Developmental Mappings and Phenotypic Complexity

Per Kristian Lehre and Pauline C. Haddow
Department of Computer and Information Science
Norwegian University of Science and Technology
Sem Sælands vei 7-9, NO-7491 Trondheim, Norway
{lehre,pauline}@idi.ntnu.no

Abstract- The effect of phenotypic complexity on distance correlation plots is investigated for two developmental mappings, a mapping based on L-systems, and a 2D cellular automata mapping. Our treatment of complexity is based on the theory of Kolmogorov complexity. A new genotype sampling algorithm called Crossection Walk is introduced.

1 Introduction

Multicellular organisms develop from single-celled zygotes to grown-up organisms in a developmental process implicitly defined by their DNAs. Biological development thus consists of a small genotype (relative to the phenotype) and a development process (implicitly defined by the genotype) which enables the genotype to develop to the phenotype.

The scalability inherent in biological development has attracted the interest of a number of researchers in the field of Evolutionary Computation (EC) and particularly in the field of evolvable hardware (EHW).

Several attempts have been made to construct artificial equivalents of a developmental process. We will refer to such approaches as *artificial development* but they are also termed ontogeny, embryogenesis or computational development.

At least two main directions can be found in this subfield. One is to make biologically plausible models of development in order to learn about development in nature. These models may potentially also be used to address applications in EC. The other is to make indirect genotype-phenotype mappings inspired by biological development.

Early work includes Kitano [Kit90] who used a matrix-rewriting grammar (based on L-systems [Lin68]) as implicit genotype-phenotype mappings for evolving artificial neural network (ANN) structures . Later, Gruau proposed a graph rewriting grammar called cellular encoding [Gru94], also for evolving ANN structures. Boers and Sprinkler also employed graph grammars [BSK01] in the same application domain. Other research includes [MT03, GB02, vRCD$^+$03, TH03, Ben03, Hor03].

The field of artificial development is an expanding field. Empirical results, although promising, may be said to be inconclusive with respect to the scalability of artificial development. This work takes a step toward answering the scalability question by considering phenotypic complexity i.e. the complexity of the solutions achievable through the application of artificial development. Our notion of complexity is based on the theory of Kolmogorov complexity [LV97]. Our goal is to see if phenotypic complexity has any effect on distance correlation plots where these plots illustrate the distance preservation between pairs of genotypes and their corresponding phenotypes for a given developmental mapping. Experiments are based on two existing developmental mappings: Kitano's matrix rewriting grammar [Kit90] and 2D cellular automata.

The rest of the paper is organized as follows: In Section 2, Kolmogorov complexity is briefly introduced, the plots are described, and the Crossection Walk sample algorithm is introduced. Details of the experimental setup are described in Section 3, and the following Section 4 contains the distance correlation plots and an interpretation of these result. Finally, in Section 5, the results are set in a larger context, along with some advices on further research.

2 Theory

We will use the following notation. If the symbol x denotes a bitstring, then the symbol $\overline{x}$ denotes the complementary bitstring and $x(k)$ is the kth bit of bitstring x. So, if the bitstring $x = 1100$, then the complementary bitstring $\overline{x} = 0011$ and the bit $x(2) = 1$. We will let $D(\cdot,\cdot)$ denote the Hamming distance on bitstrings. As such, we have $D(x,\overline{x}) = 4$ when $x = 1100$.

A *developmental mapping* may be represented by a function $\phi : \mathcal{G} \to \mathcal{P}$ where the set $\mathcal{G}$ is called the *genospace* and is equipped with a distance metric $d_{\mathcal{G}} : \mathcal{G} \times \mathcal{G} \to \mathbb{R}$, and correspondingly, the set $\mathcal{P}$ is called the *phenospace* and is equipped with a distance metric $d_{\mathcal{P}} : \mathcal{P} \times \mathcal{P} \to \mathbb{R}$. Elements of the genospace $\mathcal{G}$ are called genotypes and elements of the phenospace $\mathcal{P}$ are called phenotypes. Given two genotypes x and y and their respective phenotypes $\phi(x)$ and $\phi(y)$, we say that their *genotypic distance* is $d_{\mathcal{G}}(x,y)$ and their *phenotypic distance* is $d_{\mathcal{P}}(\phi(x),\phi(y))$. The overall situation is illustrated in Figure 1.

Our objective is to characterize how the developmental mapping ϕ preserves distances from $\mathcal{G}$ into $\mathcal{P}$ i.e. how well the genotypic distances correlate to corresponding phenotypic distances. And such correlations are studied with *distance correlation plots* where genotypic distances are plotted against phenotypic distances.

If we assume that the correlation is invariant over the whole genospace, we could obtain the distance correlation plots simply by plotting genotypic distance against phenotypic distance for randomly sampled pairs of genotypes.

But, as we will see, the assumption that distance-preservation is invariant does not always hold. A developmental mapping may have regions where distances are very well preserved, and other regions where distances are hardly

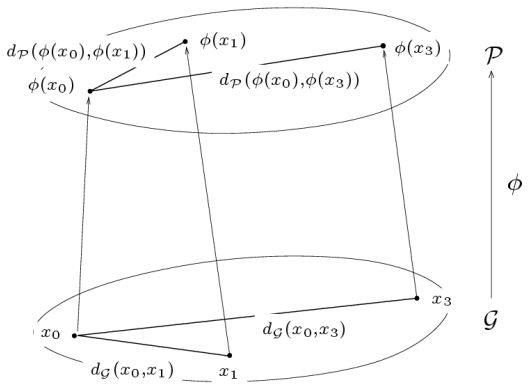


Figure 1: Artificial Developmental Mapping from Genospace to Phenospace.

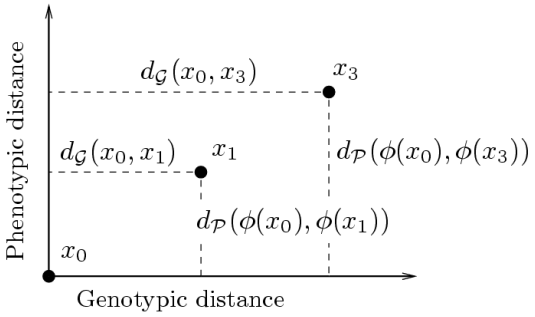


Figure 2: Illustration of a Distance Correlation Plot.

preserved at all.

We, therefore, characterize distance-preservation by sampling different regions of each developmental mapping. One distance correlation plot is made for each region.

Subsection 2.1 briefly introduces the complexity measure and how it is approximated. Subsections 2.2 and 2.3 contain descriptions of the distance correlation plots along with the sampling algorithm Crossection Walk.

2.1 Kolmogorov Complexity

What are complex bitstrings? Consider the following example:

$$\begin{aligned} x &= 10101010101010101010 \\ y &= 11010101011011101010 \end{aligned}$$

The bitstring x appears less complex than the bitstring y. The bistring x is simply ten repetitions of 10. Increasing the length of bitstring x by adding more repetitions of 10 does not make x any more complex. It is difficult to see any pattern in bitstring y.[1]

Bitstring x is simple because we can make a short description of the string. Bitstring y appears more complex since it it cannot be easily described in a few words.

Kolmogorov associated the complexity of an object with the length of the shortest description of the object. To have an unambiguous measure of an object's description length, the description length was defined as the length of the shortest program that generates the object on a fixed, universal Turing machine. (See [LV97].)

The Kolmogorov complexity measure is incomputable, so for practical experiments we need an approximation. Compression is used as an overestimate of the Kolmogorov complexity. Here, we use the Lempel-Ziv algorithm to compress the binary phenotype. Bitstrings that are easily compressible have low complexity, whilst bitstrings that cannot be compressed are complex. So the complexity is proportional to the compression ratio.

[1] For the curious reader, this bitstring is related to the words phenotypic complexity.

This way of measuring complexity has both theoretical and practical advantages. It is based on well-founded theory, and the Lempel-Ziv approximation can easily be computed using the standard compression library `zlib`.

2.2 Genotype-Phenotype Distance Correlation Plots

Distance correlation plots illustrate how well distances are preserved in the genotype-phenotype mapping. Each plot is based on a single genotype x_0 in $\mathcal{G}$. We will call this genotype x_0 the *origin genotype of the plot*. Given a sampled subset G of the genospace $\mathcal{G}$, we plot the distance between the genotypes $d_{\mathcal{G}}(x_0, x_i)$ against the distance of the corresponding phenotypes $d_{\mathcal{P}}(\phi(x_0), \phi(x_i))$ for each element x_i in G. The result is a scatter plot which indicates the correlation between genotypic distance and phenotypic distance relative to the origin genotype of the plot. Figure 2 illustrates a distance correlation plot where x_0 is the origin genotype of the plot.

The functions $d_{\mathcal{G}}(\cdot,\cdot)$ and $d_{\mathcal{P}}(\cdot,\cdot)$ are distance metrics over the genospace and phenospace respectively. In this paper, the genotypes and the phenotypes are represented as bitstrings. Therefore, we have chosen the Hamming distance as distance metric, and we will simply denote it by $D(\cdot,\cdot)$.

An example of a simple correlation plot is a sharp diagonal. This would be obtained from a direct encoding mapping. At the other extreme, we can obtain a very different correlation plot from a mapping where the genotype is the seed to a random number generator and the phenotype is the pseudo-random number generated from the seed. With a good random number generator, the plot will look like uniformly distributed noise in the correlation plot.

2.3 Genotype sampling with Crossection walks

To sample a subset G of genotypes from the genospace $\mathcal{G}$, we propose a new sampling algorithm called *Crossection Walk*. The algorithm is defined in Figure 3.

The purpose of the algorithm, is to sample random elements from $\mathcal{G}$ where the genotype distances to the origin genotype x_0 are uniformly distributed along the entire interval $[0, d_{\mathcal{G}}(x_0, \overline{x_0})]$. Intuitively, the method is a "random walk" from x_0 to the complementary binary string $\overline{x_0}$. In each step, we move one unit distance from the origin x_0, and one unit distance closer to the complement $\overline{x_0}$.

Algorithm 1: Crossection Walk

Data : A bitstring x_0 of length ℓ

Result : A set of $\ell + 1$ bitstrings of length ℓ

begin

$y_0 \leftarrow x_0$

$J \leftarrow \{1, 2, ..., \ell\}$

$\sigma \leftarrow$ a random permutation of J

for $i \leftarrow 1$ *to* ℓ **do**

$$y_i(k) \leftarrow \begin{cases} \overline{y_{i-1}}(k) & \text{when} \quad k = \sigma(i) \\ y_{i-1}(k) & \text{elsewhere} \end{cases}$$

return $G = (y_0, y_1, ..., y_\ell)$

end

Figure 3: The algorithm used to sample a subset G of genotypes from the genospace $\mathcal{G}$.

Example: Two possible results of calling the algorithm with input $x_0 = 1100$ are the sequences of bitstrings

$$\begin{aligned} G_1 &= (1100, 1000, 0000, 0010, 0011), \text{ and} \\ G_2 &= (1100, 1101, 1111, 1011, 0011). \end{aligned}$$

3 Experiments

The following experimental procedure was carried out on both developmental mappings. Let $\phi : \mathcal{G} \rightarrow \mathcal{P}$ denote any of the two mappings.

First, we randomly sampled a set X of genotypes from the genospace $\mathcal{G}$. For each genotype x in X, the complexity of the corresponding phenotype $\phi(x)$ was approximated with Lempel-Ziv compression.

Then, for each genotype x in the set of genotypes X, the Crossection Walk Algorithm was called 100 times on input x. Denote the resulting set of genotypes G_x. Then, a distance correlation plot was computed based on this set G_x. Hence, the number of spots in each plot is 100 times the distance $D(x, \overline{x})$.

The result is a set of distance correlation plots with different origins, one for each of the genotypes x in X . These plots may be said to be sampled from different regions of the landscape.

The next subsection contains details about the artificial developmental mappings used in the experiments.

3.1 The two Developmental Mappings

We use two different artificial development mappings in the experiment: a slightly modified version of Kitano's original development mapping, and and a developmental mapping based on cellular automata (CA). We will refer to this latter mapping as the *CA mapping*. The CA used in the CA mapping have two states, 0 and 1, and are run on a finite 2D grid of size 32×32 with a Moore- neighborhood (i.e. 9 neighbors). For such CA, there are 2^{2^9} different update rules.

Each genotype is a bitstring of length 512 bits and corresponds to one update rule.

To produce the phenotype from the genotype, the cellular automaton is initialized with all cells in state 0 except for a single cell in the center with state 1. The automaton is then run for 100 iterations with the local update rule corresponding to the particular genotype. The phenotype is the resulting states of the 32×32 cells in the automaton after the 100 iterations, i.e. a bitstring of length 1024.

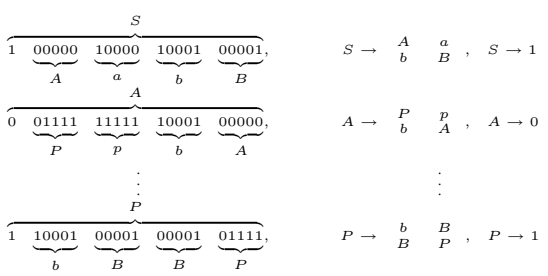
We used a slightly modified version of Kitano's development system [Kit90]. His mapping, which is based on D0L-systems (a class of L-systems), works on matrices of symbols.

In Kitano's original scheme, after the desired number of iterations, the matrix consisted of symbols and/or 1s and 0s. All symbols were then replaced in *a final iteration* by 0s to obtain the final bit matrix. This choice put a bias toward 0s, which was not a disadvantage for Kitano's application where the bits were adjacency matrix descriptions of neural network structures.

In our experiments, we want to balance the number of 1s and 0s. We therefore introduce an *extra set* of rewrite rules for upper-case symbols. Each of these rules has the form $Symbol \rightarrow \{0, 1\}$. In the final iteration, upper-case symbols are replaced by 0 or 1 according to the extra set of rewrite rules. Lower-case symbols are replaced by 0 as in the original scheme.

We run the modified developmental mapping until a matrix of 32×32 elements was produced from the grammar rules.

The grammar is encoded using a constant-size binary string. The encoding is divided into 17 blocks, one for each of the upper-case symbols $S, A, B, ..., P$. Each block is divided into five pieces. The first piece is one bit wide and is used in representing the extra rule set for the upper-case symbol. The other four pieces, each of five bits, designate a rewrite rule for the symbol. Each of the 32 symbols $A, B, ..., P, a, b, .., p$ is given a 5 bit unique identifier. A is 00000, B is 00001, a is 10000 and so on. An encoding of a grammar thus uses $17 \cdot (1 + 4 \cdot 5) = 357$ bits. An example is shown Figure 4. The left part of this figure shows a part of the binary encoding. The right part of the figure is the corresponding rewriting rules.



Binary encoding Matrix rewriting rules

Figure 4: Fixed-size binary encoding of matrix rewriting grammar.

4 Results and Discussion

The experimental results are summarized in the plots in Figure 5 and Figure 6. From each experiment, two plots were produced, a distance correlation plot and a plot of the phenotype. The plots in the left columns are distance correlation plots where the x-axis represents genotypic distances and the y-axis represents phenotypic distances to the origin genotype of the plot. These plots illustrate the correlation between the two distances. The plots in the right column are pictures of the corresponding phenotype bitstrings. The bitstrings are visualized as pixel images, where white pixels represent 0 and black pixels represent 1.

The plots are ordered with increasing phenotypic complexity as may be seen from the phenotype plots to the right. In the upper plot, the origin genotype has a phenotype with low complexity. In the bottom plot, the origin genotype has a phenotype with high complexity.

For convenience, we will denote plots where the origin genotype has a phenotype with low complexity as *low complexity plots*, and we will denote plots where the origin genotype has a phenotype with high complexity as *high complexity plots*.

Figure 5 contains distance correlation plots for the matrix rewriting grammar mapping with corresponding phenotype plots. The upper plot in Figure 5 is a low complexity plot. As described in Section 2.2, the first plot closely resembles a direct encoding mapping from genotype to phenotype. The variance in the distance correlation plot is relatively high. The corresponding phenotype picture consists of large blocks of uniform color, which is of course relatively easy to compress.

As we continue downwards through the plots, a curve starts to appear. This curve introduces a certain threshold which is indicated with an arrow labeled 1 in the bottom plot. Above this threshold, the plot is almost horizontal. Below the threshold, the plot is almost linear.

It should be noted that as we move down the plots towards the high-complexity plots, the variance reduces. The corresponding phenotype plots become more irregular, i.e. harder to compress.

In Figure 6, the distance correlation and phenotype plots for the CA mapping are shown. Beginning at the top, the plot has a relatively high variance. But the plot is not linear as was the case for the low-complexity plot in Figure 5. There seems to be a threshold at this point.

As we move down the plots, the variance in the plots reduces significiantly. The bottom plot appears as a thin, horizontal stripe. The corresponding phenotype is highly irregular.

For both mappings, high complexity plots appear smoother than the low complexity plots.

Similar effects of phenotypic complexity were observed in the other plots not shown here. The effect was most easily observed on the matrix rewriting grammar mapping.

5 Conclusion

We have shown that phenotypic complexity when measured with Kolmogorov complexity, has a clear, and strong impact on distance correlation plots for a matrix-rewriting grammar mapping and a 2D CA mapping.

High phenotypic complexity appears to have a negative effect on the distance correlation plots. In the matrix rewriting grammar mapping, very low phenotypic complexity gives plots which resembles that of direct encoding.

The fact that we observed a very similar effect with two relatively different developmental mappings strengthens our result. However, we cannot claim that all developmental mappings will behave similarly. Indeed, we hope that there may be some mappings that can cope with phenotypic complexity.

A new sampling algorithm called Crossection Walk was introduced to sample genotypes where the distance to a specific genotype is uniformly distributed.

The method used to produce the distance correlation plots is application-independent and relatively simple. Researchers proposing new developmental mappings can easily carry out the same procedure as long as the corresponding genospace and phenospace allow computable metrics.

5.1 Future Work

In order to make our results more general, the method should be carried out on other developmental mappings.

A theoretical explanation of the experimental results would be interesting.

The effect of phenotypic complexity on the performance of an evolutionary algorithm using artificial development was not discussed in this paper. This could possibly be investigated on a set of test problems.

Developmental mappings could be investigated with a similar procedure on other phenotypic properties than complexity.

Acknowledgements

The first author wishes to thank Martin Thorsen Ranang and Rolv Seehuus for helpful comments on an early draft of the document.

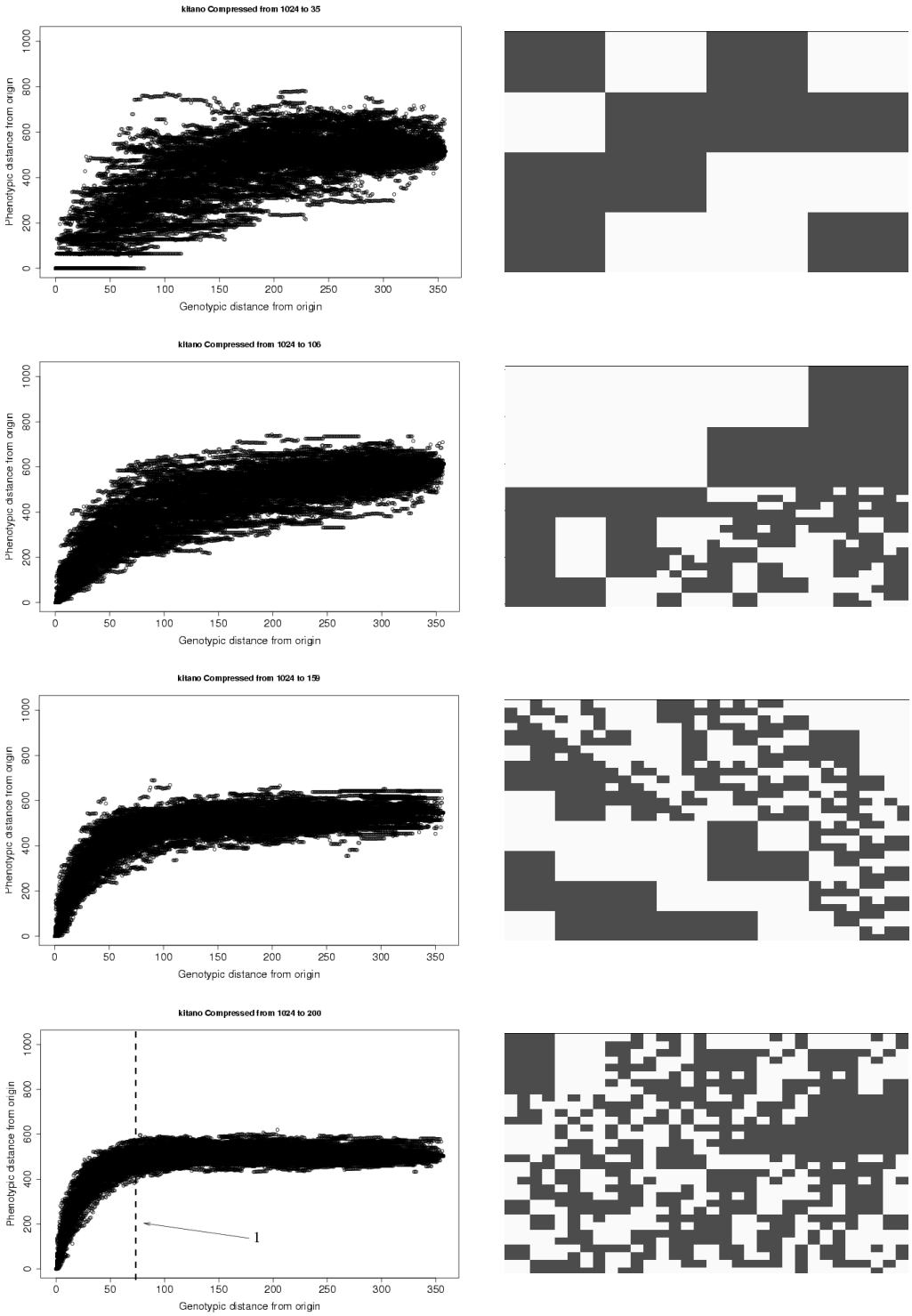


Figure 5: Left: Distance correlation plots from the matrix rewriting grammar mapping. Right: Pictures of corresponding phenotypes.

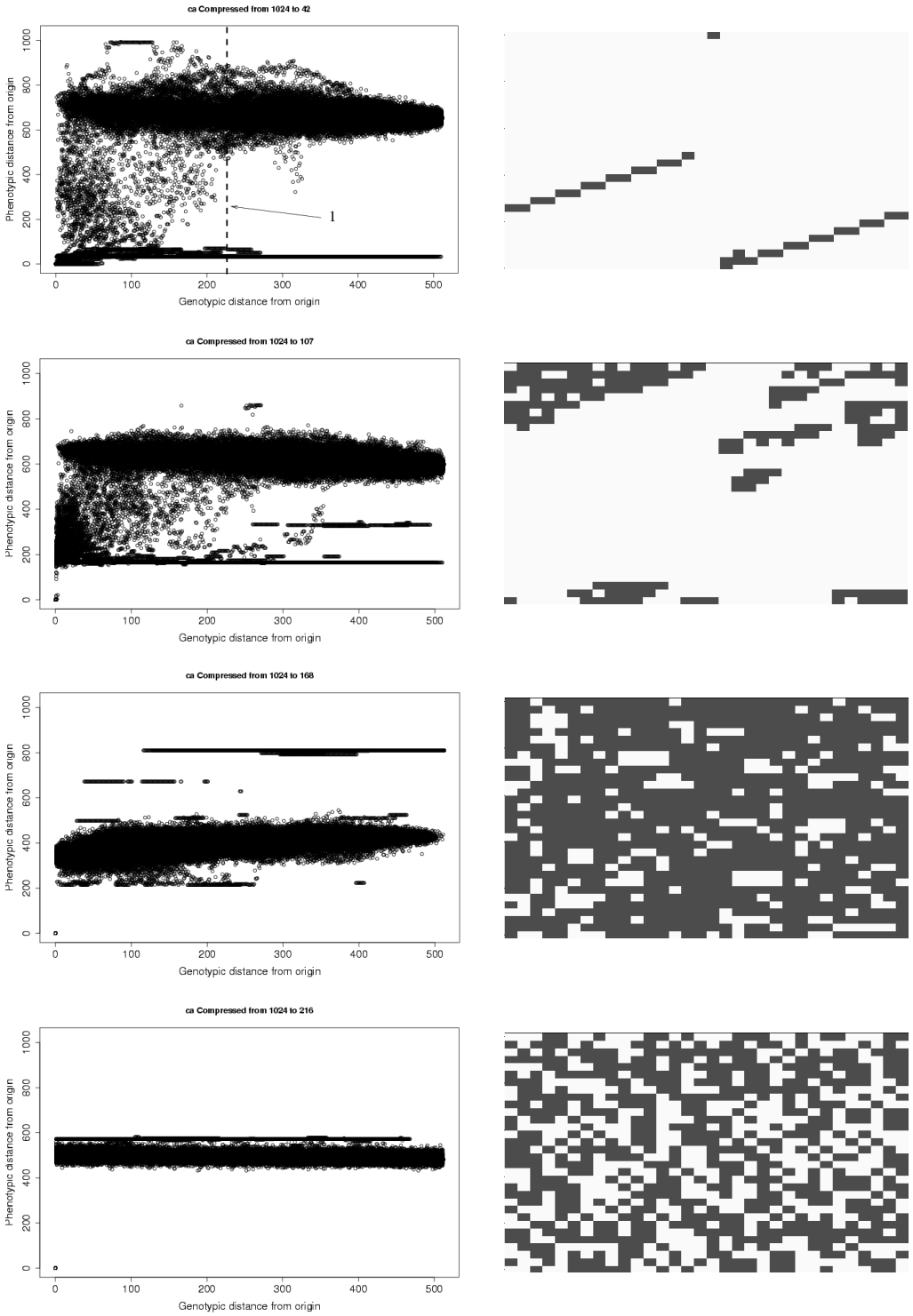


Figure 6: Left: Distance correlation plots from the CA mapping. Right: Pictures of corresponding phenotypes.

Bibliography

[Ben03] Peter J. Bentley. Evolving fractal proteins. In Andy M. Tyrrell, P. C. Haddow, and J. Tørresen, editors, *Evolvable Systems: From Biology to Hardware Proceedings of the 5th International Conference on Evolvable Systems, ICES'2003, Trondheim, Norway March 2003*, volume 2606 of *Lecture Notes in Computer Science*, pages 81–92. Springer-Verlag, Berlin, Heidelberg, 2003.

[BSK01] E.J.W. Boers and I.G. Sprinkhuizen-Kuyper. Combined biological metaphors. In M.J. Patel, V. Honavar, and K. Balakrishnan, editors, *Advances in the Evolutionary Synthesis of Intelligent Agents*, A Bradford Book, chapter 6, pages 153–183. MIT Press, Cambridge, Massachusetts, 2001.

[GB02] T.G.W. Gordon and P.J. Bentley. Towards development in evolvable hardware. In Adrian Stoica, Jason Lohn, Rich Katz, Didier Keymeulen, and Ricardo Salem Zebulum, editors, *The 2002 NASA/DoD Conference on Evolvable Hardware*, Alexandria, Virginia, 15-18 July 2002. Jet Propulsion Laboratory, California Institute of Technology, IEEE Computer Society.

[Gru94] F. Gruau. *Neural Network Synthesis using Cellular Encoding and the Genetic Algorithm.* PhD thesis, France, 1994.

[Hor03] Gregory S. Hornby. Generative representations for evolving families of designs. In E. Cantú-Paz, J. A. Foster, K. Deb, D. Davis, R. Roy, U.-M. O'Reilly, H.-G. Beyer, R. Standish, G. Kendall, S. Wilson, M. Harman, J. Wegener, D. Dasgupta, M. A. Potter, A. C. Schultz, K. Dowsland, N. Jonoska, and J. Miller, editors, *Genetic and Evolutionary Computation – GECCO-2003*, pages 1678–1689, Berlin, 2003. Springer-Verlag.

[Kit90] H. Kitano. Designing neural networks using genetic algorithm with graph generation system. *Complex Systems*, 4:461–476, 1990.

[Lin68] Aristid Lindenmayer. Mathematical models for cellular interactions in development, I & II. *Journal of Theoretical Biology*, 18:280–315, 1968.

[LV97] Ming Li and Paul M. B. Vitanyi. *An Introduction to Kolmogorov Complexity and Its Applications*. Springer-Verlag, New York, 2nd edition, 1997.

[MT03] Julian F. Miller and Peter Thomson. A developmental method for growing graphs and circuits. In Andy M. Tyrrell, P. C. Haddow, and J. Tørresen, editors, *Evolvable Systems: From Biology to Hardware Proceedings of the 5th International Conference on Evolvable Systems, ICES'2003, Trondheim, Norway March 2003*, volume 2606 of *Lecture Notes in Computer Science*, pages 93–104. Springer-Verlag, Berlin, Heidelberg, 2003.

[TH03] Gunnar Tufte and Pauline C. Haddow. Building knowledge into developmental rules for circuit design. In Andy M. Tyrrell, P. C. Haddow, and J. Tørresen, editors, *Evolvable Systems: From Biology to Hardware Proceedings of the 5th International Conference on Evolvable Systems, ICES'2003, Trondheim, Norway March 2003*, volume 2606 of *Lecture Notes in Computer Science*, pages 69–80. Springer-Verlag, Berlin, Heidelberg, 2003.

[vRCD+03] Piet van Remortel, Johan Ceuppens, Anne Defaweux, Tom Lenaerts, and Bernad Manderick. Developmental effects on tunable fitness landscapes. In Andy M. Tyrrell, P. C. Haddow, and J. Tørresen, editors, *Evolvable Systems: From Biology to Hardware Proceedings of the 5th International Conference on Evolvable Systems, ICES'2003, Trondheim, Norway March 2003*, volume 2606 of *Lecture Notes in Computer Science*, pages 117–128. Springer-Verlag, Berlin, Heidelberg, 2003.