DOI: http://dx.doi.org/10.7551/978-0-262-31709-2-ch011

Evolution of Incremental Complex Behavior on Cellular Machines

Stefano Nichele[1] and Gunnar Tufte[1]

[1] Norwegian University of Science and Technology, Department of Computer and Information Science,
Sem Selandsvei 7-9, 7491, Trondheim, Norway
{nichele, gunnart}@idi.ntnu.no

Abstract

Complex multi-cellular organisms are the result of evolution over billions of years. Their ability to reproduce and survive through adaptation to selection pressure did not happen suddenly; it required gradual genome evolution that eventually led to an increased emergent complexity. In this paper we investigate the emergence of complexity in cellular machines, using two different evolutionary strategies. The first approach is a conventional genetic algorithm, where the target is the maximum complexity. This is compared to an incremental approach, where complexity is gradually evolved. We show that an incremental methodology could be better suited to help evolution to discover complex emergent behaviors. We also propose the usage of a genome parameter to detect the behavioral regime. The parameter may indicate if the evolving genomes are likely to be able to achieve more complex behaviors, giving information on the evolvability of the system. The experimental model used herein is based on 2-dimensional cellular automata. We show that the incremental approach is promising when evolution targets an increase of complexity.

Introduction

Evolved artificial developmental systems' remarkable range of products, i.e. biological organisms, with variety in form, function and "complexity", all tailored to fill their niche, is an alluring design concept. Such bio-inspired design methodology can be used for any kind of system to create a variety of artifacts that can handle different problems.

However, knowledge and methods to be able to exploit similar core processes [14] for the design of artificial organisms are still subjects of exploration and research.

Evolved artificial developmental systems have shown many favorable features that are borrowed from natural biological systems, such as the main subject here, the ability to evolve inherent complexity as a response to evolutionary pressure [27]. Evolvable, or in particular EvoDevo systems are products of bottom-up processes, in contrast to typical top-down engineering design approaches. A system emerging as a product of a bottom-up process can target system properties out-of-bounds for traditional top-down designed artifacts. Self-organization, self-construction, adaptivity, scalability and robustness are all example of such hard to reach properties.

In contrast to the open ended evolution in nature, evolution of artificial EvoDevo systems often includes an expressed goal; fitness is a kind of usability [15] measurement. The target functionality is defined and thus placed within some complexity measure. Such complexity can be defined at several levels. The complexity of the machine's composition, i.e. the number of components and connections can be quantified. Another complexity measurement may be functionality in terms of information processing. Quantification of complexity in artifacts and biological organism has no common defined unit of measurement or ratio of comparability. However, intuitively there are differences. Such differences can be related to the composition of artifacts/organisms or as a measure of their functionality. If complexity, for an organism, is a measurement of functional properties within its environmental niche, different levels of behavioral complexity can be said to exist. In this context, high complexity is not a goal in itself; it is merely a product of the species adaptation to be able to reproduce. The genetic information included in the genotype and the developmental processes for any particular specie has evolved and diverged through adaptation from the primordial soup. As such, the behavioral complexity of organisms is a product of the evolved interplay between genetic information and developmental processes.

As a step toward more knowledge of underlying processes and finding design methodologies for the exploitation of EvoDevo for artificial systems, we investigate how a gradual change in the complexity requirements, in evolutionary time, influence on EvoDevo system's ability to evolve complexity. Further, a variation of Langton's Lambda parameter [16] is used as an indicator of evolutionary genome adaptation to resulting phenotypic complexity. The taken experimental approach uses a kind of incremental complexity evolution to simulate the process of species adaptation to a changing environment requiring growth of complexity. A 2D cellular developmental model based on Cellular Automata (CA) is used in the experiments, so as to be able to visualize artificial organisms' development in 2 dimensions.

In our incremental evolutionary approach, the evolution process tackles the problem of targeting complexity incrementally. Instead of seeking for maximum complexity in the early generations, the problem is divided in sub-problems. Generations are divided in intervals and in every interval the target complexity demand is increased, keeping the target functionality unvaried, i.e. the class of problems is the same. In such way, it may be possible to evolve favorable genes for intermediate complexity levels, which may be beneficial in order to achieve higher complexity in the long term.

Since no universally accepted definition of complexity exists, many authors use it implicitly without specifying which

notion of complexity they are using. Yet, without any common measure of genotype or phenotype complexity, any significant claim is not verifiable. Genome complexity measures may sometimes be unfitting, since the amount of information encoded in the genome is not directly proportional to the complexity of the emergent organisms. Even in nature, some unicellular eukaryotic organisms have much larger genomes than humans. In addition, there are other factors that impact on the organisms' complexity during their growth, e.g. the environment.

One may argue that complexity of an EvoDevo system may be measured in terms of information contained in a genome, by ranking through a Turing machine, by quantifying the capacity for a genome to exploit a provided area of growth, using approximations of Kolmogorov complexity [5, 6, 9]. The used complexity measure is based on compression of CA behavior in terms of trajectory and attractor lengths, a kind of adaptation of principles from Kolmogorov complexity.

The article is laid out as follows: background information and motivation on incremental evolution is presented in Section 2. In Section 3 Lambda genome parameter is introduced and in Section 4 the developmental model used in the experiments is described. Section 5 explains the genetic algorithms used herein. The experimental setup is illustrated in Section 6 and in Section 7 the results of the experiments are presented together with a discussion of the ideas and the results. Finally Section 8 concludes the work.

Incremental Evolution

A general evolutionary strategy may be too difficult for the evolution system to discover possible solutions directly. Instead, it is possible to learn complex behaviors incrementally, starting from a simple behavior and gradually making the task more challenging [1]. Incremental evolutionary approaches have been used successfully to evolve complex behaviors step by step. Many studies investigated the training of artificial neural networks with Genetic Algorithms (GAs), in order to evolve robots controller able to perform complex action sequences, e.g. complex light switching behavior [2] or robot duels controllers [3]. This approach has shown interesting results, being able to evolve converged populations to the new task. On the other hand, conventional evolutionary methods may have too high selective pressure in the early stages of the evolution, getting the GA blocked in an unfruitful area of the search space. If the population is first evolved to an easier behavior, it may be possible to discover and access a region of the solution space where even more complex behaviors are more likely to be found. As such, the ultimate complex behavior may be reached incrementally by evolving a sequence of intermediate behaviors with growing complexity:

$$\text{Behavior 1} \xrightarrow{\Delta} \text{B 2} \xrightarrow{\Delta} \ldots. \xrightarrow{\Delta} \text{B n-1} \xrightarrow{\Delta} \text{B n}$$

In this way, genotypes are evolved gradually and the search is driven on solutions that are likely to benefit and retain existing capabilities. Conventional evolution tends to fluctuate between idiosyncratic but still not interesting solutions [12]. Incremental evolution may foster continuing innovation by elaborating on available solutions.

Genome Parameter: λ

In our cellular developmental model, we are aiming to target complex phenotypic properties. Attractor length, i.e. development reaches a structure or state that is stable by self-regulation (point attractor) or a dynamic phenotypic structure that is self-reorganizing (cyclic attractor), is the chosen metric. The strategy is therefore to evolve intermediate genotypes that develop and express specific attractor lengths. Every fixed number of generations, we increase the sought attractor length value to increment the complexity demand.

In terms of evolvability, since we want to investigate if the evolving genotypes are able to evolve and develop more complex phenotypes, we attempt to measure the behavioral regime using a genome parameter. Parameters obtained from the genome information can be used to estimate the dynamic behavior of the system. In this work, the genotypes are represented as a transition rule table, where developmental actions are defined as a function of the neighborhood configuration (see next chapter for details). In this way, it is possible to analyze the different developmental actions and calculate parameters obtained from the genome table.

Several genome parameters have been previously proposed in order to measure genotype properties. Langton [16] studied a parameter λ as a measure of the activity level of the system and its disorder. A similar parameter, neighborhood dependent, is Absolute Activity presented by de Oliveira [20]. Li [21] introduced Mean Field Parameters to monitor if the majority of the regulatory actions follow the "mean" configuration. de Oliveira [20] presented a very similar parameter called Neighborhood Dominance. Binder [22, 23] introduced the Sensitivity parameter which measures the number of changes in the output of the transition table based on a change in the neighborhood, one cell at a time, over all the possible neighborhoods of the rule being considered. This has also being studied by de Oliveira [24, 20] as Context Dependence. Different properties of genome parameters have been investigated in details in [11]. In particular, the λ parameter has shown interesting abilities to discriminate genotypes in different behavioral classes, e.g. fixed, chaotic, random [13]. As such, we monitor the λ value along the evolutionary process. λ is calculated according to Equation 1.

$$\lambda = \frac{K^N - n}{K^N} \tag{1}$$

n represents the number of transitions to the quiescent state (i.e. inactive or dead state), K is the number of cells types and N is the neighborhood size (see following section for details).

Cellular Developmental Model

The developmental model used in this work is a minimalistic cellular developmental model based on cellular automata, similar to cellular models used in [25, 26, 19]. The system herein is close to the field of Morphogenetic Engineering [7], where the goal is "self-architecturing" systems. In embryomorphic systems [8], the approach is based on embryogenesis: the self-assembly of myriads of cells starting from a single zygote which holds the complete genotype information.

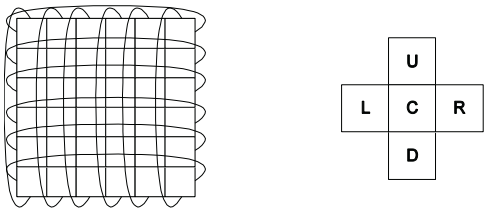

Fig. 1: Developmental model with cyclic boundary conditions and von Neumann neighborhood configuration.

A CA can be considered as a developing organism, where the genome specifications and the gene regulation information control the cells' growth and differentiation. The behavior of the CA is then represented by the emerging phenotype, which is subject to size and shape modifications, according to the cellular changes along the developmental process. Such dynamic developmental system can show adaptation, self-modification, plasticity [18] or self-replication properties [17].

L	R	U	D	C	$C_{(t+1)}$
0	0	0	0	0	0
0	0	0	0	1	{0,1,2}
0	0	0	1	0	{0,1,2}
0	0	0	1	1	{0,1,2}
0	0	1	0	0	{0,1,2}
		:			:
1	1	1	1	1	{0,1,2}
0	0	0	0	2	{0,1,2}
0	0	0	2	0	{0,1,2}
0	0	0	2	1	{0,1,2}
0	0	0	2	2	{0,1,2}
		:			:
2	2	2	2	2	{0,1,2}

Fig. 2: Developmental table with neighborhood configurations and relative developmental actions.

The model is based on a two-dimensional cellular automaton with cyclic boundary conditions, as shown in Figure 1. The number of cell types is set to three (type 1 and 2 plus the quiescent or dead cell type 0) in order to keep the property of multicellularity. A single cell (zygote) is placed in the centre of the development grid and develops according to a developmental table based on von Neumann's neighborhood (five neighbors). All the possible regulatory input combinations are explicitly represented in the development table, i.e. 243 (3^5) neighborhood configurations.

To ensure that cells will not materialize where there are no other cells around, a restriction has been set: if all the neighbors of an empty cell are empty, the cell will be empty also in the following development step. This is represented in the first entry of the developmental table in Figure 2. A more detailed description of the development model is given in [10, 11].

Genetic Algorithm

The Genetic Algorithm used in the experiments is tested with two different fitness functions: a classical fitness approach that targets the maximum complexity and an incremental growth of fitness. An incrementally growing fitness denotes a changing environment that requires more complex behaviors to survive. It must be underlined that the two different fitness functions could in theory perform the same way. The same genotypes could be discovered through evolution since there are no restrictions in the areas of the search space that are being explored. Anyway, this is very unlikely to happen, since the environment is interpreted differently on the evolutionary time scale. The GA consists of a population of ten individuals and uses a roulette wheel technique for proportionate selection of two potentially useful individuals. The worst three elements are replaced by two new individuals that are copies of the two selected ones with mutation rate 0.02 for each of the entries in the developmental table. The third new element is generated by uniform one-point crossover of the two selected individuals.

As we mentioned, the main difference between the two evolutionary strategies lies in the fitness function. In the classical scenario, the fittest individual is the one with longest attractor length. In the incremental approach the fittest is the individual with smallest difference between the actual length and the target length in that specific generation, i.e. environmental requirements at specific moments in evolutionary time.

The target length is defined as follows: in the first generation it is set as 1, i.e. point attractor. In this phase, the GA searches for phenotypes that end up with a single point attractor. Every fixed number of generations, the target value is incremented by a constant value. It is expected that in the following interval of generations, the population will be able to evolve and adapt towards the new target, i.e. an increasing complexity demand for longer attractor length along the evolutionary timeline.

Details on the development process, number of generations, length of the intervals and initial conditions of the genotypes are given in the next section, which describes the experimental setup. Source code is available upon request.

Experimental Setup

In the experiments herein, the main idea is to generate an initial population of ten genotypes, develop the corresponding phenotypes (starting from a single cell placed in the centre of the grid) until an attractor is found, evaluate the phenotypes with a fitness function and evolve the chosen genomes throughout the generations. This process is repeated for GA with standard fitness function and GA with incremental fitness function. The performances of the different algorithms are evaluated, measuring the ability to achieve sought complexity in terms of attractor length, i.e. number of development steps between two repetitions of the same state. This experimental setup is represented graphically in Figure 3.

Two different strategies of generating the initial population are investigated:

- From "dead genomes": all the transitions in the developmental table lead to the dead state and the value of λ is uniform. In this scenario, the GA has to evolve "dead" genomes, i.e. the developed phenotype results in a dead

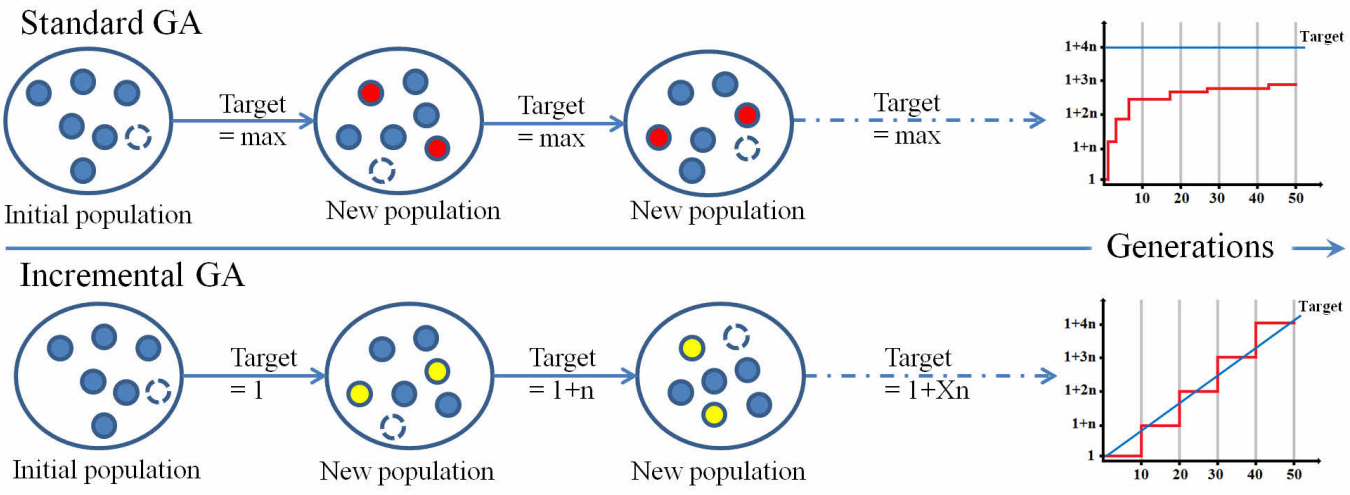

Fig. 3: Experimental Setup: GA with standard fitness on top (targeting maximum complexity), GA with incremental fitness on bottom (target fitness increased gradually)

organism after the first development step, towards "alive" genomes. This approach could be interesting since favorable genes are evolved from scratch, especially when random initialization is not possible or feasible.

- From random genomes: the initial population of genomes is initialized randomly[1]. In this way, it is possible to have extremely fit genomes, or very unfit, from the first generations. In both cases, it may be particularly difficult to evolve towards genomes with favorable characteristics to reach the defined goal. The parameter λ has a distributed value.

The way genomes are generated has a strong impact on how the resulting phenotypes will behave. Trying to understand the relation between genotypes and possible resulting phenotypes means understanding which kind of information is present in the genome and which behavioral properties may emerge. Since in our model all the possible regulatory combinations are fully specified together with the corresponding developmental actions, it is possible to calculate the Lambda genome parameter for each individual in the population. λ is calculated out of the regulative outcome in the developmental table, i.e. column C(t+1) in Figure 2. Following Langton's definition [16], a quiescent state must be chosen. We choose the void cell type (type 0) as the quiescent state. Lambda is then calculated by 1 – the ratio between transitions to the quiescent state and the total number of transitions in the developmental table. It is implicit that if the population is initialized with dead genomes, all the transitions in the developmental table will lead to the quiescent state. Thus, λ will be 0, which means genotypes with low behavioral activity. On the other hand, when the population is initialized with random genomes, λ is more likely in the vicinity of a critical behavioral regime, near the Edge of Chaos [16]. In this area of the solution space, it is more likely to find complex behaviors.

[1] Marsenne twister is used for initialization of randomized genotypes and genetic operators (mutation and crossover).

Monitoring Lambda along all the evolutionary process will give information on the ability of the population to evolve and adapt to the target complexity level. λ, as an indication of computation, has been discussed in [28]. However, from previous work [16, 11], we know that the attractor length of a certain organism is strongly related to its λ value, which can be calculated from the genome composition. As such, Lambda could be used to drive evolution in desired parts of the search space where the desired behavior is more likely to be found. This is part of ongoing experimentation. λ here is measured to gain information of the evolution of genome composition and interpreted as an indicator of the evolvability of the system, which may confirm our hypothesis that an increase in complexity is more likely to happen if evolutionary search leads towards the desired behavioral regime.

Results and Discussion

In the experiments herein, the array size of the CA was set to 4x4. The size of the arrays was chosen as to be able to carry out experiments in reasonable computational time.

Organisms of 4x4 cells may be considered rather small; however, the theoretical maximum attractor length is 3^{16}. As such, even at the chosen array size, the number of development steps to reach an attractor could be rather big.

Experiment 1: Dead Genomes

The first set of experiments consists in comparing the behavior of standard GA and incremental GA, staring from dead genomes. In both cases, the GAs run for 30000 generations. The standard approach targets the maximum attractor length for all the 30000 generations whether the incremental approach increments the target attractor length by 10 development steps every 20 generations. It is noticeable that there are clear advantages with an incremental approach for the evolution of complexity.

In Figure 4 (a) the results of a canonical GA are presented. Here the target was the maximum complexity. It is clear that a standard approach could discover in some early generations

good candidates but often the algorithm gets trapped in some unfruitful regions of the search space and for several generations there are no improvements. All the single samples are represented by the thin lines. The dashed line is the average over all the tests and after 30000 generations the attractor length has a value around 4000 development steps. Here the deviation is quite big, since in some cases the maximum length is far from the average. For example, in one case it is close to 12000 development steps while in many other cases is lower than 2000. In Figure 4 (b) the results for the incremental approach are plotted. The straight line represents the target complexity value for each generation, measured as number of development steps inside the attractor. The thin lines here follow quite accurately the target line. The dashed line represents again the average. It is possible to see that average and target are overlapping for the first 15000 generations, whether in the last 15000 generations the average is slightly lower than the target. Overall, the average attractor length after 30000 generations is around 13000 development steps. Here the deviation, represented by the dotted line, is quite small. This means that the incremental approach is able to minimize the distance from the target in each generation. Figure 4 (c) is a comparison of the two different strategies.

It is evident that the incremental approach overcomes the standard approach since the first 2500 generations. After that, the canonical GA struggles to find good solutions on average and has difficulties to evolve and jump up in complexity. On the other hand, the incremental approach shows very promising results even if in the last generations there is a small degradation of the performances. Overall, the difference between the averages is significant ($p<.0001$, Student's t-test).

Finally, Figure 4 (d) represents a comparison of the measured Lambda parameter in each generation. The standard GA evolves to genotypes with a maximum parameter value of 0.4. The incremental GA discovers genotypes with Lambda between 0.6 and 0.7. This means that those genotypes are in a completely different behavioral regime. Earlier work from Langton [16] identified a critical value of Lambda where the behavioral regime of the system encounters a phase transition between ordered and chaotic dynamics. In such area of the search space it is more likely to find the primitive functions to support computation: transmission, storage and modification of information. Further support to this hypothesis is given by previous work on the investigation of probable relationship between attractor length and Lambda [10]. In the experiments herein, λ is used as a measurement and its increase is not a

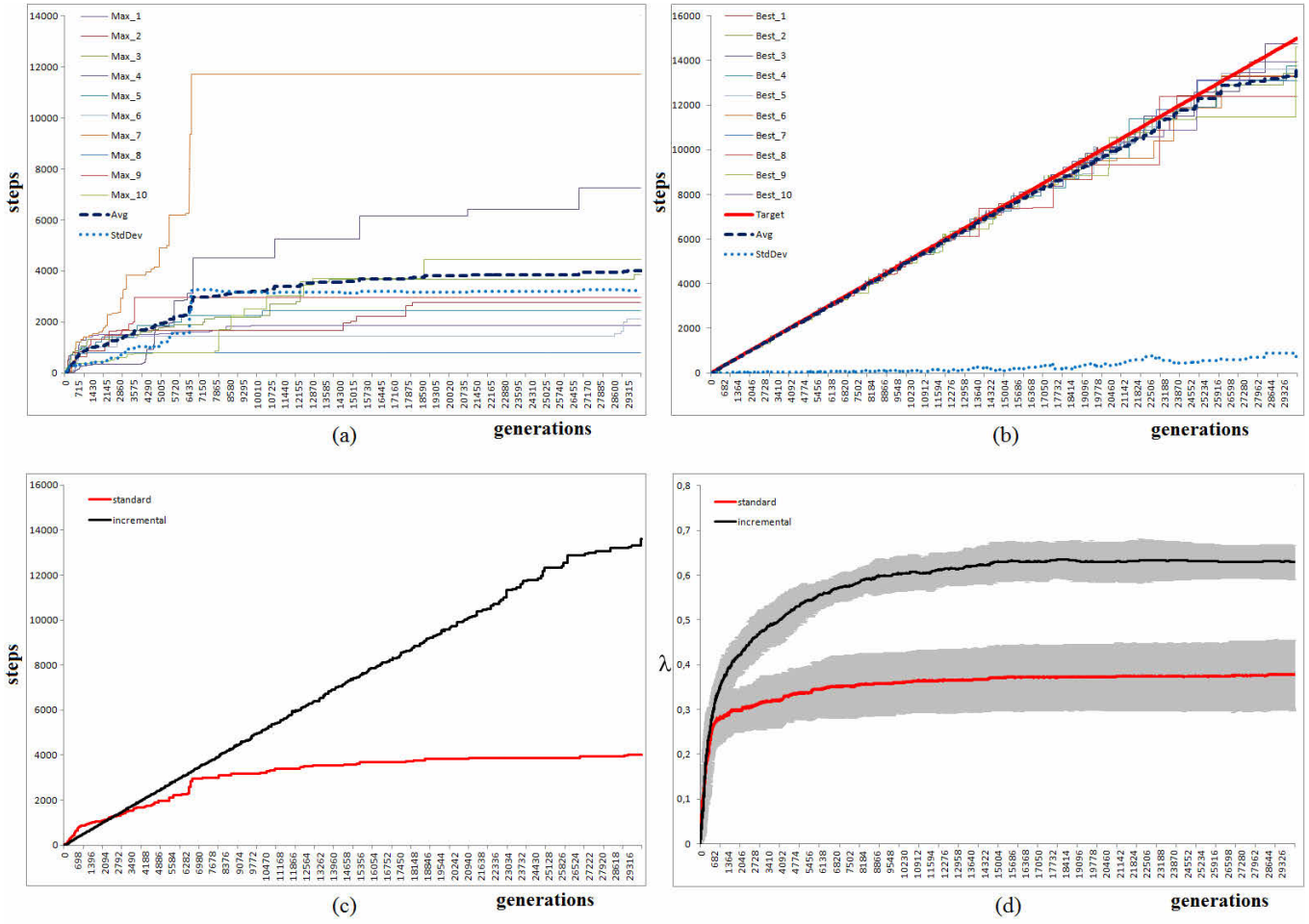

Fig. 4: Results for Experiment 1, developmental tables initialized with "dead genomes". Avg. and std. dev. over 10 runs. (a) standard GA approach: generations vs. attractor length (thin lines represent single runs); (b) incremental GA approach: generations vs. attractor length (thin lines represent single runs); (c) comparison of averages: standard GA approach vs. incremental GA approach; (d) comparison of Lambda parameter: λ for standard GA approach vs. λ for incremental GA approach.

goal in itself. Figure 5 shows a plot over the λ space where genotypes were generated according to a specific parameter value and the resulting attractor length was measured.

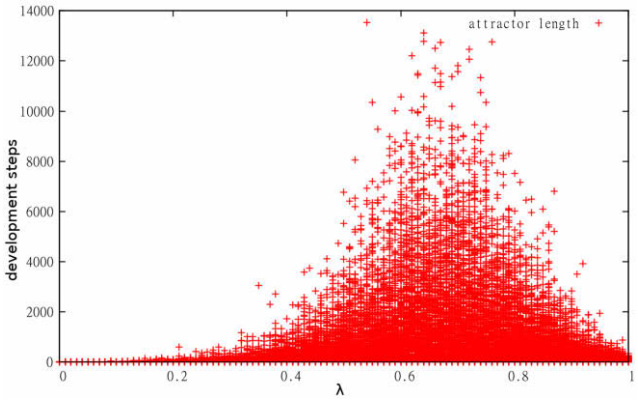

Fig. 5: attractor length as function of λ. Results for 4x4 organisms and 1000 tests for each λ value. Adapted from [10].

In those experiments, the cellular automata configuration was the same as in the experiments herein: 2-dimensional grid with neighborhood of size 5 and 3 possible cell types. With such configuration, the most heterogeneous genotypes are generated when λ is 0.66. In fact, in the scattered plot, it is more likely to find long attractors in that area of the solution space. On the other hand, when λ is around 0.4, the behavioral regime is in an intermediate region where organisms show ordered dynamics. As such, it may be more challenging to evolve towards longer attractor lengths. Relating this results with those in Figure 4 (d), it is possible to conclude that the standard GA gets trapped in an area of the search space where the sought behavior (maximum complexity) is less likely to be found. Moreover, it may be difficult for the GA to escape from such region of the search space. The incremental approach is able to evolve genotypes with parameter value around 0,65, which may be beneficial to find longer attractors. Even if not so good solutions are found in that area of the search space, it may be still more probable for the GA to be able to discover better solutions, since the sought behavior is more likely to appear.

Extended Evolutionary Time

Subsequently, the incremental GA is executed again for 500.000 generations and the target is incremented by 10 development steps every 500 generations. This is done to check the behavior of the GA when the algorithm has more generations to evolve and adapt the population towards the new complexity value.

In Figure 6 (a) the target line and the actual line (average) are completely overlapping. This means that, given enough time to the population to evolve to the sought complexity level, it is possible to keep increase the complexity with minimum deviation from the target. In each generation interval, the genetic algorithm is able to discover favorable genes and use it as a starting point for the next intervals. Evolution is based on already present capabilities, developed incrementally.

Figure 6 (b) and Figure 6 (c) respectively represent the average distance from the target and the percent average distance from the target. It is possible to observe that such span predictably increases along the 500000 generations. Even that, the average distance from the target level suddenly decreases below 1% since the first generations. In conclusion, it may be possible to tune the generation intervals in a way that evolution has enough time to evolve the whole population and prepare it to the following complexity improvements.

Experiment 2: Randomized Genomes

In the second set of experiments, the behavior of standard GA and incremental GA is tested again for 50000 generations, starting from random genomes, i.e. genomes initialized with a uniform random distribution among the three cell types. By doing that, the standard GA proceeds to select the individual in the population with longest attractor length, targeting maximum complexity. As such, in Figure 7 (a) the average attractor length does not start from the origin. During the first few generations there are several jumps in complexity but after this fruitful stage the average line stabilizes and tends to become flat. Figure 7 (b) summarizes the results for the incremental approach. In this case, since complexity is evolved gradually, individuals with long attractor lengths are left aside in favor of individuals that are closer to the sought initial behavior, i.e. point attractor.

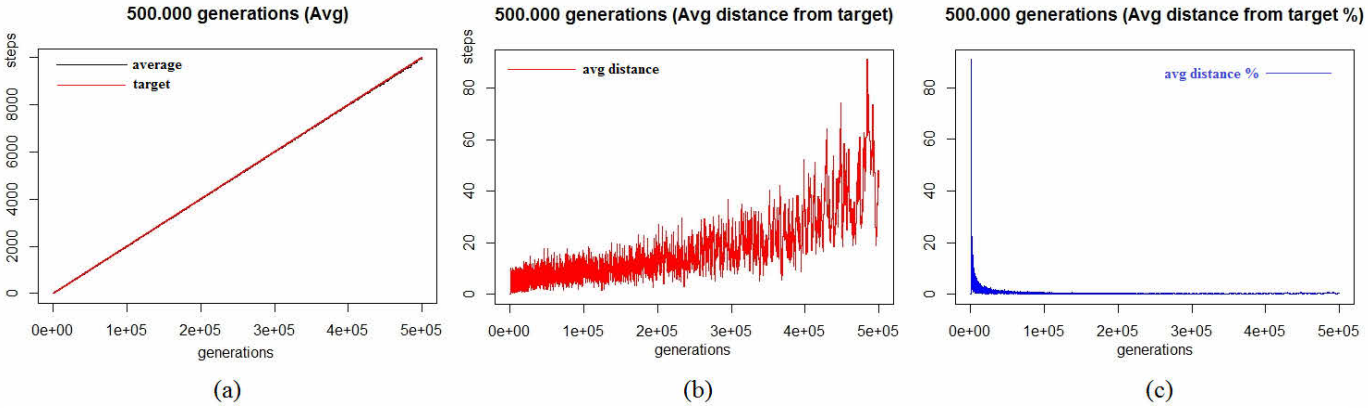

Fig. 6: Incremental GA approach with extended evolutionary time (500000 generations), developmental tables initialized with "dead genomes", average over 6 runs. (a) Target attractor length and actual attractor length (overlapping); (b) Average distance from target in development steps; (c) Average distance from target %

Implicitly, in the beginning the algorithm is forcing organisms to exhibit low complexity to survive, since only in the subsequent generations the environment will become more demanding and will require evolving in complexity. During the first 25000 generations the target line and the actual line are very close. In the last 25000 generations the reached complexity level is very similar to the GA with standard fitness, both in terms of average attractor length and standard deviation. The difference between the averages is not significant (p=.0850, Student's t-test).

Figure 7 (c) and 7 (d) show the average λ along the generations. Since genotypes were initialized randomly, it is more likely that the developmental tables are the most heterogeneous [16].

As a result, λ is positioned already in an area of the solution space where highly complex individual are likely to appear. For the standard GA, the average λ is smoother and does not show high activity of the GA. Once the algorithm finds good solutions, it is more probable that will hardly improve with better solutions. This could be a drawback for evolvability, especially if one would like to evolve intermediate complexity levels. On the other hand, for the incremental approach, λ fluctuates more within the behavioral region, exhibiting high activity of the GA that continues to explore the solution space. Moreover, the incremental strategy would fit better if the system would need to reach intermediate complexity levels.

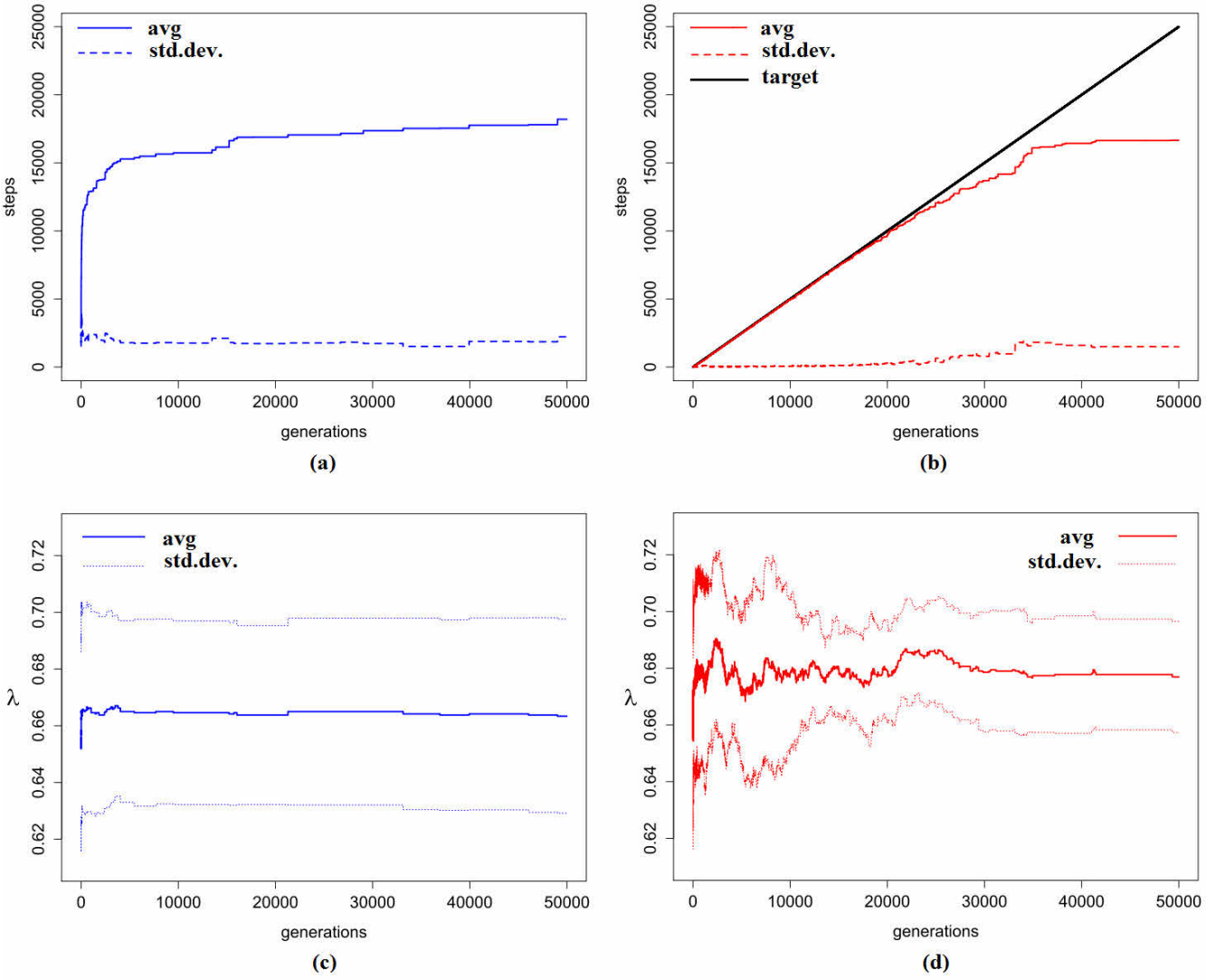

Fig. 7: Results for Experiment 2, developmental tables initialized with random genomes. Average and standard deviation over 10 runs. (a) standard GA approach: generations vs. attractor length (avg. and dev.); (b) incremental GA approach: generations vs. attractor length (avg. and dev.); (c) Lambda parameter for standard GA approach (avg. and dev.); (d) Lambda parameter for incremental GA approach (avg. and dev.).

Conclusion

The presented experiments investigated the emergence of complexity in cellular machines, using two different evolutionary strategies: a standard approach, where the target was the maximum complexity and an incremental approach, where complexity was gradually evolved.

We showed that the incremental approach has clear advantages when evolution targets an increase of complexity, especially when the population's genome parameter is towards uniform, i.e. intermediate complexity levels in equilibrium with the environmental pressure. Such knowledge is important at the design stage of EvoDevo systems, where developmental actions are not manually programmed but discovered through evolutionary processes.

We also proposed the usage of Lambda genome parameter to detect the behavioral regime. This may be useful to indicate if the evolving genomes are likely to be able to achieve more complex behaviors, giving information on the evolvability of the system. Such ability of adaptive evolution is necessary for a system to be able to evolve complexity.

Moreover, when it comes to adaptivity, the results herein show that genomes with a given parameter value will most likely mutate to genomes with similar developmental behavior, as long as the mutation results in an offspring with similar parameter value. Our current work is focused on the usage of genome parameters to guide evolution towards favorable areas of the solution space. Furthermore, genome parameters may help to keep the population closer to the desired developmental behavior and supervising its genetic distance. This is in tune with at least two points of the current challenges in the field of artificial life [4]: "explain how rules and symbols are generated from physical dynamics" and "develop a theory of information processing, information flow and information generation for evolving systems".

As a future work, it may be possible to compare the robustness of solutions evolved incrementally versus solutions evolved with a standard approach. In particular, how fragile they are to external perturbation, both at genotype level, i.e. mutations in the rule table, and at phenotype level, i.e. perturbation of the system state during development.

References

[1] F. Gomez and R. Miikkulainen. Incremental evolution of complex general behavior. Adaptive Behavior, 5: pages 317-342 (1997).

[2] M. Schuster and C. Harman. Incremental evolution of complex light switching behavior. CS 81 Adaptive Robotics. Swarthmore College (2006).

[3] K. O. Stanley and R. Miikkulainen. Competitive coevolution through evolutionary complexification. Journal of artificial intelligence research, 21: pages 63-100 (2004).

[4] M. Bedau, J. McCaskill, N. Packard, S. Rasmussen, C. Adami, D. Green, T. Ikegami, K. Kaneko and T. Ray. Open problems in artificial life. Artificial Life, 6: pages 363-373 (2000).

[5] T. Kowaliw. Measures of complexity for artificial embryogeny. GECCO 2008, pages 843-850. ACM (2008)

[6] P. K. Lehre and P. C. Haddow. Developmental mappings and phenotypic complexity. In Congress on Evolutionary Computation, CEC2003, pages 62–68. IEEE (2003).

[7] R. Doursat, H. Sayama and O. Michel. Morphogenetic engineering: toward programmable complex systems. Understanding Complex Systems Series, Springer-Verlag (2012).

[8] R. Doursat, C. Sánchez, R. Dordea, D. Fourquet and T. Kowaliw. Embryomorphic engineering: emergent innovation through evolutionary development. In Morphogenetic Engineering: Toward Programmable Complex Systems, R. Doursat, H. Sayama and O. Michel, pages 275-311. "Understanding Complex Systems" Series, Springer-Verlag (2012).

[9] S. Harding and W. Banzhaf. Organic Computing, chapter Artificial Development, pages 201–220. Springer Verlag (2008).

[10] G. Tufte and S. Nichele. On the correlations between developmental diversity and genomic composition. 13th Annual Genetic and Evolutionary Computation Conference, GECCO 2011, pages 1507-1514. ACM (2011).

[11] S. Nichele and G. Tufte. Genome parameters as information to forecast emergent developmental behaviors. 11th International Conference on Unconventional Computation and Natural Computation, UCNC 2012, pages 186-197. Springer (2012).

[12] D. Floreano and S. Nolfi. God save the red queen! Competition in co-evolutionary robotics. Evolutionary computation, 5 (1997).

[13] S. Wolfram. Universality and complexity in CA. Physica D, 10 (1-2): pages 1–35 (1984).

[14] M. Kirschner and J. Gerhart. The plausibility of life: resolving Darwin's dilemma. Yale University press (2006).

[15] U. Krohs and O. Kroes. Functions in biological and artificial worlds. The Vienna series in theoretical biology. MIT press (2009).

[16] C. G. Langton. Computation at the edge of chaos: phase transitions and emergent computation. In S. Forrest, editor, Emergent Computation, pages 12–37. MIT Press (1991).

[17] C. G. Langton: Self-reproduction in cellular automata. Physica D, 10: pages 135–144 (1984).

[18] M. J. West-Eberhard. Developmental plasticity and evolution. Oxford Univ. Press (2003).

[19] G. Tufte. Evolution, development and environment toward adaptation through phenotypic plasticity and exploitation of external information. S. Bullock, J. Noble, R. Watson, and M. A. Bedau, editors, Artificial Life XI, pages 624–631. MIT Press, Cambridge, MA (2008).

[20] G. de Oliveira, P. de Oliveira and N. Omar. Definition and application of a five-parameter characterization of one-dimensional cellular automata rule space. MIT, Artificial Life7: pages 277-301 (2001).

[21] W. Li. Phenomenology of nonlocal cellular automata. Santa Fe Institute. Journal of Statistical Physics, 68(5-6): pages 829-882 (1992).

[22] P. M. Binder. A phase diagram for elementary cellular automata. Complex Systems, 7, pages 241-247 (1993).

[23] P. M. Binder. Parametric ordering of complex systems. Physical Review E, vol. 49 n. 3, pages 2023-2025 (1994).

[24] G. de Oliveira, P. de Oliveira and N. Omar. Guidelines for dynamics-based parameterization of one-dimensional cellular automata rule space. John Wiley & Sons, Inc. Vol. 6, No. 2 Complexity (2001).

[25] T. Kowaliw, P. Grogono and N. Kharma. Environment as a spatial constraint on the growth of structural form. GECCO 2007, pages 1037–1044, New York, NY, USA (2007).

[26] J. F. Miller and W. Banzhaf. Evolving the program for a cell: from french flag to boolean circuits. In S. Kumar and P. J. Bentley, editors, On Growth, Form and Computers, pages 278–301. Elsevier Limited Oxford UK (2003).

[27] R. Dawkins. The blind watchmaker. Chapter 3: accumulating small changes. Norton & Company Inc. (1986).

[28] M. Mitchell, P. T. Hraber and J. P. Crutchfield. Revisiting the Edge of Chaos: evolving cellular automata to perform computations. Complex Systems 7: pages 89-130 (1993).