Investigation of Genome Parameters and Sub-Transitions to Guide Evolution of Artificial Cellular Organisms

Stefano Nichele, Håkon Hjelde Wold and Gunnar Tufte

Norwegian University of Science and Technology
Department of Computer and Information Science
Sem Selandsvei 7-9, 7491, Trondheim, Norway

{nichele@idi, haakonhw@stud, gunnart@idi}.ntnu.no

Abstract—Artificial multi-cellular organisms develop from a single zygote to complex morphologies, following the instructions encoded in their genomes. Small genome mutations can result in very different developed phenotypes. In this paper we investigate how to exploit genotype information in order to guide evolution towards favorable areas of the phenotype solution space, where the sought emergent behavior is more likely to be found. Lambda genome parameter, with its ability to discriminate different developmental behaviors, is incorporated into the fitness function and used as a discriminating factor for genetic distance, to keep resulting phenotype's developmental behavior close by and encourage beneficial mutations that yield adaptive evolution. Genome activation patterns are detected and grouped into genome parameter sub-transitions. Different sub-transitions are investigated as simple genome parameters, or composed to integrate several genome properties into a more exhaustive composite parameter. The experimental model used herein is based on 2-dimensional cellular automata.

Keywords: Artificial Development, Evolution, Complexity, Emergence, Cellular Automata.

1 Introduction

Evolved artificial developmental (EvoDevo) systems have shown many favorable features that are also present in natural biological systems, such as the ability to evolve robust genomes [1]. However, robustness and evolvability may not be always rowing in the same direction. A biological organism may be considered robust if, after genome mutations, it keeps the same ability or functional properties. In contrast, evolvability is a property that promotes genetic variation in order to produce adaptive evolution, being able to evolve through natural selection. One may think that too high robustness would not provide enough genetic diversity whereas too high evolvability would cause more disadvantageous mutations, thus annihilating adaptation. In EvoDevo systems, small changes in the genome often lead to completely different emergent phenotypes. It is particularly difficult to understand which changes will be produced to the developing organism by each genetic operator, e.g. mutation, crossover, and which phenotypic traits will be affected. As such, evolutionary algorithms spend a relevant amount of time generating low fitness solutions that may not give any genetic contribution to the population and thus being often discarded. We investigate if genome information could be used to guide evolution. Our results indicate that genome parameters could predict the developing behavior based on genome composition and thus help to guide evolutionary search in the right area of the search space, where the sought behavior is more likely to be found.

The article is laid out as follows: background information and motivation is presented in Section 2. In Section 3 the developmental model used in the experiments is presented. Section 4 describes Lambda genome parameter, meaning and usage. The experimental setup is illustrated in Section 5. In Section 6 and 7 the results of the experiments are presented together with the discussion. Section 8 concludes the work.

2 Motivation and Background

Artificial developmental systems can be considered as complex systems [2], where there is no central controller and the developed artificial organisms are the result of an emergent process out of the local interactions of simple cells. Many developmental systems target specific phenotypic structures or structural properties [3], whether some others execute a particular computational task that emerges out of the development of the machine' structure [4]. Programming an artificial developmental system to produce such emergent computation cannot be done using traditional engineering approaches. One solution could be to exploit nature's way of tackling problems, namely evolution by natural selection. Evolutionary algorithms have been widely used as population-based metaheuristic optimization algorithms [5]. In general, those evolutionary techniques do not make any assumption about the underlying fitness landscape. An indirect genotype-to-phenotype mapping can result in two very similar genotypes developing into two very different phenotypes. A developmental mapping may be represented by a function that maps elements in the genotype space to elements in the phenotype space. Such spaces may have regions where small distances between genotypes are preserved into small differences between resulting phenotypes, whether in some other regions distances are hardly preserved at all [12]. In practice, small mutation can have a huge impact on the emergent phenotype. This can be problematic if solutions are to be discovered by evolutionary algorithms. Having a genome parameter that may predict the emergent behavior could be useful to reduce phenotypic distance. Such information could contribute to guide evolutionary search throughout the solution space, where the target phenotype may plausibly appear.

3 Evolution and Development

The relation between natural evolution and development in biological systems is still a fairly unexplored area [13]. Investigation of natural evolution makes it hardly possible to obtain experimental proofs due to the time scale of evolution. In evolved artificial systems there is no such problem. It is possible to execute experiments in a reasonable time and investigate different evolutionary factors that may influence on developmental paths. However, there is a lack of knowledge of what kind of information must be present in the genome in order to obtain a sought phenotypic behavior. We try to exploit genome regulatory information in a simple developmental model and investigate if such information could contribute to guide evolution in the vast solution space, i.e. toward where the target developmental trajectory is more likely to emerge.

3.1 Cellular Developmental Model

The developmental model used herein is based on cellular automata, i.e. synchronized cellular updates, parallel operation and discrete cell states. As such, the totality of regulative inputs can be coded completely in the genome (this does not imply that all of the genome information is expressed in the phenotype). To be able to have a complete regulatory network for all possible input states the model needs to be minimalistic. However, some features are not reduced to the minimum. The number of cell states is set to three instead of two. This was done to keep the concept of multi-cellular organism and cells differentiation, i.e. two types of cells in addition to the cells that are defined to be dead (void/quiescent). To be able to keep the principle of a growing (expanding) organism there is a constraint on how a cell can come "alive". This constrain is to only allow cells that have at least one neighbor expressing a cell type different from void to be able to come alive. The organisms develop in a two dimensional grid world, starting from a single cell placed in the grid (zygote). The placement of the first cell is of no importance as the grid uses cyclic boundary conditions. The local cellular communication is based on von-Neumann neighborhood (5 neighbors) and includes only cell type information, i.e. no environmental influence. With three cell types multicellularity is possible and at the same time the number of all possible cellular states in the defined neighborhood is not extremely large, i.e. max 243 (or 3^5). A developing organism will consist of different construct of these three cells. A more detailed description on the developmental model is given in [7, 8].

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. 1. Genome developmental table: regulatory input and cellular actions.

The table in Figure 1 is a scaled down illustration of all possible regulatory combinations. For the first entry in the table, i.e. all regulatory inputs set to 0, the output of the development process is fixed at 0. This is done to fulfill the stated constraint related to growth. All other regulatory inputs have a possibility of regulating the cell to be at any of the available cell types, indicated by the triplet *{0, 1, 2}*.

Figure 2 shows an example of a developing organism. At Development Step (DS) 0 the organism consists of only a single cell of type 1 (the zygote), at DS 1 the first cell has divided and differentiated into three cells of type 2. At DS 2 – DS 4 the change in phenotypic structure along the developmental path can be observed. The last shown organism is at DS 2000000.

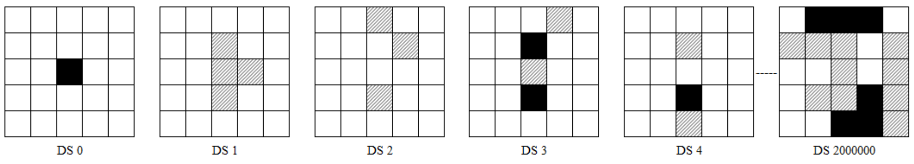

Fig. 2. Example of developing organism at intermediate development steps.

3.2 Quantification of Phenotypes

Having defined genetic information for the cellular model, a quantifiable measure has to be identified for the developed organisms. Properties that can be used need to provide information on the developing organism as a whole and the occurring phenotypic changes [11]. For a given organism, the initial cell (zygote) follows a developmental path and after a transient phase reaches an attractor, i.e. a final stable state or a self-reorganizing cycle. A complete trajectory identifies the whole lifecycle of the organism [6, 8] together with morphology, size and behavior changes. Trajectory length is an abstract measure that does not code for any computational task, e.g. majority, synchronization or a given phenotypic structure, since moving from node to node in a trajectory is the computation. For the scope of this research, no specific problem is implied and generalization is crucial. As such, trajectory length is the chosen metric of phenotypic complexity.

3.3 Evolution of Genome Information

In the model described earlier, the gene regulatory information is composed by the cell state of the five cells in the neighborhood. The evolvable information is then represented by the column C(t+1) in Figure 1, which describes the outcome of the gene regulation process. Moreover, such explicit representation of all possible cellular actions opens the possibility to identify sub-groups of developmental rules (sub-transitions):

- Growth rules: sub-transitions that represent a void cell (type 0) becoming alive (type 1 or type 2);
- Differentiation rules: one of the alive cells switches to the other alive cell type;
- Death rules: one of the alive cell types becomes void;
- No-change rules: the cell does not change its state.

Death sub-transitions are a special case of differentiation rules where alive cells differentiate to quiescent cells. This group of transitions is also used to calculate Lambda parameter, as described in the following section.

The developmental trajectory from zygote to multi-cellular organism can be represented by the state transition in Figure 2. Such trajectory produces a genome activation pattern at each development step that can be measured in terms of sub-transitions activated in the genome. For example, from DS 0 to DS 1 only one differentiation sub-transition is activated, together with three growth sub-transitions and twenty one no-change sub-transitions. No death sub-transitions are triggered.

4 Lambda Genome Parameter

Parameters obtained from the genome information can be used to estimate the dynamic developmental behavior of the emerging organisms. Langton [6] tried to find a relation between CA behavior and a parameter λ. He observed that the basic functions required for computation (transmission, storage and modification of information) are more likely to be achieved in the vicinity of phase transitions between ordered and disordered dynamics (edge of chaos). He hypothesized that it is easier to find rules capable of complex computation in a region where the value of λ is critical. Since the developmental model is composed by 3^5 regulatory combinations, all the possible regulatory inputs and relative outputs (growth, differentiation, death or no action) are fully specified in the developmental table. In order to calculate λ, it is necessary to define a quiescent state, the void cell (type 0) in our case. Lambda is defined as follows:

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

λ can be calculated according to Equation 1, where n represents the number of transitions to the quiescent (dead) state, K is the number of cells types (three in our case) and N is the neighborhood size (five in the Von Neumann neighborhood). In this way, the value of λ is based only on local properties of the neighborhood and in particular the cellular actions that are present in every cell.

Previous works [6, 7] have shown a clear relation between Lambda parameter and developed organisms' trajectory length. In particular, it is possible to identify a parameter space interval where organisms are more likely to have long life cycles. As such, Lambda (or other genome parameters) could be used to guide evolution when the target fitness is based on organisms' developing trajectories, as the work herein.

As shown in Equation 1, Lambda is determined by the ratio of sub-transitions in the developmental table that lead to the quiescent state over the total number of sub-transitions in the regulatory table. As such, λ takes into account only developmental rules that describe a cell death, i.e. one of the alive cell types becomes void. In other words, it does not consider other sub-transition groups (growth, differentiation, no-change). Lambda can be considered a single sub-transition genome parameter.

4.1 Genome Parameter Sub-Transitions

Lambda parameter has been shown to be able to differentiate different developmental behaviors (transient, attractor and trajectory length) for boolean CAs [6, 9, 10] and with organisms with 3 cell types [7, 8]. When the number of possible cell states gets bigger, Lambda may not be able to capture genome properties related to transitions to the chosen quiescent states. This is due to the presence of a growing number of sub-transition classes. Lambda's meaning would then be loose and be interpreted just as one of the many sub-transition classes in the genome table. On the other hand, there exist many sub-transitions parameters that hold the same parameter distribution as Lambda. Such sub-transition parameters could be then composed to create a custom parameter that captures the same genome properties as Lambda does.

5 Experimental Setup

In the experiment herein the presented developmental model with organism size of 4x4 cells is used. This leads to a theoretical maximum trajectory length and attractor length of 3^{16} development steps.

The genetic algorithm uses a population of 24 genotypes with elitism. The 8 worst individuals are replaced in each generation by newly generated offspring, selected through proportionate selection. Each two selected genotypes undergo one-point uniform crossover with probability 0.7 and mutation with probability 0.02 per gene. The genotype initial population is initialized with void genomes, i.e. all the transitions in the developmental table lead to the quiescent state. This means that the resulting phenotypes are the most unfit and difficult to evolve, i.e. dead organisms that end-up in a point attractor after a single development step. This is done to provide an even starting point for comparison.

In the standard GA (used as reference), the fitness is proportional to the developed trajectory length (the longer the fitter). In the GA that uses Lambda genome parameter contribution in the fitness function, the combined fitness (CFitness) is calculated as follows:

$$CFitness = Fitness + Fitness \times \frac{Abs(HiLambda - Lambda)}{HiLambda} \times ratio \tag{2}$$

In Equation 2, the used ratio is 0.2 (1) and HiLambda represents the Lambda value where the longest trajectory length is more likely to be found (0.66 in our model), i.e. critical Lambda [6].

6 Genome Parameter to Guide Evolution

In this first experiment, performances of a conventional genetic algorithm are compared to a GA that encapsulates Lambda in the fitness function. The chosen trajectory length targets are set as 1000, 5000, 10000 and 15000 development steps (average over 1000 runs for target 1000, average over 20 runs for the other targets due to runtime). Results are shown in Figures from 3 to 6 respectively. In all the four consid-

1 The ratio that gave best results for GA with λ in fitness from randomized genomes is 0.05. Otherwise 0.2 is used in all the plotted results.

ered cases, the effect of Lambda in fitness is clearly visible.

In Figure 3 the target trajectory length was set as 1000 development steps. The conventional GA performs better than the one with Lambda contribution in the fitness function for few generations. From generation 17 the effects of Lambda driving evolution are more evident and the algorithm converges faster toward the target.

GA	Void genomes (plotted)		Randomized genomes	
Reference GA	443,60	-	372,00	-
Lambda fitness	365,62	-17,57%	351,33 (1)	-5,56%

Table 1. Comparison of reference GA and GA with Lambda contribution in the fitness function. Target trajectory length is 1000 development steps. Avg over 1000 runs.

Table 1 shows numerical results when trajectory length target is 1000 development steps. This is analyzed from two different initial conditions: void initialized genomes (all the transitions in the developmental table lead to the quiescent state and the organism are the most unfit, plotted in Figure 3) and randomly initialized genomes (not plotted here). From void genomes the reference GA needs 443,6 generations on average (over 1000 runs), whether Lambda parameter in the fitness function is 17,57% faster (unpaired 2-tail t-test, p<0,0001). From randomized genomes the latter is still 5,56% faster, finding solutions in 351,33 generations on average compared to 372 generations needed by the conventional GA (it is important to mention that the GA with Lambda in fitness with void initialization is even faster than the reference GA with randomized initialization).

The same trend is shown in Figure 4, where the target was longer. Here the difference in convergence speed is more accentuated. Results in Figure 5 and 6 confirm that there is a point where the two lines cross each other. After that specific generation the algorithm with Lambda contribution converges faster than the conventional approach. It is clear that the ability of Lambda to detect longer trajectories in certain areas of the parameter space is beneficial when trajectory length is the target behavior. In all the presented scenarios, both approaches show an asymptotic tail towards 0 (minimum distance from target fitness). The difference is in the speed of convergence to the asymptotic target distance. Lambda in the fitness function is promising.

As a side experiment, λ was used outside the fitness function with a conventional GA. In such experiment λ was used as a discard parameter where genomes were discarded after selection if the parameter value was not matching a defined interval of acceptance. Here the results were not promising and in some cases the system was not evolvable.

7 Genotype Sub-Transitions

Lambda genome parameter measures sub-transitions in the genome developmental table that lead to the quiescent state. Other sub-transitions are present in the genome

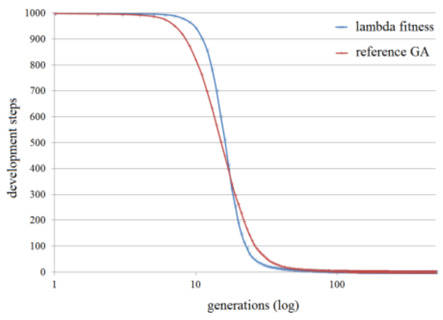

Fig. 3. Comparison of reference GA and GA with Lambda contribution in the fitness function. Target trajectory length is 1000 development steps. Distance from target (y) over generations (x). Avg over 1000 runs.

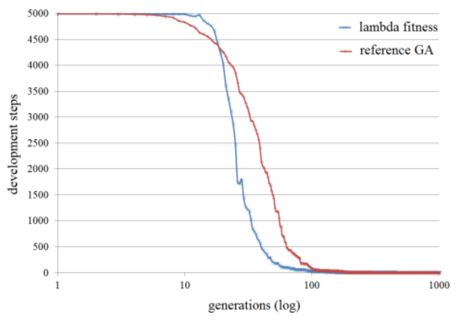

Fig. 4. Comparison of reference GA and GA with Lambda contribution in the fitness function. Target trajectory length is 5000 development steps. Distance from target (y) over generations (x). Avg over 20 runs.

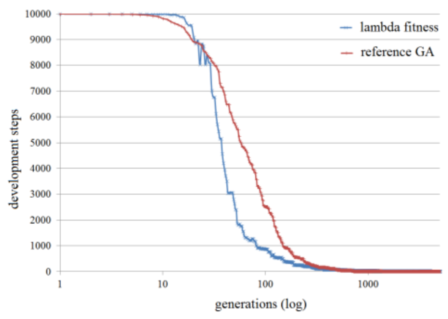

Fig. 5. Comparison of reference GA and GA with Lambda contribution in the fitness function. Target trajectory length is 10000 development steps. Distance from target (y) over generations (x). Avg over 20 runs.

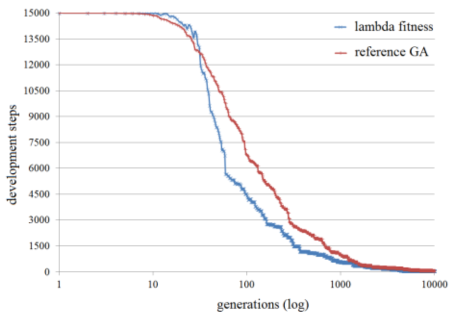

Fig. 6. Comparison of reference GA and GA with Lambda contribution in the fitness function. Target trajectory length is 15000 development steps. Distance from target (y) over generations (x). Avg over 20 runs.

table, i.e. growth, differentiation and no-change. Here it is investigated if other sub-transition classes could replace Lambda in forecasting the emergent behavior, thus being able to be used in multi-cellular developmental systems with more cell types. In such case, Lambda may represent too few genotype properties, thus not being able to drive evolution. Moreover, several sub-transitions could be used together to compose custom parameterizations of the rule-space.

Figure 7 shows an example of sub-transition classes' activation patterns for a specific developed organism, successfully evolved in the previous set of experiments. Here each line represents one of the sub-transitions (growth in red, death in blue, differentiation in green and no-change in purple). The plot shows the number of cells that trigger a specific sub-transitions class in each development step. Pattern repetitions may indicate that an attractor has been reached or that same pattern repetition happens in different areas of the grid world where the organism is developed. This may indicate self-regulation based on topologic properties. The top-line (purple) shows activation of no-change sub-transitions, which is often the most used sub-class, whether growth (red) and death (blue) are often overlapping and seem to have a similar trend.

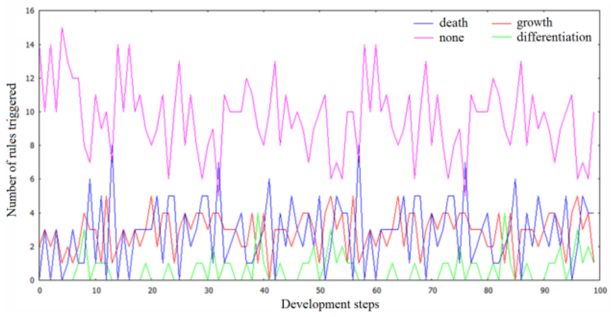

Fig. 7. Sub-transitions (growth, death, differentiation, no-change) activation pattern (y) over the generations (x) for the given example organism.

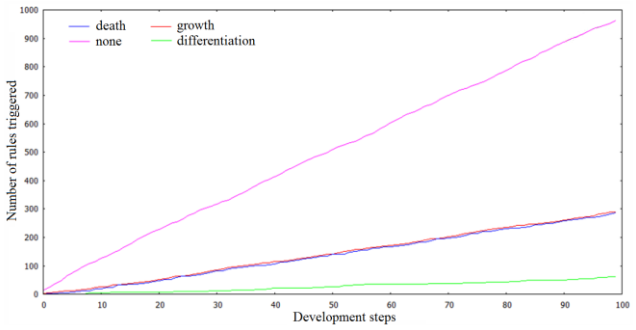

Fig. 8. Cumulative sum of sub-transitions activation pattern for the given example organism.

Figure 8 plots the cumulative sub-transition usage during development, for the same given organism. It is clear that growth rules and death rules have to be balanced. This was observed with target trajectories of different lengths. Differentiation rules and no-change rules did not show any specific trend.

Figure 9 plots developed trajectory lengths for 100000 randomly generated genotypes. As such, the distribution of transition rules in the genotype is scrambled and distributed in the rule sub-transition space. In Figure 9 only death sub-transitions are considered. The plot shows a similar distribution to Lambda parameter, since the death sub-transitions capture the same genome properties as Lambda. Figure 10 plots the same organisms when the considered genome parameter is differentiation sub-transitions. Here the relation between the considered sub-parameter and trajectory length is weaker and organisms are less concentrated around the critical parameter value. Same results were obtained for other sub-transition groups.

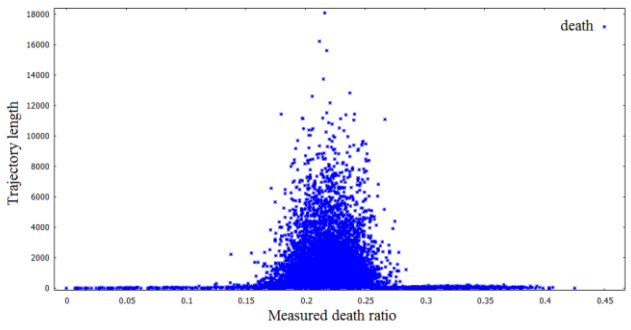

Fig. 9. Death sub-transition parameter distribution (x) and resulting trajectory length measured as #development steps (y). 100000 organisms.

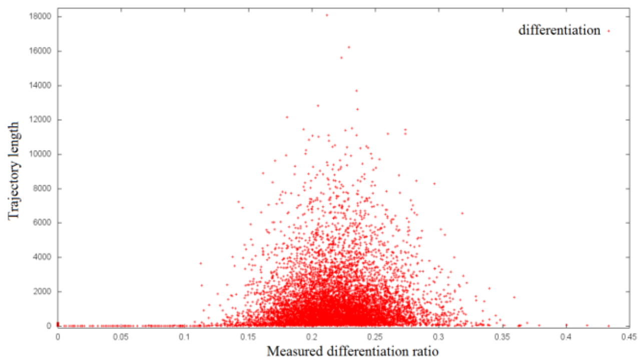

Fig. 10. Differentiation sub-transition parameter distribution (x) and resulting trajectory length measured as #development steps (y). 100000 organisms.

In Figure 11 a composed sub-parameter is considered, namely the difference between number of death and growth sub-transitions. It is evident that when growth and death rules are balanced it is possible to develop long trajectories (as the target in our experimental work). On the other hand, when those sub-transitions are not balanced, the organism produces a very short trajectory cycle. This information may be important at the design stage of an EvoDevo system, when Lambda is not able to characterize the behavioral regime due to a larger number of available cell types. In such case, Lambda would be no more than one out of the many sub-transition groups, thus lacking a clear relation between trajectory length and the parameter space.

Finally, Figure 12 shows a zoomed-in plot of the same 100000 generated organisms in Figure 9, 10 and 11, where genomes with unbalanced death-growth difference develop short trajectories. In contrast, balanced rule-sets develop longer trajectories, being able to filter-out unfit genomes. That was not possible if only a single sub-transition class was considered. It is important to highlight that Lambda is a single-transition parameter.

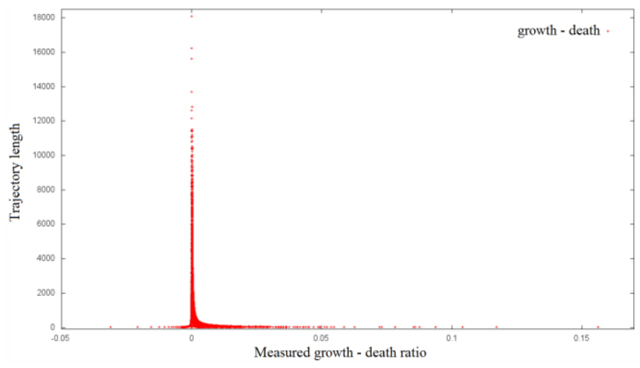

Fig. 11. Growth-Death sub-transition parameter distribution (x) and resulting trajectory length measured as #development steps (y). 100000 organisms.

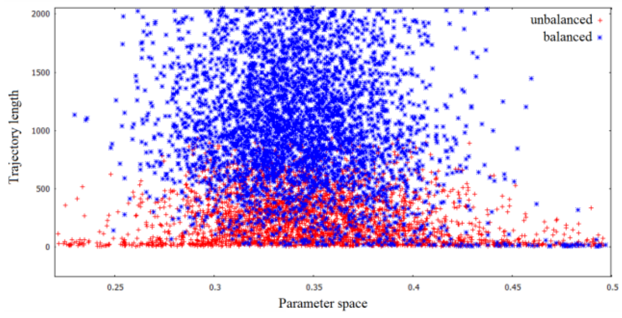

Fig. 12. Genomes unfiltered vs. genome filtered with growth-death sub-transition parameter.

8 Conclusion

The presented experiments investigated how to exploit genotype information of artificial cellular organism to guide evolution in favorable areas of the solution space, where the sought phenotypic behavior is more likely to be found. Genome information has been used to calculate Lambda genome parameter. Such parameter was incorporated into the fitness function to speed-up convergence to the target trajectory, as shown in the plots in Figure 3, 4, 5 and 6, for different target trajectory lengths. Previous work [8] has shown that other parameters besides Lambda, e.g. Sensitivity [14], Mean Field Parameters [15], have similar abilities to forecast emergent behaviors. Thus, it may be interesting to extend the investigation and compare results obtained with other parameters.

The used genome representation allowed identifying genome sub-transitions other than those used to calculate Lambda (transitions to the quiescent state). The identified sub-transition groups (growth, differentiation, death, no-change) can be considered single transition parameters and thus used as Lambda. They can also be composed together to produce a multiple-transition genome parameter when Lambda may not be able to characterize the phenotype behavior due to increased number of cell types. In particular, death-growth transition difference has been shown to be well suited to identify artificial organisms that produce long trajectory, as in Figure 11, and filter out organisms with short trajectories, as shown in Figure 12. It may be interesting to investigate sub-transitions' potential the same way as Lambda was used here.

The approach used herein shows that exploiting genome information during evolution could increase the evolvability of the system, when there is an indirect genotype to phenotype mapping and fitness is a measure of phenotypic properties.

As a future work, it may be possible to investigate the robustness of solutions evolved with a fitness measure based on both phenotype and genotype information. In particular, how fragile evolved organisms 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. A. Wagner, "Robustness and evolvability: a paradox resolved", Proceedings of the Royal Society B - Biological Sciences 275, no. 1630 (2008): p. 91-100.
2. Y. Bar-Yam, "Dynamics of complex systems", Studies in Nonlinearity, Westview Press (1997): p. 864.
3. J. F. Miller, "Evolving developmental programs for adaptation, morphogenesis, and self-repair", in 7th European Conference on Artificial Life, Lecture Notes in Artificial Intelligence, Springer (2003): p. 256-265.
4. G. Tufte and P. C. Haddow, "Towards development on a silicon-based cellular computation machine", Natural Computation (2005), 4 (4): p. 387–416.
5. F. Glover and G.A. Kochenberg, "Handbook of metaheuristics", International Series on Operations Research and Management Science, Springer (2003): p. 570.
6. C. G. Langton, "Computation at the edge of chaos: phase transitions and emergant computation", In S. Forrest, editor, Emergent Computation, MIT Press (1991): p. 12–37.
7. G. Tufte and S. Nichele, "On the correlations between developmental diversity and genomic composition", 13th Annual Genetic and Evolutionary Computation Conference, GECCO 2011, ACM (2011): p. 1507-1514.
8. 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, Springer (2012): p. 186-197.
9. 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 (2001), Artificial Life 7: p. 277-301.
10. 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).
11. T. Kowaliw, "Measures of complexity for artificial embryogeny", GECCO 2008, ACM (2008): p. 843-850.
12. F. Rothlauf, "Locality, distance distortion, and binary representations of integers". Working Paper 11/2003. University of Mannheim.
13. T. D. Pollard, "No question about exciting questions in cell biology". PLoS Biol 11(12): e1001734 (2013). doi:10.1371/journal.pbio.1001734.
14. P. M. Binder, "Parametric ordering of complex systems". Physical Review E, vol. 49 n. 3 (1994), p. 2023-2025.
15. W. Li, "Phenomenology of nonlocal cellular automata". Santa Fe Institute (1992). Journal of Statistical Physics, 68(5-6): p. 829-882.