21 lattices. These efficacies are still below the best published rules in both sizes [12][9] and [18]. However, the GA parameters (T_p, N_g, and N_{IC}) used in our initial experiments are modest compared with these previous works. Finally, more robust GA parameters were applied trying to obtain better rules.

We performed a new series of runs using T_p = 500, N_g = 500 and N_{IC} = 500 and the best adaptive strategies with the *BWLR-symmetry* parameter: *A1* and *A3*. The best rules for 20×20 and 21×21 lattices were found respectively by approaches *A1* and *A3*. Their hexadecimal are:

000000010F0500770007171105251557041D0003254F0
051001F33DFD7FF1F7F041911071F0F1557261707F7
57BF7B7F9FBF1531175F7C77FF1F9FF7FF7F3F77

and

0304000101750010085504 0D075555FF0303000401050
7374F5F0C2D4557171F0B5715111357051043177E7F
7F575FFF4F07FF3F57177F354F7FFFFF7F37FFFF.

The global mean performance of the new best rules was tested applying both rules in different lattice sizes (from 6×6 to 21×21 lattices). The comparative performance of this rules are presented in Table 3. For all sizes, the rules were evaluated in samples of 100,000 Gaussian distributed lattices. An explanation is necessary about the tests using even size lattices: in such case initial configurations with equal number of 0's and 1's are possible. In this case, DCT is not well defined since it is not possible to decide the majority bit. However, in all other situations, DCT could be carried out. So, we just excluded the non-decidable lattices from our tests. A comparative analysis of our new rules with the previous published ones was performed: they were also tested in different lattice sizes and their results are showed in Table 3. The previous best rules available for even and odd lattices were published in works [12][9] and [18]. Their hexadecimal codes are respectively:

000400051D2F801F0005131B034D301504140042155F
103355172B7DD57F3F3D5043003D1F0FC577311D16
FF073F775F1605005F37BFFC7FFFDF777F4FFBF7F

and

0000000100110141 0B1514050D2B7757010113070305
4557009557D703957FF70B1B455505131345177F475F
077F7F77FF57475D1F7F6765BF3F3FBFFFFF7F77.

Looking over Table 3, except for the smaller lattice size (6×6), the new rule evolved in 20×20 - obtained using strategy *A1* - is better than the previous one evolved using 12×12 in [12][9]. Similarly, the new rule evolved in 21×21 using only approach *A3* overcome the previous one in [18] in almost all lattice sizes, except for 15×15 lattice size. Besides, the efficacy of the four rules oscillates from odd to even lattices, as pointed in [12][9]. Due to this oscillation, the even-size evolved rules have a better performance in all even lattices, while the odd-size evolved rules present a better performance in all odd lattice tested. However, odd-size

evolved rules presents a more severe oscillation than even-size evolved ones. Besides, the new rule found using 21×21 presents an oscillation minor than the previous published one.

Table 3. Efficacies obtained using 2D 512-bits CA rules in different lattice sizes.

Lattice Size	*Evolved in even-size lattices*		*Evolved in odd-size lattices*	
	12×12 **[12]**	*20×20 (A1)*	*21×21* **[18]**	*21×21 (A3)*
6×6	**87.177**	83.357	52.04	**64.33**
7×7	84.863	**85.965**	87.24	**87.58**
8×8	85.453	**85.97**	56.87	**65.84**
9×9	83.476	**84.873**	85.93	**86.46**
10×10	83.849	**85.081**	60.46	**68.49**
11×11	82.83	**83.731**	84.87	**85.24**
12×12	82.78	**83.898**	63.95	**70.97**
13×13	81.969	**83.02**	84.34	**84.65**
14×14	81.465	**83.004**	66.95	**73.53**
15×15	80.671	**82.432**	**84.27**	84.18
16×16	80.458	**82.137**	69.83	**76.06**
17×17	79.659	**81.797**	83.58	**83.67**
18×18	79.217	**81.449**	72.03	**77.70**
19×19	78.409	**81.262**	83.07	**83.14**
20×20	77.944	**80.981**	73.95	**78.74**
21×21	76.838	**80.732**	82.23	**82.89**
Average	**81.57%**	**82.95%**	**75.32%**	**78.16%**

6. CONCLUSIONS

Three different adaptive strategies were investigated in the present work regarding evolutionary search for DCT rules. They are related to the way in which lattice samples are generated to evaluate individuals in a GA-based approach. All of them start using easy lattices to classify and as effective rules are being obtained the difficult level is progressively increased toward the target evaluation. This evaluation consists on using the hardest instances to measure the efficacy of a CA rule when classifying them, that is, lattices with about 50% of 0s and 1s.

A1 strategy starts using a uniformly distributed sample, *A2* strategy starts using small size lattices and *A3* strategy starts using a Gaussian distributed sample with the discard of lattices with proportion of 1s between 43% and 57%. In all strategies, as efficient rules are being obtained, the lattices used during evaluation are changed to be closer to the target ones. They were confronted with a simple fixed strategy using uniformly distributed lattices during GA generations. Our results showed that when the adaptive strategies are applied alone, the only actual improvement was obtained using *A1*. The other two returned results worse than the fixed strategy. However, when the adaptive strategies were applied jointly with a parameter-based heuristic (which uses *BWLR_symmetry* parameter), *A3* strategy returns results as good as *A1*, outperforming the fixed evaluation. In general, *A1* returns the best rules when they are evolved using even-size lattices, while *A3* is more effective when odd-lattices are used. Using *A1* and *A3* in experiments with 20×20 and 21×21 lattices, we were able to find CA rules better than the best one published. However, a strength oscillation was observed when applying the 21×21evolved rule over different even and odd size lattices. This behavior was also observed in [12] when analyzing previous 21×21 rules, which have been evolved using jointly *A3* and *A2*. Thus, an open question is to investigate if the discard of instances with distribution very close to 50% as performed when using *A3* has any relation to the weak performance of such rules in even size lattices. We are conducting new experiments to explore such issue. Finally, comparing the efficacies of the new best 20×20 and 21×21 evolved rules regarding DCT problem, we claim that although the second one surpass the first one in all odd-size evaluated (from 7×7 to 21×21), the 20×20 evolved rule is clearly better to solve the problem. When trying to understand how CA can be used as a new computational paradigm, a rule to solve a specific task should be as general as possible and not so dependent on any CA parameter, as the parity of the lattice size. Therefore, the 20×20 rule is much more equilibrated than the other since it only presents a slightly oscillation in the smaller sizes. Such better performance can also be highlighted by the confidence intervals obtained in the experiments reported in Table 3: 82.95% ± 0,84 (20×20 evolved rule) and 78.16% ± 3,52 (21×21 evolved rule).

Although Tables 1 and 2 present only performances obtained when rules are applied to the same lattice size for which they were evolved, we can say that in general the rules obtained using *A1* strategy are more equilibrated when applied to different lattice size even when they are obtained using odd size. That is, CA rules obtained using *A1* strategy and odd-sized lattices return smaller oscillations than those obtained using *A3* strategy. Therefore, we can say that *A1* strategy shows some competitive advantage and can be elected as the best adaptive approach tested here. Strategy *A3* seems to be adequate only to odd sized lattices and it give some particular characteristic to those evolved rules to present a more expressive oscillation.

Another conclusion of this work is related to the usage of strategies alone or jointly used with parameter-based heuristic. It was possible to observe that when using individually (without parameter-based heuristic), two strategies (*A2* and *A3*) seem to be very severe for adaptive fitness. However, when the parameter-based heuristic was applied jointly with these strategies, they dramatically change their performance, returning some of the best rules, especially in the case of *A1*. It shows that in the presence of a parameter-based heuristic – which gives a general driven to promissory regions of rule space - all different fitness evaluation strategies were benefited. Thus, although parameter-based heuristic are not enough to find good rules for computational tasks alone as results in [16] suggests, they are good global drivers for other more specific local strategies to be applied in the evolutionary search for CA rules.

7. ACKNOWLEDGMENTS

GMBO thanks CNPq and FAPEMIG support.

8. REFERENCES

[1] Andre, D., Bennett III, F. and Koza, J. 1996. Discovery by Genetic Programming of a Cellular Automata Rule that is Better than any Known Rule for the Majority Classification Problem. In: *Genetic Programming Conference*. Stanford.

[2] Capcarrère, M., Sipper, M. and Tomassini, M. 1996. Two-state, r=1, cellular automata that classifies density. Physical Review Letters. vol. 77(24), pp. 4969-4971.

[3] de Oliveira, P.P.B., Bortot and Oliveira, G.M.B. 2006. The best currently known cellular automata rules for density classification and the evolutionary mechanisms that led to them. Neurocomputing. vol. 70, pp. 35-43, 2006.

[4] Juillé, H. and Pollack, J. 1998. Coevolving the "Ideal" Trainer: Application to the Discovery of Cellular Automata Rules. In: Genetic Programming Conference, Madison.

[5] Land, M. and Belew, R. 1995. No Perfect Two-State Cellular Automata for Density Classification Exists. Physical Review Letters. vol. 74(25), pp. 5148.

[6] Mitchell, M., Hraber, P. and Crutchfield, J. 1993. Revisiting the Edge of Chaos: Evolving Cellular Automata to Perform Computations. Complex Systems. vol. 7, pp. 89-130.

[7] Mitchell, M. 1996. Computation in Cellular Automata: A Selected Review. In: Nonstandard Computation. Weinheim: VCH Verlagsgesellschaft.

[8] Morales, F., Crutchfield, J. and Mitchell, M. 2000. Evolving two-dimensional cellular automata to perform density classification. Parallel Computing. vol. 27, pp. 571-585.

[9] Oliveira, G.M.B., de Oliveira, P.P.B. and Omar, N. 2000. Evolving solutions of the density classification task in 1D cellular automata, guided by parameters that estimate their dynamic behavior, In: Artificial Life VII. pp. 428-436.

[10] Oliveira, G.M.B., de Oliveira, P.P.B. and Omar, N. 2001. Definition and applications of a five-parameter characterization of one-dimensional cellular automata rule space. In: Artificial Life (MIT Press). vol. 7(3), pp. 277-301.

[11] Oliveira, G.M.B. and Siqueira, S.R.C. 2006. Parameter Characterization of Two-Dimensional Cellular Automata Rule Space. Physica D. vol. 217(1), pp. 1-6.

[12] Oliveira, G.M.B, Martins, L.G.A., Fynn, E. and de Carvalho, L.B. 2009. Some Investigations About Synchronization and Density Classification Tasks in One-dimensional and Two-dimensional Cellular Automata Rule Spaces. In: 15th International Workshop on Cellular Automata and Discrete Complex Systems. ENTCS, vol. 252, pp. 121-142.

[13] Packard, N. 1988. Adaptation toward the Edge of Chaos. Dynamic Patterns in Complex Systems. pp. 293-301.

[14] Reynaga, R. and Amthauer, E. 2003. Two-dimensional cellular automata of radius one for density classification task $\rho = ½$. Pattern Recognition Letters. Elsevier. vol. 24 (15), pp. 2849-2856.

[15] Swiecicka, A. and Seredynski, F. 2006. Multiprocessor schedulingand rescheduling with use of cellular automata and artificial immune system support. IEEE Transactions on Parallel Distributed Systems,17(3):253–262.

[16] Weinert, W. and Lopes, H. 2010. Evaluation of dynamic behavior forecasting parameters in the process of transition rule induction of unidimensional cellular automata. Biosystems. vol. 99 (1), pp. 6-16.

[17] Wolfram, S. 2002. A New Kind of Science, Wolfram Media.

[18] Wolz, D. and de Oliveira, P.P.B. 2008. Very effective evolutionary techniques for searching cellular automata rule spaces. Journal of Cellular Automata. vol. 3, pp. 289-312.