STUDIA UNIV. BABEŞ–BOLYAI, INFORMATICA, Volume **LVI**, Number 1, 2011

EVOLUTIONARY COMPUTING IN THE STUDY OF COMPLEX SYSTEMS

DAVID ICLĂNZAN[(1)], RODICA IOANA LUNG[(2)], ANCA GOG[(1)], AND CAMELIA CHIRA[(1)]

ABSTRACT. Complex systems and their important principles of emergence, auto-organization and adaptability are being intensively studied by researchers in a variety of fields including physics, biology, computer science, sociology and economics. The aim of this paper is to highlight potential interesting research directions lying at the intersection of nature-inspired computing and complex systems and able to generate new insights on modelling complexity. The field of evolutionary computation comprises classes of nature inspired search, design and optimization methods that can be applied to a variety of complex problems. Complex systems and relevant evolutionary computation methods are reviewed and analysed in this paper. Several aspects relating evolutionary computation to emergent and self-organization phenomena are emphasized.

1. Introduction

A complex system is any system containing a large number of interacting entities (agents, processes, etc.) which are interdependent. The system behaviour cannot be identified by considering each individual entity and combining them, but considering how the relationships between entities affect the behaviour of the whole system. The main features of complex systems include emergence, self-organization, evolution and adaptability. Emergence occurs when the behaviour of a system cannot be reduced to the sum of the behaviour of the parts. Self-organization is the process by which elements interact to create spatio-temporal patterns of behaviour that are not directly imposed by external forces. The formation of complex systems, and the structural/functional change of such systems, is a process of adaptation. Evolution

Received by the editors: March 22, 2011.

2010 *Mathematics Subject Classification.* 68-02.

1998 *CR Categories and Descriptors.* A.1 General Literature [**INTRODUCTORY AND SURVEY**]; I.2.8 Computing Methodologies [**ARTIFICIAL INTELLIGENCE**]:Problem Solving, Control Methods, and Search – *Heuristic methods*.

Key words and phrases. evolutionary computation, complex systems, emergence, self-organization, cellular automata.

is the adaptation of populations through inter-generation changes in the composition of the population. Learning is a similar process of adaptation of a system through changes in its internal patterns. An extensive study of complex systems and cellular automata as important tools in the analysis of complex interactions and emergent systems has been presented in [17].

The development of complex organizations, structures and mechanisms found in nature are explained by Charles Darwin's theory of evolution [19], one of the greatest scientific achievements of all times. This powerful evolutionary paradigm stands behind a class of nature inspired optimization methods, called Evolutionary Computation (EC). Some optimization methods belonging to the EC class share more than the basic principles and operators [39]. Certain high-level phenomena like the Baldwin effect [38, 65, 73, 71, 5, 28], coevolution and arm races [64, 61, 11, 72, 31], parasitism [56, 63], exaptation [24, 33, 35] and speciation [60] are found in methods and models belonging to EC.

Daniel Dennett has underlined that, "evolution will occur whenever and wherever three conditions are met: replication, variation (mutation), and differential fitness (competition)" [26], thus the digital transference of the natural selection paradigm must be also capable of producing self-organization and adaptability. Several such examples are reviewed in this paper based on a broad analysis of complex systems, EC major areas of interest and important aspects that connect emergent self-organizing phenomena with evolutionary computation. The main emerging research areas are discussed and potential interesting directions able to generate new insights on modelling complexity are emphasized.

The paper is organized as follows: Section 2 presents evolutionary techniques as tools for obtaining complex behaviours; Section 3 describes how EC can be used to evolve irreducible complexity; Section 4 presents the significant role that EC plays in many aspects of the Artificial Life domain; Section 5 shows that evolutionary techniques can be applied for automating the design of heuristic search methods and Section 6 contains conclusions and further research directions.

2. Evolving complex behaviours

Inspired by the richness and robustness of behavioural complexity exhibited by living organisms, EC has been used to tackle a broad variety of problems regarding the development of desired behaviours or strategies. Indeed, truly emergent phenomena are those that cannot be controlled or foreseen. Their effect - beneficial or destructive - will only appear during or after the interaction of the components takes place. Therefore, the task of designing a system presenting desired emergent behaviour or the task of avoiding undesired emergent behaviour becomes complex and challenging [9]. Since it is by definition impossible to know how design choices made on the component level affect the overall system behaviour such tasks may be realized only by means of computational extensive and expensive simulations. This kind of simulations can be approached by evolutionary computation tools.

There are several characteristics of Evolutionary Algorithms (EAs) that make them suitable for dealing with the challenges mentioned above:

- EAs are black box optimization heuristics, they do not impose any restrictions on the fitness function and thus can be used in a complex simulation environment;
- EAs can be efficiently parallelized, even on heterogeneous hardware platforms like computer grids [1];
- Approximation methods for fitness evaluation can significantly reduce the computational complexity of the simulation;
- EAs are known to be able to cope well with uncertainty in evaluation and capable to adapt to changing environments [7];
- The well established field of Evolutionary Multiobjective Optimization provides a range of methods and tools for dealing with multiple objectives which is a realistic approach in studying complex systems and possible emergent behaviour [23];
- EAs are highly adaptable to different solution concepts - using generative relations such as Pareto dominance or Nash ascendancy can lead to different types of solutions - the Pareto frontier or the Nash equilibria of a game respectively [29];
- EAs are known to be adaptive but they can be also interactive - both features making them useful in studying or designing possible emergent behaviour.

EAs have been successfully used in designing complex systems and inducing desired emergent behaviour. One example is the problem of designing en-route caching strategies [8] where genetic programming is engaged to design effective caching strategies. An EA was used with traffic simulation to design a traffic light controller in a multi-objective setting, attempting to minimize travel time as well as the number of stops.

Chellapilla and Fogel [16] used a genetic algorithm (GA) to evolve neural networks that could play the game of checkers. The major breakthrough of the paper is represented by the fact that a competitive strategy could be evolved given only the spatial positions of pieces on the checkerboard and the piece differential. The GA optimized artificial neural networks to evaluate alternative positions in the game without relying on any specific credit assignment algorithm - a rewarding mechanism that would normally require human expertise.

Evolutionary search has been applied to develop strategies for many different games like Othello [52] or GO [62]. However, these results suggest that the evolutionary principles may be successfully applied also to problems that have not yet been solved by human expertise.

Evolutionary methods were also successfully applied by Andre and Teller to develop a program for controlling a team of robot soccer players [4]. They used a genetic programming algorithm which operated with a set of primitive control functions such as turning, moving and kicking. The fitness function rewarded good play in general, rather than scoring specific tasks. No code or elementary building block was provided to teach the team how to achieve complex objectives, like ball tracing, kicking the ball in the correct direction, keeping the ball on the opponent's side, goal scoring etc. The robot team, called Darwin United, entered the international RoboCup[1] tournament, an annual soccer tournament between teams of autonomous robots. Darwin United performed quite well, outranking half of the human-written, highly specialized entries.

Cellular Automata (CA) are decentralized structures of simple and locally interacting elements (cells) that evolve following a set of rules [74]. Programming CA is not an easy task, especially when the desired computation requires global coordination. CA provides an idealized environment for studying how (simulated) evolution can develop systems characterized by "emergent computation" where a global, coordinated behaviour results from the local interaction of simple components [49].

The most widely studied CA task is the density classification problem (DCT) [48]. The task refers to finding the density most present in the initial cellular state. Packard [57] made the first attempts to use genetic algorithms for finding CA rules for the density task. Genetic programming [41], coevolutionary learning [40] and gene expression programming [30] have also been engaged for this problem. Genetic algorithms for computational emergence in the density classification task have been extensively investigated by Mitchell et al [20, 49, 50, 58]. The human designed Gacs-Kurdyumov-Levin rule with a performance of 81.6% as all other known subsequent human-written rules for this problem, were surpassed by rules developed by simulated evolution. Andre et al. [3] found a rule performing with an 82.23% accuracy by using genetic programming. The best currently reported DCT rule has a performance of 89% [54], which was evolved by a two-tier evolutionary algorithm.

[1] `http://www.robocup.org/`

The potential of evolutionary models to efficiently approach the problem of detecting CA rules to facilitate a certain global behaviour is also confirmed by other current research results [46, 77, 32, 25].

Another CA computational task intensely studied is the synchronization task (ST) [21], where the goal is to find 1D binary CA able to reach a configuration that cycles from all cells 0 in one time step to all cells 1 in next time step (starting from an arbitrary initial configuration). Evolutionary models have been successfully engaged for the synchronization task in several studies [21, 46]. There are several one-dimensional radius-3 CA rules able to solve ST for any arbitrary lattice configuration [21] with an efficacity of approximately 95%. Genetic algorithms proposed in [53, 46] for the synchronization task are able to find radius-2 rules with high efficacy.

Research has also been focused on evolving behaviours in multidimensional CA. Morales et al. [51], Alonso and Bull [2] have studied the DCT in a two dimensional setting. In [10] the authors use GA to evolve behaviour in multidimensional CA for DCT, the checkerboard problem that requires the formation of an alternating simple pattern and finally for generic bitmap evolution. The authors found that symmetrical bitmaps seem to be easier to generate than asymmetric ones and that multidimensional CA can solve certain problems faster than one dimensional CA. Chavoya and Duthen successfully evolve CA to produce predefined 2D and 3D shapes [14]. In a later work they apply a GA to evolve an extended artificial regulatory network to produce predefined 2D cell patterns [15].

The evolutionary discovery of rules that produce global synchronization is significant, since these exemplify the automated development of sophisticated emergent computation in decentralized, distributed systems such as CA. These discoveries are encouraging for the prospect of using EC to automatically evolve behaviour for more complex tasks, like predicting chaotic sequences.

3. Evolving irreducible complexity

An important challenge to evolutionary theory is to explain the origin and development of complex organismal features. Michael Behe, the originator of the term irreducible complexity (IC), defines an IC system as one "composed of several well-matched, interacting parts that contribute to the basic function, wherein the removal of any one of the parts causes the system to effectively cease functioning" [6]. IC is a central argument for proponents of intelligent design, revolving around the belief that such systems demonstrate that modern biological forms could not have evolved naturally.

Evolutionary biologists have shown that through evolutionary mechanism like deletion or addition or multiple parts, change of function or addition of second function to a part, gradual modification and loss of previously existing scaffolding can contribute to the production of IC systems.

In [34] the authors use a simple GA with variable-length chromosomes over a dynamic fitness function, in order to demonstrate that GAs can produce systems composed of multiple parts contributing to a specified complex function, where all components are critical. Therefore, such a design is irreducible.

The problem to be solved by the GA is a game defined over a 30×14 board, where the goal is to attain fitness scores greater than zero. As better solutions are evolved, the game responds by becoming more and more difficult.

By default, each grid cell in the game board is blank. The genes of individuals code mappings onto the game board; the encoded game cells, called "boxes" have one of four types, designated by the letters *S,A,L,R*. As the game begins, a virtual ball falls through the board, entering at the column index 5, carrying an initial point value of 15. The four box types can modify this value and the direction of travel through the board and can also duplicate the ball. The goal of the game is to multiply game balls and steer them to the sink column, with index 8. Game balls that exit the board on different columns do not affect the fitness score.

The effect of the four box types is the following:

(1) The *Split box* (S-box) duplicates the game ball, each with a point value one less than the original and with a different exit path from the box.
(2) The *Add box* (A-box) increases the point value of a game ball passing through it.
(3) The *Left boxes* (L-boxes) and *Right boxes* (R-boxes) modify the ball's direction of travel (right and left turn).

The fitness score of an individual x is defined as:

$$f(x) = max(0, p(x) - l(x) \cdot P) \tag{1}$$

where $p(x)$ is the number of collected points, $l(x)$ is the chromosome length, and P is a population-wide penalty value which increases with time to make the game more difficult as the individuals evolve. Individuals with fitness zero are considered non-viable, and they do not participate in the tournament selection employed by the GA.

Graham et al. [34] conducted experiments to verify if a GA can construct irreducible complex solutions to this game. An individual was regarded irreducible complex if its fitness was greater than zero, it contained more then five boxes (simple solutions were not of interest) and the removal of any single part (box) from the phenotype resulted in the fitness of the individual dropping to zero. With small population sizes of 50 individuals, they were able to constantly evolve irreducible solutions to this game. Such a highly-evolved solution is depicted in Fig. 1.

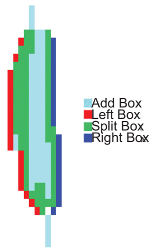

FIGURE 1. An example of a highly-evolved individual as reported in [34]. It can sustain a penalty of $P \cong 1.1 \times 10^{12}$ and collects approximately 3.6×10^{13} points.

The game solution strategies found by the GA are variations of the general pattern observable in Fig. 1. The source and sink column is connected with a cluster of A and S-boxes that multiply the game balls and increase their point value. Each side of the cluster if braced by L and R-boxes, that route the balls back into the cluster and towards the sink column.

The work[2] has demonstrated with the help of a simulated environment that simple evolutionary mechanisms can produce irreducible complexity, consisting of more than 100 parts.

In [18], Clayton depicts how IC systems can be obtained in a simple system that operates on a regular two-dimensional triangular lattice. The nodes in the considered lattice are binary. Whenever two adjacent nodes are set to the value 1 or ON, an edge automatically connects them. A group of connected nodes forms a system which is considered viable if and only if it forms a closed geometric shape and its fitness is directly proportional with its perimeter (larger systems are preferred). Through a simulated evolution, the author demonstrates that irreducible complexity evolves in this model in response to natural selection, favoring the larger systems with fewer parts. A continual increase in complexity was observed in 100 evolutionary steps, resulting in IC systems containing between 6 and 30 parts.

GAs are also used to address the watchmaker analogy, which affirms that the highly complex inner workings of a system (ex. watch) necessitate an intelligent designer. The experiment[3] demonstrates that if the components of a watch are allowed to be combined and the resulting design undergoes natural selection, a functional watch can be evolved. The clocks evolve through a series of transitional forms, with ever increasing complexity. At initialization,

[2]An online demonstration of the game and algorithm can be found at `http://www.stellaralchemy.com/ice/index.php`.

[3]`http://richarddawkins.net/videos/1322-evolution-is-a-blind-watchmaker`

98% of the designs are non functional, the remaining designs being simple pendulums. As they evolve, proto-clocks, more sophisticated clocks gradually develop, starting from 1 hand and resulting in up to 4 hands. The most sophisticated clock obtained had 21 interacting parts.

4. Evolving digital organisms and ecosystems

Natural evolution as well as its simulated variant can produce complex system configurations and behaviours. Therefore, EC plays a significant role in many aspects of the Artificial Life (AL) domain as behaviour strategies, methods of communication, swarm intelligence and many other topics are commonly explored using evolutionary search techniques (see the "From Animals to Animats" or "Artificial Life" conference series). Furthermore, the dynamics of Darwinian evolution and different hypotheses and models of evolution are often studied through digital organisms - artificial life-forms, that are defined as self-replicating digital models that mutate, compete and evolve.

One of the first experiments with digital organisms was conducted by ecologist Thomas S. Ray in the Tierra model [69], where computer programs capable of mutation and self-replication compete for central processing unit time (energy) and access to main memory (resources). The model has been used to conduct experiments regarding evolutionary and ecological dynamics, including dynamics of punctuated equilibrium, host-parasite coevolution and density-dependent natural selection. In the Tierra framework the fitness function is endogenous; there is simply survival or death of a digital organism.

A related framework, AVIDA, where each digital organism lives in its own protected region of memory and is executed by its dedicated virtual CPU, was used to conduct research in the digital evolution of complex features [43]. The digital organisms evolve to perform certain computational tasks, from which the most complicated one is the equality operator - requiring at least 19 simpler, precisely ordered instructions.

Other noteworthy examples of digital organism simulators include:

(1) *Evolve 4.0*[4] a 2D cellular automata where each cell can behave independently as unicellular organism or be a part of a multicellular creature. The digital organisms can grow, movie, feed and replicate.
(2) *Darwinbots* [5], a digital environment of interacting and fighting bots, where the behaviour of the bots is specified by their genome.
(3) *breve*[6], a 3D simulator for multi-agent systems and artificial life, with support for physical simulation and collision detection.

[4] `http://stauffercom.com/evolve4/`
[5] `http://www.darwinbots.com/WikiManual/index.php?title=Main_Page`
[6] `http://www.spiderland.org/breve/`

(4) *Polyworld*[7], an ecosystem of agents which search for food, mate, replicate and hunt. The individuals actions are governed by arbitrary architecture neural networks employing Hebbian learning. The neural network is encoded in each individual's genome that is mutated and replicated. Recent results [75] experimenting with Polyworld had underlined an association between small-world network structures of the controlling networks and complex neural dynamics.
(5) *AnimatLab*[8], is a recently developed simulation tool combining biomechanical simulation and biologically realistic (spiking) neural networks.

Lohn's CA, evolved by a GA to be capable of self-replication [45] and the plant-like biomorphs introduced by Dawkins [22] are another examples of evolutionary AL-forms.

Sims [67, 66] demonstrates the development of animal-like morphologies by simulating Darwinian evolutions of virtual block creatures. The fitness of the initially randomly generated block creatures is measured in their ability to perform a given task, for example swimming in a simulated water environment. The creatures undergoing natural selection and variation developed successful behaviours for swimming, walking, jumping, following, and competing for control of a (resource) cube.

5. Self-* search

Emergent phenomena observed in natural systems have been used as an inspiration for designing many evolutionary computation models. For example Ant colony optimization (ACO) [27] or Particle Swarm Optimization (PSO) [55] methods mimic emergent features mentioned above to solve complex search and optimization problems. Emergent phenomena are carefully observed and used as an inspiration for designing new efficient techniques.

Nevertheless, as evolutionary search is capable of producing highly coadapted complex systems that are often irreducible, there is a growing research interest in evolutionary techniques for automating the (self) design of heuristic search methods. Successful approaches alleviate the need for human experts in the process of designing efficient problem dependent optimization methods (heuristics).

There are two basic approaches to turn simpler methods into self-* algorithms or hyperheuristics: one is built upon machine learning techniques to identify good parameter settings, proper operators and algorithmic building blocks; the second one uses a meta-level search over the parameterization of the base method, where the selection of the good features can be decided in a

[7] `http://beanblossom.in.us/larryy/Polyworld.html`
[8] `http://www.animatlab.com/`

fast greedy way, susceptible of finding (weak) local-optima or by a computationally more expensive evolutionary means.

Regardless of the machine-learning or meta-search based approach, related self-adaptive and meta-level methods revolve around three general processes in automated heuristic design:

(1) Adjusting or tuning the method's control parameters, an approach exemplified by adaptive self-tuning Evolution Strategies [36] or automatically selected perturbation step size in Iterated Local Search [68].
(2) Dynamic selection of existing algorithmic components, ex. managing the search operators in an EC algorithm [47] or the application of various linkage learning techniques for developing competent crossover operators [37, 59, 76, 44].
(3) Generating new heuristics from basic sub-components, an approach implemented by the "Teacher"[9] framework [70].

The literature regarding this field is immense and it can not be covered in this review. For a more in depth discussion, we forward the interested reader to recent reviews on this subject [42, 13, 12].

However, we would like to point out that the evolutionary paradigm can be recursively applied to enhance EC methods. Self-adaptation is an implicit parameter adaptation technique enabling the evolutionary search to tune the strategy parameters automatically by evolution [42].

6. Conclusions

Evolutionary computation techniques have been successfully applied for problems that arise from the study of complex systems principles of emergence, auto-organization and adaptability.

A two way relationship between the two domains can be observed. On the one hand, their interaction gave rise to new efficient optimization techniques inspired by emergent phenomena and helped improving different heuristic methods in terms of tuning control parameters or dynamic selection of components. On the other hand, evolutionary techniques have been used for designing complex systems. For example, complex desired behaviours and strategies have been evolved by means of evolutionary techniques. Cellular automata is a great example of global, coordinated behaviour that results from the local interaction of simple components. Several other examples include caching strategies, traffic controllers, strategies for difficult games etc. Another major application of EC is the production of irreducibly complex systems characterized by the fact that removing any of the systems parts causes the system to cease functioning. We also present the role that EC has in

[9]An acronym for TEchniques for the Automated Creation of HEuRistics

the Artificial Life domain, for aspects like behaviour strategies, methods of communication, swarm intelligence and many other topics.

There are many research perspectives at the intersection of nature-inspired computing and complex systems worth to be further explored. We emphasize the potential of computer simulations using multi-agent modelling and evolutionary computing techniques and investiging complex network and cellular automata models for the analysis of complex systems. Different interaction models at the micro/macro (inidividual/population) level that induce emergent behavior can be studied using evolutionary computation and further explored in modelling complex systems.

7. Acknowledgments

This research is supported by Grant PN II TE 320, Emergence, auto-organization and evolution: New computational models in the study of complex systems, funded by CNCSIS, Romania.

References

[1] E. Alba, C. Cotta, *Parallelism and evolutionary algorithms*, IEEE Transactions on Evolutionary Computation, 6 (2002), pp. 443-462.

[2] R. Alonso-Sanz, L. Bull, *A very effective density classifier two-dimensional cellular automaton with memory.* Journal of Physics A: Mathematical and Theoretical, 42:485101 (2009).

[3] D. Andre, F.H. Bennett, J.R. Koza, *Discovery by genetic programming of a cellular automata rule that is better than any known rule for the majority classification problem*, In Proceedings of the First Annual Conference on Genetic Programming, GECCO '96, Cambridge, MA, USA (1996), pp. 3-11.

[4] D. Andre, A. Teller, *Evolving team darwin united*, In Minoru Asada and Hiroaki Kitano, editors, RoboCup-98: Robot Soccer World Cup II, Springer Verlag (1999), pp. 346-351.

[5] J. Bala, K. De Jong, J. Huang, H. Vafaie, H. Wechsler, *Using learning to facilitate the evolution of features for recognizing visual concepts.*, Evolutionary Computation, 4 (1996), pp. 297-311.

[6] M.J. Behe, *Darwin's black box: the biochemical challenge to evolution*, Free Press New York (1996).

[7] J. Branke, *Evolutionary Optimization in Dynamic Environments*, Kluwer Academic Publishers, Norwell, MA, USA (2001).

[8] J. Branke, P. Funes, F. Thiele, *Evolving en-route caching strategies for the internet*, In Genetic and Evolutionary Computation GECCO 2004, LNCS Springer Berlin / Heidelberg, vol. 3103 (2004), pp. 434-446.

[9] J. Branke, H. Schmeck, *Evolutionary design of emergent behaviour*, In Organic Computing, volume 21 of Understanding Complex Systems, Springer Berlin / Heidelberg (2008), pp. 123-140.

[10] R. Breukelaar, Th. Bäck, *Using a genetic algorithm to evolve behaviour in multi dimensional cellular automata: emergence of behaviour*, In Proceedings of the 2005 conference on Genetic and evolutionary computation, GECCO '05, New York, NY, USA (2005), pp. 107-114.

[11] A. Bucci, J.B. Pollack, *On identifying global optima in cooperative coevolution*, In Proceedings of the 2005 conference on Genetic and evolutionary computation, GECCO '05, New York, NY, USA (2005), pp. 539-544.
[12] E.K. Burke, M. Hyde, G. Kendall, G. Ochoa, E. Ozcan, R. Qu, *A survey of hyper-heuristics*, Technical Report NOTTCS-TR-SUB-0906241418-2747, School of Computer Science and Information Technology, University of Nottingham (2010).
[13] K. Chakhlevitch and P. Cowling, *Hyperheuristics: recent developments*, Adaptive and Multilevel Metaheuristics (2008), pp. 3-29.
[14] A. Chavoya, Y. Duthen, *Using a genetic algorithm to evolve cellular automata for 2d/3d computational development*, In Proceedings of the 8th annual conference on Genetic and evolutionary computation, GECCO '06, New York, NY, USA (2006), pp. 231-232.
[15] A. Chavoya, Y. Duthen, *Use of a genetic algorithm to evolve an extended artificial regulatory network for cell pattern generation*, In Proceedings of the 9th annual conference on Genetic and evolutionary computation, GECCO '07, New York, NY, USA (2007), pp. 1062-1062.
[16] K. Chellapilla, D.B. Fogel, *Evolving an expert checkers playing program without using human expertise*, IEEE Transactions on Evolutionary Computation, 5:4 (2001), pp. 422-428.
[17] C. Chira, A. Gog, R. Lung, D. Iclanzan, *Complex Systems and Cellular Automata Models in the Study of Complexity*, Studia Informatica series, Vol. LV, No. 4 (2010), pp. 33-49.
[18] S.S. Clayton, *A simple model for the evolution of irreducible complexity*, Citeseer DOI 10.1.1.60.6797 (2006).
[19] C. Darwin, *On the Origin of Species*, John Murray, London, 1859.
[20] R. Das, M. Mitchell, J. P. Crutchfield, *A genetic algorithm discovers particle-based computation in cellular automata*, Parallel Problem Solving from Nature Conference (PPSN-III). Springer-Verlag (1994), pp. 344-353.
[21] R. Das, J.P. Crutchfield, M. Mitchell, J.E. Hanson, *Evolving globally synchronized cellular automata*, In Proceedings of the 6th International Conference on Genetic Algorithms, San Francisco, CA, USA, Morgan Kaufmann Publishers Inc (1995), pp. 336-343.
[22] R. Dawkins, *The Blind Watchmaker*, Penguin Books, 1986.
[23] K. Deb, D. Kalyanmoy, *Multi-Objective Optimization Using Evolutionary Algorithms*, John Wiley & Sons, Inc., New York, NY, USA, (2001).
[24] P. de Oliveira, *Simulation of exaptive behaviour*, In Yuval Davidor, Hans-Paul Schwefel, and Reinhard Mnner, editors, Parallel Problem Solving from Nature - PPSN III, LNCS vol. 866, Springer Berlin / Heidelberg (1994), pp. 354-364.
[25] P.P.B. de Oliveira, J.C. Bortot, G. Oliveira, *The best currently known class of dynamically equivalent cellular automata rules for density classification*, Neurocomputing, 70:1-3 (2006), pp. 35-43.
[26] DC Dennett, *The new replicators*, The encyclopedia of evolution, 1 (2002), pp. 83-92.
[27] M. Dorigo, T. Stutzle, *Ant Colony Optimization*, Bradford Company, Scituate, MA, USA (2004).
[28] K.L. Downing, *The baldwin effect in developing neural networks*, In Proceedings of the 12th annual conference on Genetic and evolutionary computation, GECCO '10, New York, NY, USA (2010), pp. 555-562.
[29] D. Dumitrescu, R.I. Lung, T.D. Mihoc, *Generative relations for evolutionary equilibria detection*, In Proceedings of the 11th Annual conference on Genetic and evolutionary computation, GECCO '09, New York, NY, USA (2009), pp. 1507-1512.

[30] C. Ferreira, *Gene Expression Programming: A New Adaptive Algorithm for Solving Problems*, Complex Systems, 13:2 (2001), pp. 87-129.

[31] S. Ficici, J.B. Pollack, *Challenges in coevolutionary learning: Arms-race dynamics, open-endedness, and mediocre stable states*, In Proceedings of the Sixth International Conference on Artificial Life, MIT Press (1998), pp. 238-247.

[32] A. Gog, C. Chira, *Cellular Automata Rule Detection Using Circular Asynchronous Evolutionary Search*, HAIS 2009, LNCS 5572 (2009), pp. 261-268.

[33] L. Graham and F. Oppacher, *Speciation through exaptation.* In Artificial Life, 2007. ALIFE'07. IEEE Symposium on, IEEE (2007), pp. 433-439.

[34] L. Graham, F. Oppacher, and S. Christensen, *Irreducible complexity in a genetic algorithm*, In IEEE Congress on Evolutionary Computation, IEEE (2007), pp. 3692-3697.

[35] L. Graham, *Exaptation and functional shift in evolutionary computing*, PhD thesis, Carleton University, Ottawa, Ont., Canada, Canada (2009).

[36] N. Hansen and A. Ostermeier, *Completely derandomized self-adaptation in evolution strategies.* Evolutionary computation, 9:2 (2001), pp. 159-195.

[37] G.R. Harik, D.E. Goldberg, *Learning linkage*, In Richard K. Belew and Michael D. Vose, editors, FOGA, Morgan Kaufmann (1996), pp. 247-262.

[38] G.E. Hinton, S.J. Nowlan, *How learning can guide evolution*, Adaptive individuals in evolving populations, Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 0-201-48369-6 (1996), pp. 447-454.

[39] J.H. Holland, *Adaptation in natural artificial systems*, University of Michigan Press, Ann Arbor (1975).

[40] H. Juille, J.B. Pollack, *Coevolving the ideal trainer: Application to the discovery of cellular automata rules*, In John R. Koza, Wolfgang Banzhaf, Kumar Chellapilla, Kalyanmoy Deb, Marco Dorigo, David B. Fogel, Max H. Garzon, David E. Goldberg, Hitoshi Iba, and Rick Riolo, editors, Genetic Programming 1998: Proceedings of the Third Annual Conference, University of Wisconsin, Madison, Wisconsin, USA, Morgan Kaufmann (1998), pp. 519-527.

[41] J. R. Koza, *Genetic Programming: On the Programming of Computers by Means of Natural Selection*, MIT Press (1992).

[42] O. Kramer, *Self-adaptive heuristics for evolutionary computation*, Springer Verlag, ISBN: 3540692800 (2008).

[43] R.E. Lenski, C. Ofria, R.T. Pennock, C. Adami, *The evolutionary origin of complex features*, Nature, 423:6936 (2003), pp. 139-144.

[44] Z. Li, E.D. Goodman, *Learning building block structure from crossover failure*, In GECCO '07: Proceedings of the 9th annual conference on Genetic and evolutionary computation, New York, NY, USA (2007), pp. 1280-1287.

[45] J.D. Lohn, J.A. Reggia, *Discovery of self-replicating structures using a genetic algorithm.* In IEEE International Conference on Evolutionary Computation, vol. 2, IEEE (1995), pp. 678-683.

[46] A. S. Mariano, G. M. B. de Oliveira, *Evolving one-dimensional radius-2 cellular automata rules for the synchronization task*, AUTOMATA-2008 Theory and Applications of Cellular Automata, Luniver Press (2008), pp. 514-526.

[47] J. Maturana, F. Lardeux, F. Saubion, *Autonomous operator management for evolutionary algorithms*, Journal of Heuristics (2010), pp. 1-29.

[48] M. Mitchell, P.T. Hraber, J.P. Crutchfield, *Revisiting the edge of chaos: Evolving cellular automata to perform computations*, Complex Systems, 7 (1993), pp. 89-130.

[49] M. Mitchell, J.P. Crutchfield, R. Das, *Evolving cellular automata with genetic algorithms: A review of recent work*, In Proceedings of the First International Conference on Evolutionary Computation and Its Applications (EvCA'96). Russian Academy of Sciences (1996).

[50] M. Mitchell, M. D. Thomure, N. L. Williams, *The role of space in the Success of Coevolutionary Learning*, Proceedings of ALIFE X - The Tenth International Conference on the Simulation and Synthesis of Living Systems (2006).

[51] F. Jiménez Morales, J. P. Crutchfield, M. Mitchell, *Evolving two-dimensional cellular automata to perform density classification: a report on work in progress*, Parallel Comput., 27 (2001), pp. 571-585.

[52] D.E. Moriarty, R.Miikkulainen, *Discovering complex othello strategies through evolutionary neural networks*, Connection Science, 7 (1995), 195-209.

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

[54] G.M.B. Oliveira, L.G.A. Martins, L.B. de Carvalho, E. Fynn, *Some investigations about synchronization and density classification tasks in one-dimensional and two-dimensional cellular automata rule spaces*, Electron. Notes Theor. Comput. Sci., 252 (2009), pp. 121-142.

[55] A.E. Olsson, *Particle Swarm Optimization: Theory, Techniques and Applications*, Nova Science Publishers, Inc., Commack, NY, USA (2010).

[56] B. Olsson, *A host-parasite genetic algorithm for asymmetric tasks*, In Claire Ndellec and Cline Rouveirol, editors, Machine Learning: ECML-98, LNCS vol. 1398, Springer Berlin / Heidelberg (1998), pp. 346-351.

[57] N. H. Packard, *Adaptation toward the edge of chaos. Dynamic Patterns in Complex Systems*, World Scientific (1988), pp. 293-301.

[58] L. Pagie, M. Mitchell, *A comparison of evolutionary and coevolutionary search*, Int. J. Comput. Intell. Appl., 2:1 (2002), pp. 53-69.

[59] M. Pelikan, D.E. Goldberg, E. Cantú-Paz, *Linkage problem, distribution estimation, and bayesian networks*, Evolutionary Computation, 8:3 (2000), pp. 311-340.

[60] Z.A. Perry, *Experimental study of speciation in ecological niche theory using genetic algorithms*, PhD thesis, University of Michigan, Ann Arbor, MI, USA (1984), AAI8502912.

[61] M.A. Potter, K.A. De Jong, *Cooperative coevolution: An architecture for evolving coadapted subcomponents*, Evolutionary Computation, 8 (2000), pp. 1-29.

[62] N. Richards, D.E. Moriarty, R. Miikkulainen, *Evolving neural networks to play Go*, Applied Intelligence, 8:1 (1998), pp. 85-96.

[63] P. Robbins, *The effect of parasitism on the evolution of a communication protocol an artificial life simulation*, In Proceedings of the third international conference on Simulation of adaptive behaviour: from animals to animats 3: from animals to animats 3, Cambridge, MA, USA, MIT Press (1994), pp. 431-437.

[64] C.D. Rosin, R.K. Belew, *Methods for competitive co-evolution: Finding opponents worth beating*, In Proceedings of the Sixth International Conference on Genetic Algorithms, Morgan Kaufmann (1995) pp. 373-380.

[65] G.G. Simpson, *The Baldwin effect*, Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA (1996), pp. 99-109.

[66] K. Sims, *Evolving 3D morphology and behaviour by competition*, Artificial Life, 1:4 (1994), pp. 353-372.

[67] K. Sims, *Evolving virtual creatures*, In Proceedings of the 21st annual conference on Computer graphics and interactive techniques, ACM (1994), pp. 15-22.
[68] D. Thierens, *Adaptive operator selection for iterated local search*, In Thomas Sttzle, Mauro Birattari, and Holger Hoos, editors, Engineering Stochastic Local Search Algorithms. Designing, Implementing and Analyzing Effective Heuristics, LNCS vol. 5752, Springer Berlin / Heidelberg (2009), pp. 140-144.
[69] E. Thro, *Artificial life explorer's kit*, chapter Natural A-Life. Sams Publishing, ISBN 0672303019 (1993).
[70] B.W. Wah, A. Ieumwananonthachai, *Teacher: A Genetics-Based System for Learning and for Generalizing Heuristics*, vol. Evolutionary Computation: Theory and Applications, World Scientific Publishing Co. Pte. Ltd (1999), pp. 179-211.
[71] D. Whitley, V. Gordon, K. Mathias, *Lamarckian evolution, the baldwin effect and function optimization*, In Yuval Davidor, Hans-Paul Schwefel, and Reinhard Mnner, editors, Parallel Problem Solving from Nature - PPSN III, LNCS vol. 866, Springer Berlin / Heidelberg (1994), pp. 5-15.
[72] R.P. Wiegand, M.A. Potter, *Robustness in cooperative coevolution*, In Proceedings of the 8th annual conference on Genetic and evolutionary computation, GECCO '06, New York, NY, USA (2006), pp. 369-376.
[73] J. Wiles, J.R. Watson, *How learning can guide evolution in hierarchical modular tasks*, In Proceedings of the 23rd Annual Conference of the Cognitive Science Society (2001), pp. 1130-1135.
[74] S. Wolfram, editor, *Theory and Applications of Cellular Automata*, Advanced series on complex systems, vol. 1, World Scientific Publishing, P. O. Box 128, Farrer Road, Singapore 9128 (1986).
[75] L. Yaeger, O. Sporns, S. Williams, X. Shuai, S. Dougherty, *Evolutionary selection of network structure and function*, In Proc. of the Alife XII Conference, Odense, Denmark (2010), pp. 313-320.
[76] T.L. Yu, K. Sastry, D.E. Goldberg, *Linkage learning, overlapping building blocks, and systematic strategy for scalable recombination*, In GECCO '05: Proceedings of the 2005 conference on Genetic and evolutionary computation, New York, NY, USA (2005), pp. 1217-1224.
[77] Y. Zhao, S. A. Billings, *Identification of the Belousov-Zhabotinsky Reaction using Cellular Automata Models*, International Journal of Bifurcation and Chaos, 17:5 (2007), pp. 1687-1701.

(1) Department of Computer Science, Babes-Bolyai University, Kogalniceanu 1, 400084 Cluj-Napoca, Romania

E-mail address: david.iclanzan@gmail.com, {anca,cchira}@cs.ubbcluj.ro

(2) Faculty of Economics and Business Administration, Babes-Bolyai University, Kogalniceanu 1, 400084 Cluj-Napoca, Romania

E-mail address: rodica.lung@econ.ubbcluj.ro