Measuring Phenotypic Structural Complexity of Artificial Cellular Organisms

Approximation of Kolmogorov Complexity with Lempel-Ziv Compression

Stefano Nichele and Gunnar Tufte

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. Artificial multi-cellular organisms develop from a single zygote to different structures and shapes, some simple, some complex. Such phenotypic structural complexity is the result of morphogenesis, where cells grow and differentiate according to the information encoded in the genome. In this paper we investigate the structural complexity of artificial cellular organisms at phenotypic level, in order to understand if genome information could be used to predict the emergent structural complexity. Our measure of structural complexity is based on the theory of Kolmogorov complexity and approximations. We relate the Lambda parameter, with its ability to detect different behavioral regimes, to the calculated structural complexity. It is shown that the easily computable Lempel-Ziv complexity approximation has a good ability to discriminate emergent structural complexity, thus providing a measurement that can be related to a genome parameter for estimation of the developed organism's phenotypic complexity. The experimental model used herein is based on 1D, 2D and 3D Cellular Automata.

Keywords: Developmental Systems; Emergence; Structural Complexity; CAs.

1 Introduction

Artificial developmental systems take inspiration from biological development, where a unicellular organism, i.e. a zygote, develops to a multi-cellular organism by following the instructions encoded in its genome. The genome contains the building instructions and not a description of what the organism will look like. Several artificial developmental systems take inspiration from cellular models [2-4], where the construct and constructor element is a cell. Thus, each cell in the system is a building block of the system and encapsulates the genome information that regulates the cellular actions, e.g. growth, differentiation, apoptosis. The emergent phenotype can result in a very simple or extremely complex structure. Our goal is to measure the complexity of the phenotypic structure and relate it to the genome information. The notion of structural complexity used is based on the theory of Komogorov complexity [5, 8]. As stated by the Incomputability Theorem (proof in [5]), Kolmogorov complexity is incomputable. Compression algorithms are often used as an approximation of the Kolmogorov complexity [11, 20]. In the experimental work herein, Lempel-Ziv algorithm is used to estimate the phonotypic structural complexity. In the genotype space, genomes are characterized and described by the Lambda genome parameter [6]. Such parameter has shown interesting abilities to discriminate genotypes in different behav-

ioral classes, e.g. fixed, chaotic, random [7]. It is thus investigated if λ is useful to relate genotype composition to the structural complexity of the emergent phenotypes.

The article is laid out as follows: Section 2 presents background and motivation. In Section 3 cellular automata are formally defined. Section 4 describes measures of structural complexity and the notion of Kolmogorov complexity and approximations. Section 5 introduces the developmental model and Lambda genome parameter. Section 6 describes the experimental setup and results. Section 7 includes analysis and discussion and Section 8 concludes the work.

2 Background and Motivation

In the field of Artificial Embryogeny, the goal is often to exploit emergent complexity out of the parallel local interactions of a myriad of simple components. To evaluate such systems' ability, a clear notion of complexity is necessary. Consider the notion of "edge of chaos" [6], a critical region of a parameter space where the system is between order and randomness. In ordered regimes there are only a few distinct possible configurations whether with total randomness the system exhibits the same statistical distribution of behaviors for any initial condition. Therefore, it is in the edge of chaos that systems exhibit high complexity to support advanced features favorable to perform computation. Such emergent complexity, if meaningfully measured, could be predicted using genome parameters. Langton [6] introduced the Lambda parameter to differentiate behavioral regimes where different levels of complexity could emerge.

A developmental mapping may be represented by a function that maps elements in the genotype space with elements in the phenotype space. Such mapping 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. In practice, small mutations can have a huge impact on the emergent phenotype. Therefore, a genome parameter predicting phenotypic behavior is useful as a guidance tool to keep resulting phenotypes within a complexity regime, reducing phenotypic difference as long as the genome parameter is kept within defined bounds.

3 Cellular Automata

Cellular automata (CA), originally studied by Ulam [18] and von Neumann [19] in the 1940s, are idealized versions of parallel and decentralized computing systems, based on a myriad of small and unreliable components called cells. Even if a single cell itself can do very little, the emergent behavior of the whole system is capable to obtain complex dynamics. In cellular computing each cell can only communicate with a few other cells, most or all of which are physically close (neighbors). One implication of this principle is that there is no central controller; no one cell has a global view of the entire system. The metaphor with biology can be exploited on cellular systems because the physical structure is similar to the biological multi-cellular organisms.

Formally, a cellular automaton is a countable discrete array of cells i with a discrete-time update rule Φ that executes in parallel on local neighborhoods of a specified radius r. In every time step the cells allow values in a finite alphabet A of symbols: $\sigma^i_t \in \{0, 1, ..., k\text{-}1\} \equiv A$. The local update rule is $\sigma^i_{t+1} = \Phi(\sigma^{i-r}_t, ..., \sigma^{i+r}_t)$. At

time t, the state s_t of the cellular automaton is the configuration of the finite or infinite spatial array: $s_t \in A^N$, where A^N is the set of all possible cell value permutations on a lattice of size N. The CA global update rule $\Phi: A^N \rightarrow A^N$ executes Φ in parallel to all sites in the lattice: $s_t = \Phi\, s_{t-1}$. For finite N, the boundary cells are usually dealt with by having the whole lattice wrap around into a torus, thus boundary cells are connected to "adjacent" cells on the opposite boundary. In this paper, 1D, 2D and 3D cellular automata with cyclic boundary conditions are considered.

4 Measuring Structural Complexity

Several complexity measures are proposed in literature, both to quantify genotype and phenotype complexity, e.g. [17]. For genotypes, size may not be an important factor. Even in nature, some unicellular eukaryotic organisms have much larger genomes than humans. Another possibility is to evaluate genotype complexity based on the number of activated genes. Such an activity measure may strongly relate on initial conditions, resulting in a non-precise complexity measure. However, emergent complexity appears at the phenotype level. Important factors can be related to cell organization or functions that the organism is able to perform. Within such an approach Kolmogorov complexity complies well to be able to capture such features.

4.1 Kolmogorov Complexity

The notion of complexity is used differently in distinct fields of computer science. Kolmogorov complexity could be used for understanding emergent complexity in artificial developmental systems.

Let us consider the following strings representing two different states of a 1 dimensional cellular automaton of size 20 at time step t:

a = *"01010101010101010101"* $\qquad$ b = *"01234567894978253167"*

We can intuitively see that string b is more complex than string a. String a is just a repetition of *"01"* whether string b does not seem to show any repeating pattern, i.e. string a is less complex because we can represent it with a shorter description than for string b. Kolmogorov complexity represents the length of the shortest description of a string. In his work, Kolmogorov made use of a Universal Turing Machine to define complexity in an unambiguous way.

Definition *(Kolmogorov complexity)*: Fix a Turing Machine U. We define the Kolmogorov function, $C(x)$ as the length of the smallest program generating x. This is shown in Equation (1).

$$C(x) = min_p \{|p| : U(p) = x\} \tag{1}$$

It is proven by the *Invariance Theorem* [5] that the particular choice of the universal machine only affects $C(x)$ by a constant additive factor and in particular, $\forall x, C(x) \leq |x| + c$. Kolmogorov complexity is incomputable in theory and thus, some approximations are needed.

4.2 Incomputability Theorem

If the problem of computing the Kolmogorov complexity of a string x is to be handled, the way to proceed is to run all the programs which compute x as output and then find the shortest among them, thus testing all the possible programs. Unfortunately, there is no way of knowing if a program halts or not, hence the undecidability of the halting problem [9] implies the incomputability of Kolmogorov complexity. Fortunately, in practice we are not interested in the exact value of the Kolmogorov complexity. Data compression algorithms could be used, to some extent, to approximate it. In fact, strings that are hardly compressible have a presumably high Kolmogorov complexity. Complexity is then proportional to the compression ratio. As stated earlier, the Kolmogorov complexity of a string x is always less than or equal to the length of the string x itself plus a small constant: $C(x) \leq |x| + O(1)$. Yet, as proven by the *Incompressiblity Lemma* [5], there are some strings that are not compressible, i.e. random strings. Formally, a string x is c-incompressible if $C(x) \geq |x| - c$.

4.3 Lempel-Ziv Compression Algorithm

Compression algorithms have been widely used as approximations of Kolmogorov complexity. For example, Lehre, Hartmann and Haddow [10-11], successfully computed approximations of Kolmogorov complexity as measures of genotype and phenotype complexity, using Lempel-Ziv compression algorithm. Zenil and Villareal-Zapata [20] studied one-dimensional cellular automata rules' behavior using approximations of Kolmogorov Complexity. Compression algorithms tend to compress repeated patterns and structures, thus being able to detect structural features in phenotype states. In the experiments herein, we use Deflate [12] algorithm, which is a variation of LZ77 [13]. Deflate is a loseless data compression algorithm that combines LZ77 and Huffman coding [14]. This choice is based on the fact that Deflate is a computationally inexpensive operation and, as long as the state compression process is precisely defined, it is independent of the dimensionality of the state.

If 1D cellular automaton is considered, the correspondent string representing the state of the system at a certain time step could be compressed directly. For a 2D cellular automaton of size 3 by 3, as an example, single rows are concatenated together to compose the state string r:

0	1	0
1	1	2
1	0	0

$\rightarrow$ *r = "010112100"*

The same procedure is applied for a 3D cellular automaton, where all the rows are listed for all the depth levels. Such measure is dimensionality-independent, since it can be used for 1D, 2D or 3D cellular automata. The state string r can now be compressed using the Deflate algorithm, which produces the compressed state string t.

$$t = Deflate\ (r)$$

The next step would be to calculate the length q of the compressed string t.

$$q = Length\ (t)$$

q can then be used to compare the approximate complexity of the states. However, the value has to be normalized in order to compare complexities for different dimensionalities and grid sizes. It is necessary to find lower and upper bounds for the compressed string length, in order to scale the value of q. To do that, it is possible to consider the least and the most complex states. Again, for a 3 by 3 CA, the bounds are:

$$r_{min} = \text{"000000000"} \qquad r_{max} = \text{"012345678"}$$

r_{min} yields the lowest compressed size q_{min} for states of the given dimension and size. Likewise, r_{max} which has no identical symbols, yields the highest compressed size q_{max}:

$$q_{min} = Length(Deflate(r_{min})) \qquad q_{max} = Length(Deflate(r_{max}))$$

The normalized structural complexity measure c of the state s is then:

$$c = (q - q_{min}) / (q_{max} - q_{min})$$

One last remark on the structural complexity calculation is about position and orientation of structures in a state. In fact, it may be better if such measure is transformation invariant. States that represent the same structure, only in a different orientation or position in the grid, should have equal structural complexity. Let us consider the following example:

0	0	0		2	0	0
0	0	0	→	1	0	0
2	1	2	T	2	0	0

In such case, it is evident that the state on the left has equivalent structure to the state on the right, since the transformation T rotates the state 90 degrees. Even though, the measured structural complexity of the row-concatenated-state is different. As such, the following measures are evaluated, as specified in Table 1:

Table 1.

1	Simple Deflate compression	The CA state is represented as a concatenated string and directly compressed.
2	Average of all rotations	The CA state is rotated in all the possible orientations and the correspondent state strings are compressed. The average is computed.
3	Average of all translations	The CA state is shifted in all the possible positions and the correspondent state strings are compressed. The average is computed.
4	Rotations + translations	Both point 2 and 3. The CA state is rotated in all the possible orientations. Each of them is shifted in all the possible positions and the correspondent state strings are compressed. The overall average is computed.

5 Cellular Developmental Model

The developmental model used in this work is an embryomorphic system [1] based on cellular automata, where the goal is the self-assembly of cells from a single zygote which holds the complete genotype information. A CA can be considered as a developing organism, where the genome specification and the gene regulation information control the cells' growth, differentiation and apoptosis. The global emergent behavior

of the system is then represented by the emerging phenotype, which is subject to size, shape and structure modifications along the developmental process.

The experimental work is conducted on cellular automata with different dimensionalities, 1 – 3, and neighborhood configurations, 3 – 7.

All CAs have cyclic boundary conditions. Each cell has 3 possible states (cell type 0: empty/dead cell, cell type 1 and cell type 2). The grid is initialized with a cell of type 1 (zygote) in the middle of the grid and develops according to a genotype based on a cellular developmental table that fully specifies all the possible regulatory input combinations, i.e. all 3^n neighborhood configurations are explicitly represented (n represents the neighborhood size). To ensure that cells will not materialize where there are no other cells around, a restriction has been set in the developmental table: if all the neighbors of an empty cell are empty, the cell will be empty also in the following development step. A more detailed description of the developmental model is given in [15-16].

During the development process, a unicellular organism grows to a multi-cellular organism. Two different life phases are identified: the transient phase and the attractor. The transient phase begins with the initial state of the CA (zygote) and ends when the organism reaches its adult form and an attractor begins. Note that this definition is not biologically correct. The attractor represents the time lapse between two repetitions of the same state, i.e. the same state is encountered twice. A complete trajectory is then defined as the sum of transient phase and attractor.

The Lambda Genome Parameter obtained from the genome information can be used to estimate the dynamic behavior of the system and thus can be related to the emergent complexity of the phenotype. Langton [6] studied the parameter λ as a measure of the activity level of the system. λ has shown to be particularly well suited to discriminate genotypes that will develop phenotypes in different behavioral classes, e.g. fixed, chaotic, random [7]. Lambda is calculated out of the regulative outcome of the developmental table, i.e. the output at time $t+1$ based on a specific neighborhood configuration at time t. According to Langton's definition, a quiescent state must be chosen. We choose the empty cell (type 0) as the quiescent state. λ is then calculated according to Equation 2, where n represents the number of transitions to the quiescent state, K is the number of cell types (three in our case) and N is the neighborhood size, as defined in Table 2.

$$\lambda = \frac{K^N - n}{K^N} \tag{2}$$

Langton 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 genotypes capable of complex computation in a region where the value of λ is critical.

In the experiments herein, genomes are generated in the whole Lambda spectrum, from 0 to 1, using a similar method to Langton's random table method [6], i.e. for every entry in the developmental table, with probability $(1-\lambda)$ the cell type at the next developmental step is quiescent (type 0); with probability (λ), the cell type at the next

developmental step is generated by a uniform random distribution among the other cell types (type 1 or 2).

Previous work [16] has shown that Lambda is able to discriminate genotypes that will end up with very long or extremely short trajectories and attractors. In this paper we investigate relationship between λ, as a genotype measurement, and emergent structural complexity of the corresponding phenotypes.

6 Experimental Setup

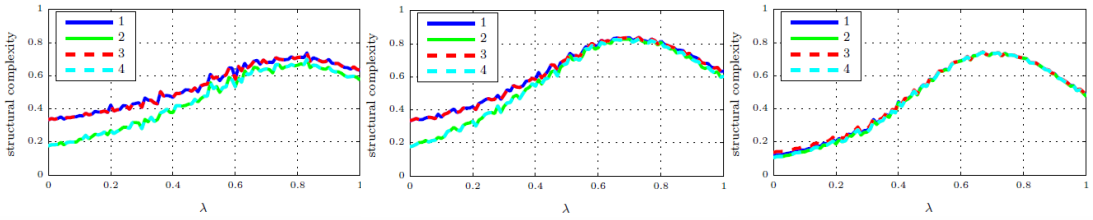
Two different sets of experiments are conducted. First, the four different complexity measures in Table 1 are tested. 100 developmental tables are generated for each λ value with granularity 0.01. The correspondent genotypes are developed starting from a single cell on different cellular architectures (1, 2 or 3D with 3, 5 or 7 neighbors), as specified in Table 2, experiment 1. The structural complexity is then measured for the whole trajectory and for the attractor.

Plots from Figure 1.1 to 1.7 show the results for each configuration in Table 2 – experiment 1, where the four lines represent the complexity measures in Table 1. The x-axes plots the whole Lambda spectrum whether the y-axes is the measured structural complexity.

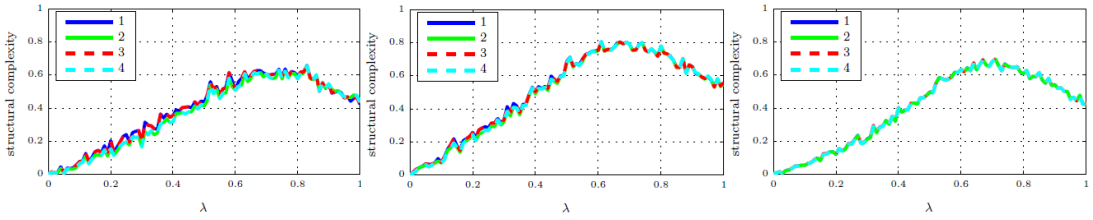
Table 2.

Dimensionality	***Size***	***Cells***	***Neighborhood radius***
Experiment 1:			
1D	9	9	3
1D	9	9	5
1D	16	16	5
1D	8	8	7
2D	3x3	9	5
2D	4x4	16	5
3D	2x2x2	8	7
Experiment 2:			
1D	25	25	3
1D	27	27	3
1D	25	25	5
1D	27	27	7
2D	5x5	25	5
3D	3x3x3	27	7

The results show that there is a clear relation between the genome parameter value and all the complexity measures, independently from the dimensionality, neighborhood configuration and grid size. Moreover, it is clear that such complexity measures, in relation to λ, are able to characterize both trajectory structural complexity and attractor structural complexity. In most of the cases, the four lines are almost always



(a) *Trajectory.*



(b) *Attractor.*

Figure 1.1: *1D 9 cells, 3-neighbourhood*

(a) *Trajectory.*

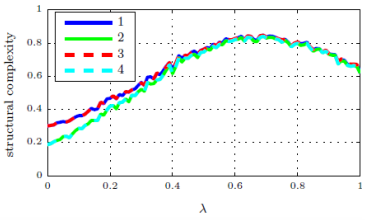
(b) *Attractor.*

Figure 1.2: *1D 9 cells, 5-neighbourhood*

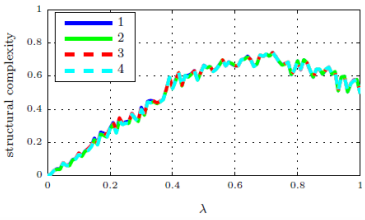
(a) *Trajectory.*

(b) *Attractor.*

Figure 1.3: *1D 16 cells, 5-neighbourhood*

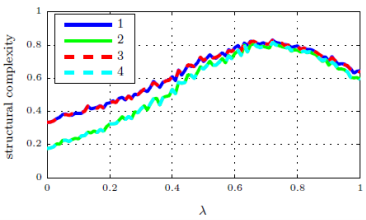


(a) *Trajectory.*

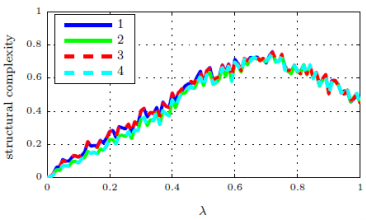


(b) *Attractor.*

Figure 1.4: *1D 8 cells, 7-neighbourhood*

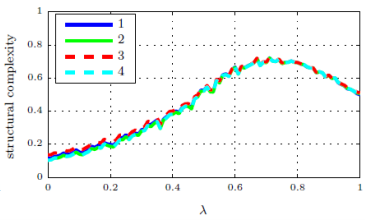


(a) *Trajectory.*

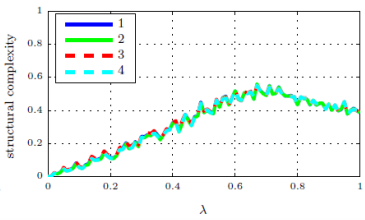


(b) *Attractor.*

Figure 1.5: *2D 9 cells (3×3), 5-neighbourhood*

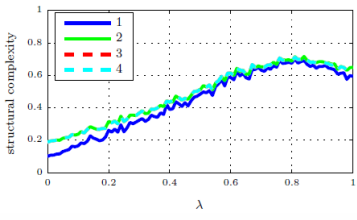


(a) *Trajectory.*

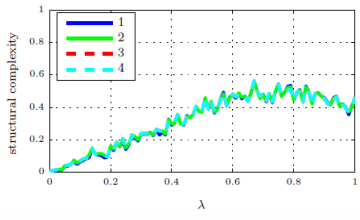


(b) *Attractor.*

Figure 1.6: *2D 16 cells (4×4), 5-neighbourhood*



(a) *Trajectory.*



(b) *Attractor.*

Figure 1.7: *3D 8 cells ($2 \times 2 \times 2$), 7-neighbourhood*

Fig. 1. Results for the first set of experiments defined in Table 2. Lambda on x and structural complexity on y. Lines 1, 2, 3 and 4 represent the measures in Table 1.

overlapping, except for some trajectories with λ value between 0 and 0.4, which represents the ordered behavioral regime. In conclusion, it is not necessary to perform expensive rotation and translation before the actual compression. Thus, in the remainder of the paper, the term structural complexity refers to the approximation of Kolmogorov complexity using Deflate as an implementation of Lempel-Ziv.

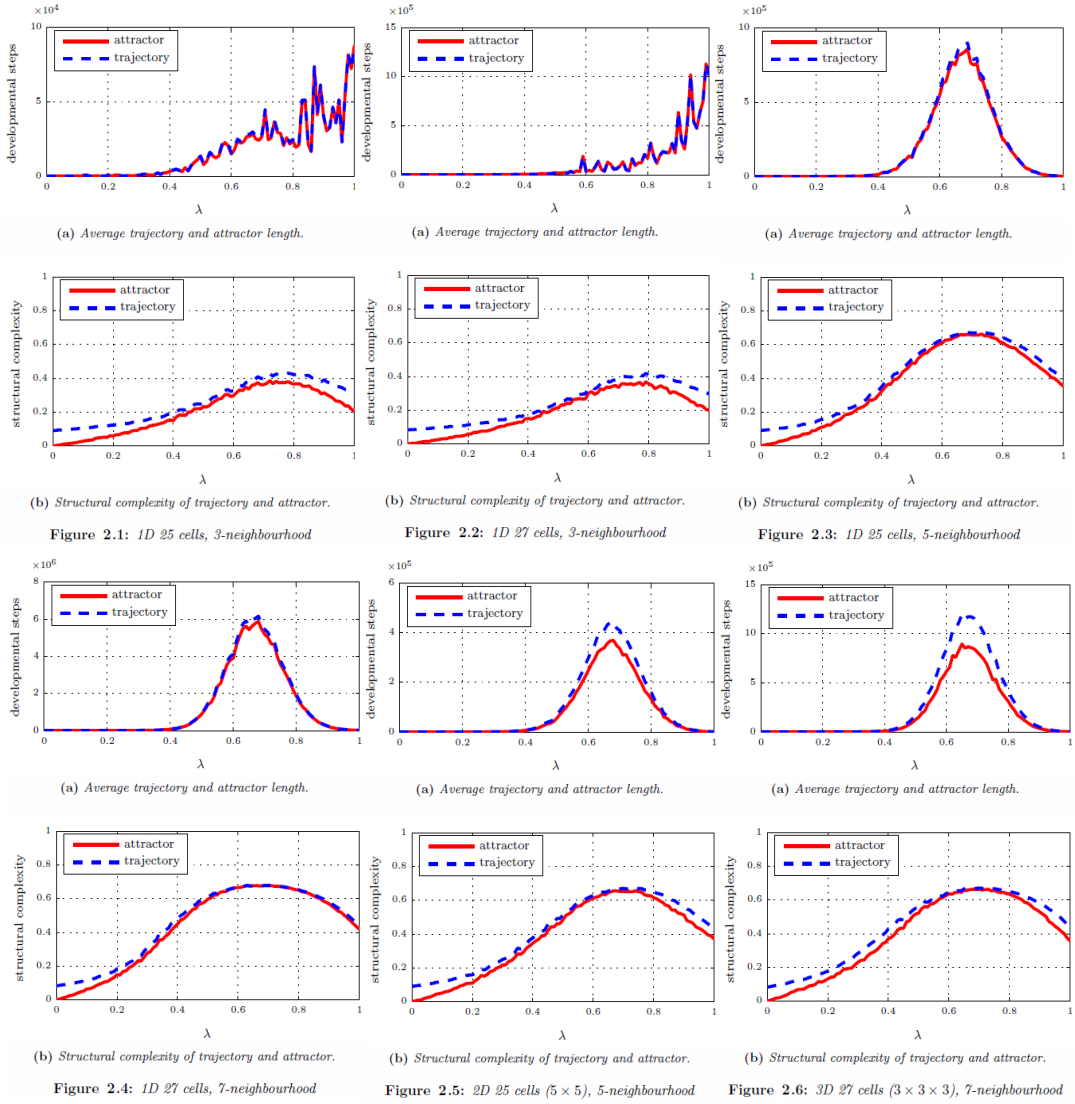
In the second set of experiments, more accurate tests are performed. 1000 developmental tables are generated for each λ value with granularity 0.01. The correspondent genotypes are again developed starting from a zygote on different cellular architectures as shown in Table 2, experiment 2. The structural complexity for trajectory and attractor is measured in relation with Lambda and compared with the correspondent trajectory and attractor length, measured as the number of development steps. Results are shown in Figure 2.1 to 2.6.

7 Discussion

The experiments presented show that the proposed measure of phenotypic structural complexity is able to capture emergent properties of artificial developmental systems. Figures 2.1(a) and 2.2(a) show consistent results with those obtained by Langton [6], where Lambda is not able to accurately describe the search space for 1D CA with rather small neighborhood radius and 3 cell types. Remarkably, the structural complexity describes well the parameter space, with low complexity when Lambda is close to 0 and higher complexity where λ reaches the critical value around 0.66. This can be observed in Figure 2.1(b) and 2.2(b). If only the plots (b) are analyzed, from 2.1(b) to 2.6(b), it is possible to spot that the structural complexity curve has the same shape for any configuration.

For 1D CA with small neighborhood the curve is flattened whether for 1D with bigger neighborhoods, 2D and 3D is wider. Overall, the maximum structural complexity that emerges is always slightly over 0.6, meaning that adding dimensions to the developing structure and keeping the total number of cells constant does not increase the structural complexity of the developed organisms. In that sense, 1D, 2D and 3D organisms with same size have the same relative potential to show complex structures. This is an interesting result if one considers adding a new dimension to an EvoDevo system to achieve higher structural complexity. Again, it is possible to observe this result comparing Figure 2.3(b) and 2.5(b) where the developing structures are 1D and 2D respectively, both with 25 cells and 5 neighbors. Same result for Figure 2.4(b) and 2.6(b), using 1D and 3D CA, both with 27 cells and 7 neighbors.

Comparing Figure 2.4(b) and 2.5(b), it is possible to observe that moving from a 1D CA to a 2D CA the parameter region with higher structural complexity is larger with a single dimension. In fact, the shape is more stretched and almost flat on the peak, whether with two dimensions becomes spikier. This seems to be an effect of the enlarged neighborhood. Looking carefully at plot in Figure 2.6(b), this effect caused by an increased neighborhood is still noticeable, even if mitigated by the addition of dimensions. The same behavior is not present in Figure 2.3(b), where development happens on a 1D automaton and structural complexity is analogous to 2D automaton with same neighborhood configuration and same number of cells, as represented in Figure 2.5(b).



(a) *Average trajectory and attractor length.*

(b) *Structural complexity of trajectory and attractor.*

Figure 2.1: *1D 25 cells, 3-neighbourhood*

(a) *Average trajectory and attractor length.*

(b) *Structural complexity of trajectory and attractor.*

Figure 2.2: *1D 27 cells, 3-neighbourhood*

(a) *Average trajectory and attractor length.*

(b) *Structural complexity of trajectory and attractor.*

Figure 2.3: *1D 25 cells, 5-neighbourhood*

(a) *Average trajectory and attractor length.*

(b) *Structural complexity of trajectory and attractor.*

Figure 2.4: *1D 27 cells, 7-neighbourhood*

(a) *Average trajectory and attractor length.*

(b) *Structural complexity of trajectory and attractor.*

Figure 2.5: *2D 25 cells* (5×5)*, 5-neighbourhood*

(a) *Average trajectory and attractor length.*

(b) *Structural complexity of trajectory and attractor.*

Figure 2.6: *3D 27 cells* ($3 \times 3 \times 3$)*, 7-neighbourhood*

Fig. 2. Results for the second set of experiments defined in Table 2.

Overall, it appears that extending the neighborhood setting results in a wider area where higher structural complexity is reachable. On the other hand, moving from a one-dimensional structure to a two or three-dimensional structure produces more sudden increases in structural complexity for parameter values between 0.3 and 0.5.

8 Conclusion

This paper investigated the emergent structural complexity of artificial cellular organisms at the phenotype level, using approximations of Kolmogorov complexity. Since

Kolmogorov complexity in not computable in theory, Deflate compression algorithm based on Lepmpel-Ziv has been used. Such complexity measure is well suited for understanding emergent properties of artificial developmental systems. In particular, it has been shown that structural complexity is strongly related to Lambda genome parameter and its ability to detect different behavioral regimes. This makes it possible to understand if genome information could be used to predict the emergent structural complexity of developing phenotypes. Moreover, the measurement we have used is dimensionality independent and has been experimented on 1D, 2D and 3D CA.

Another observed result is that structural complexity has shown to be powerful enough to characterize the parameter space even when the dimensionality, number of states per cell and neighborhood size are rather small. In such cases, it would not be possible to obtain predictions about trajectory and attractor length at the genotype stage, thus being uncertain about the emergent behavioral regime of the system. As a future work, it may be possible to exploit the potential of Lambda genome parameter to guide evolution towards desirable levels of phenotypic structural complexity.

Acknowledgment

Thanks to John Anthony for carrying out part of the experimental work during his master thesis project.

References

1. Doursat, R., Sánchez, C., Dordea, R., Fourquet, D., Kowaliw, T.: Embryomorphic engineering: emergent innovation through evolutionary development. In Morphogenetic Engineering, R. Doursat, H. Sayama, O. Michel, 275-311. UCS Series, Springer-Verlag (2012).
2. Kumar, S., Bentley, P.J.: Biologically inspired evolutionary development. 5th International Conference on Evolvable Systems (ICES03), LNCS, 57–68. Springer (2003)
3. Miller, J.F., Banzhaf, W.: Evolving the program for a cell: from French flag to boolean circuits. S. Kumar and P. J. Bentley editors, On Growth, Form and Computers, 278–301. Elsevier Limited Oxford UK (2003)
4. Tufte, G.: 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, 624–631. MIT Press, Cambridge, MA (2008)
5. Li, M., Vitanyi, P.M.B.: An introduction to Kolmogorov complexity and its applications. Springer-Verlag. New York, 2nd edition, (1997)
6. Langton, C.G.: Computation at the edge of chaos: phase transitions and emergent computation. S. Forrest editor, Emergent Computation, 12–37. MIT Press (1991)
7. Wolfram, S.: Universality and complexity in CA. Physica D, 10(1-2): 1–35 (1984)
8. Kolmogorov, A.N.: Three approaches to the quantitative definition of information. Problems Inform. Transmission 1 (1): 1–7 (1965)
9. Turing, A.: On computable numbers, with an application to the Entscheidungsproblem, Proceedings of the London Mathematical Society, Series 2, 42: 230–265 (1936)
10. Hartmann, M., Lehre, P.K., Haddow, P.C.: Evolved digital circuits and genome complexity. NASA/DoD Conference on Evolvable Hardware '05, 79–86. IEEE Press (2005)
11. Lehre, P.K., Haddow, P.C.: Developmental mappings and phenotypic complexity. Proceedings of 2003 IEEE CEC, volume 1, 62–68. IEEE Press (2003)
12. Deutsch, L.P.: DEFLATE Compressed data format specification version 1.3 (1996)

13. Ziv, J., Lempel, A.: A universal algorithm for sequential data compression. IEEE Transactions on Information Theory, 23(3): 337-343 (1977)
14. Huffman, D.A.: A method for the construction of minimum-redundancy codes, Proceedings of the I.R.E., 1098–1102 (1952)
15. Tufte, G., Nichele, S.: On the correlations between developmental diversity and genomic composition. 13th Annual Genetic and Evolutionary Computation Conference, GECCO 2011, 1507-1514. ACM (2011).
16. Nichele, S., Tufte, G.: Genome parameters as information to forecast emergent developmental behaviors. 11th International Conference on Unconventional Computation and Natural Computation, UCNC 2012, 186-197. Springer (2012).
17. Kowaliw, T.: Measures of complexity for artificial embryogeny. GECCO 2008, 843-850. ACM (2008)
18. Ulam, S.: Los Alamos National Lab 1909-1984. Los Alamos: Los Alamos Science. Vol. 15 special issue, 1-318 (1987)
19. von Neumann, J.: Theory and organization of complicated automata. A. W. Burks, (1949) 29-87 [2, part one]. Based on transcript of lectures delivered at the University of Illinois in December 1949
20. Zenil, H., Villareal-Zapata, E.: Asymptotic behaviour and ratios of complexity in cellular automata. arXiv/1304.2816 (2013)