Evolution and Development of a Multi-Cellular Organism: Scalability, Resilience and Neutral Complexification

Diego Federici* and Keith Downing
Norwegian University of Science and Technology
Department of computer and information science
N-7491 Trondheim, Norway
Email: {federici | keithd} @idi.ntnu.no

* corresponding author
Phone: +47 73551018

February 15, 2005

keywords: genetic algorithms, development, scalability, fault tolerance, neutral complexification

Abstract

To increase the evolvability of larger search spaces, several indirect encoding strategies have been proposed. Among these, multicellular developmental systems are believed to offer great potential for the evolution of general, scalable and self-repairing organisms. In this paper we reinforce this view, presenting the results achieved by such a model and comparing it against direct encoding. Extra effort has been made to make this comparison both general and meaningful.

Embryonal stages, a generic method showing increased evolvability and which can be applied to any developmental model, are introduced. Development with embryonal stages implement what we refer to as direct 'Neutral Complexification': a direct genotype complexification mechanisms by neutral duplications of expressed genes.

Results show that, even for high complexity evolutionary targets, the developmental model proves more scalable. The model also shows emergent self-repair, which is used to produce highly resilient organisms.

1 Introduction

Without apriori specific knowledge to use, it is well understood that the bigger the search space the less efficient the search will be. This makes the evolution of large phenotypes one of the most serious problems in the field of evolutionary computation (EC).

On the other hand, biological systems seem to have come to terms with it since living organisms can easily contain trillions of cells (with each cell being itself a structure of baffling complexity). These systems rely upon an artifice, the emergent process during which a single replicating cell develops into the mature organism. Inside each cell, an identical restricted set of genes interact to provide the instructions for development. Morphology emerges as initially identical cells interact with their local environment assuming specific roles.

Ontogeny is the result of the distributed process during which a relatively small genotype decompresses into a large phenotype. For example, it is estimated that there are only $30K$ genes in the human genotype ($45M$ DNA bases), while $50T$ (Tera) cells constitute a mature phenotype.

However, even if evolving direct representations of structures of similar complexity is inconceivable, is not yet well understood under which circumstances a developmental process can be beneficial to EC.

Specifically open questions regard:

- the evolvability of complex phenotypes
- the scalability of methods based on Artificial Embryogeny (AE)
- the properties of such systems

This paper addresses these issues with an empirical study of a model of multicellular development, which is compared to direct encoding. In addition we include a discussion and propose a test suited for this analysis.

Results show that:

- Development with Embryonal Stages (DES), a method based upon the biological mechanisms of gene duplication and diversification, has significant benefits for evolvability. DES implements neutral complexification and operates by preventing pleiotropy among different stages of development.
- Development results are more scalable than direct encoding for both regular and high complexity targets.

The remainder of the paper is organized as follow: section 2 contains an introduction on artificial and biological development and describes neutral complexification. Section 3 discusses the evolutionary task. Section 4 describes the adopted multicellular development model, while section 5 gives details of the evolutionary methods for Direct Encoding (DE) and Artificial Embryogeny (AE). Section 6 presents results from the simulations concerning the use of embryonal stages, scalability comparison between DE and AE, performance under

various AE settings and emergent and evolve regeneration features of the AE model. Section 7 contains the conclusions.

2 Development

Since the search space grows exponentially with the genotype size, the evolution of large phenotypes should benefit from parsimonious encodings. Among these indirect encodings, development uses a genetically encoded growth program in several recursive steps. Matrix Rewriting [20] and Cellular Encoding [14] are emblematic approaches to such a definition; both are based on grammatical rules.

Rewriting rules can be applied an arbitrary number of times, so that the genotype size should be highly independent of the phenotype size. On the other hand, development introduces a level of indirection between genotype and phenotype. The level of indirection is proportional to the amount of gene reuse, which is often linked to the number of developmental steps. For example, consider the following grammars:

$$\begin{array}{l} S \rightarrow aSa \\ S \rightarrow aSb \end{array} \quad \text{producing} \quad \begin{array}{c} aaaaa...S...aaaaa \\ aaaaa...S...bbbbb \end{array} \tag{1}$$

A single change in the rule/genotype induces a phenotypic change proportional to the number of substitution/developmental steps. The result is that the correlation between genotype and phenotype is reduced with each rule reuse. A decreasing correlation normally leads to rougher fitness landscapes, which means harder searches. Ultimately, the impact of the search space compression on the fitness landscape is what rules the performance of artificial developmental systems.

In AE, genes activated early in development are very costly to change. With a sort of Domino Effect, modifications affecting early phases of development will have huge phenotypic consequences. Big leaps in the phenotypic space go against the basic evolutionary assumption of small incremental refinements.

Interestingly, this is valid also in natural evolution. Observing the development of embryos, it is seen that the more related two species the more similar their ontogeny. It is actually possible to judge phylogenetic relatedness by looking for how long embryos of different species share a common developmental dynamic:

> It is generally observed that if a structure is evolutionary older than another, then it also appears earlier than the other in the embryo. Species which are evolutionary related typically share the early stages of embryonal development and differ in later stages. [...] If a structure was lost in an evolutionary sequence, then it is often observed that said structure is first created in the embryo, only to be discarded or modified in a later embryonal stage. Examples include:

> Whales, which have evolved from land mammals, don't have legs, but tiny remnant leg bones lie buried deep in their bodies. During embryonal development, leg extremities first occur, then recede. Similarly, whale embryos (like all mammal embryos) have hair at one stage, but lose most of it later. All land vertebrates, which have evolved from fish, show gill pouches at one stage of their embryonal development. The common ancestor of humans and monkeys had a tail, and human embryos also have a tail at one point; it later recedes to form the coccyx. The swim bladder in fish presumably evolved from a sac connected to the gut, allowing the fish to gulp air. In most modern fish, this connection to the gut has disappeared. In the embryonal development of these fish, the swim bladder originates as an out pocketing of the gut, and the connection to the gut later disappears. Wikipedia [34]

This conservative nature of ontogeny is a direct consequence of the aforementioned 'Domino Effect' since redesign is harder than incremental changes. Therefore, mutations causing big phenotypic effects are counterproductive because, with good approximation, they never produce better individuals.

Another issue relates to the regularity of the phenotypes that AE produces. Development is de facto a decompression of the genotype and since compression is generally higher for regular targets, a serious question is how much these methods are viable for the evolution of targets of high complexity.

Hints in this direction, also come from a study [21] on Matrix Rewriting [20], showing how the genotype-phenotype correlation decreases with the complexity of the phenotype. Again, low correlation decreases the chances of evolution by small incremental steps.

Micro-array analysis and gene sequencing are providing new data to understand how molecular evolution took place in organic systems. It is now clear that the most genetic innovations were originated by Gene Duplication [27] (see also Figure 1):

- Genotypes are not of fixed size. Notably, entire groups of genes can be duplicated (figure 1:B). Duplicated genes may not alter development at first since their expression is controlled by position invariant promoters and regulators. Evidence suggests that 90% of eukaryotic genes was produced by gene duplication [31].

- Duplicates can diverge assuming different roles (figure 1:C–E). Phenotypic traits that were linked by the expression of the same gene, can now evolve independently. This phase is often referred to as complexification, and can take two different forms: neofunctionalization (figure 1:C) and subfunctionalization (figure 1:D) [36].

The steps depicted in Figure 1 can be translated to the following steps of a

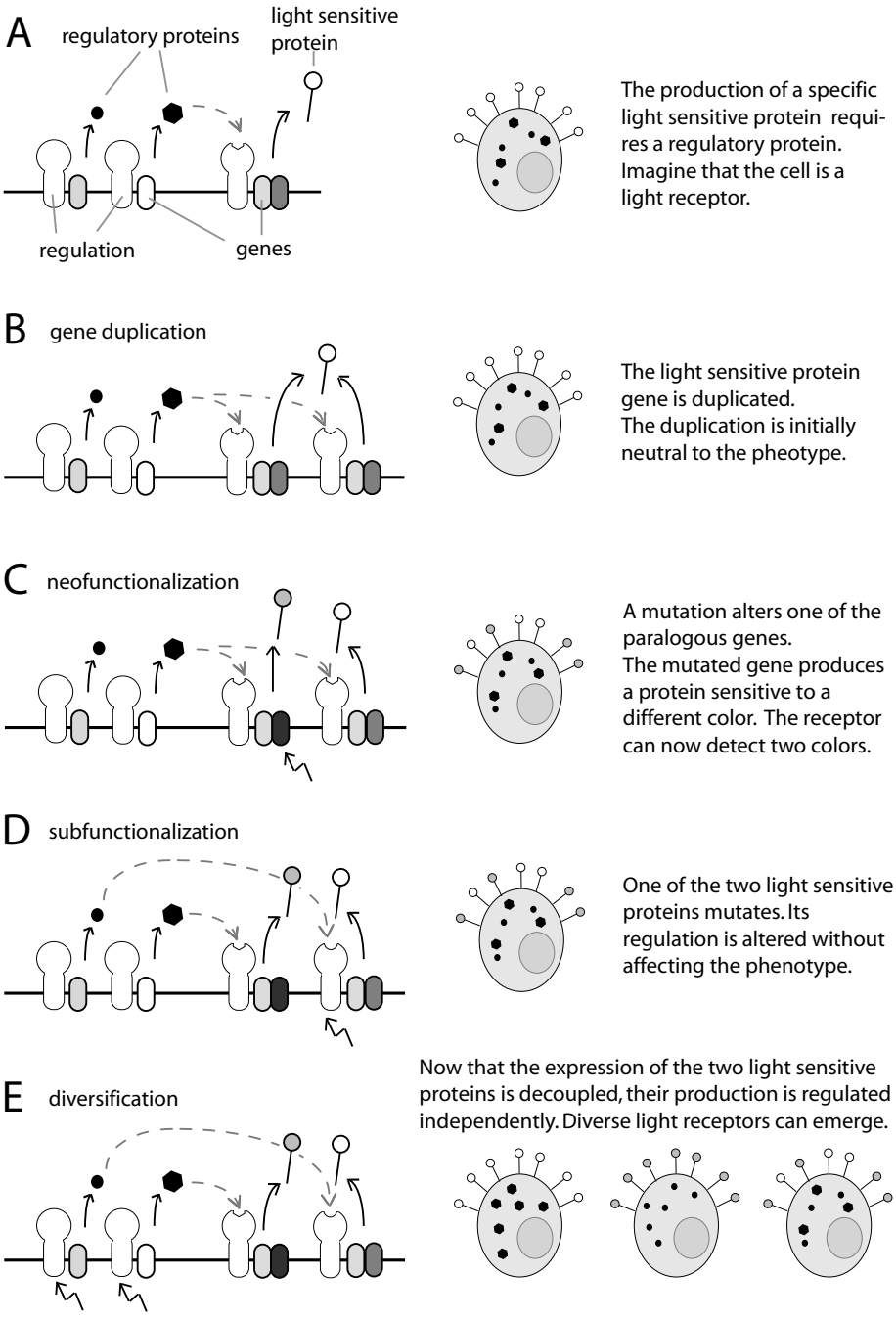


Figure 1: An example of phenotypic complexification by gene duplication. In this case gene duplication is initially neutral to the development of the phenotype. The genotypic redundancy introduced can eventually be exploited to increase the complexity of the phenotype.

grammar evolution:

	genotype	phenotype	type	
A)	$\{S \to AS\|\epsilon;\ \ A \to a\}$	aaaa...aa		
B)	$\{S \to AS\|\epsilon;\ \ A \to a\|a\}$	aaaa...aa	duplication	(2)
C)	$\{S \to AS\|\epsilon;\ \ A \to a\|b\}$	abba...ab	neofunc.	
D)	$\{S \to AS\|BS\|\epsilon;\ A \to a;\ B \to b\}$	abba...ab	subfunc.	

Neofunctionalization regards duplicated (paralogous) genes which assume a novel functionality. Example include the color specific opsins found in the light receptors of the retina and the Hox genes responsible for the vertebrate body structure. Subfunctionalization is the possibility that regulation of paralogous genes gets changed while the genes retain the same function. An example is found in the Zebra-Fish, where the genes engrail1 and engrail1b are expressed in different parts of the body, pectoral and hindbrain respectively, while they both produce the same protein. The engrail1b gene, originated from a duplication of engrail1, has diverged in its regulation while conserving the same functionality. It is possible to imagine that in the future they could eventually assume different specific functions.

The case of the engrail1 genes, appears as a Neutral Complexification (NC) of the genotype: an increase of genetic material which is functional (expressed) but neutral to the phenotype.

NC makes possible an increase of sophistication with small incremental steps consisting of neutral duplication events followed by divergence of paralogous genes.

For evolution, NC allows an initial fast exploration of a restricted search space, followed by an enlargement of the genotype which can achieve higher levels of specialization. It is possible to imagine that the two phases will repeat over and over, as individuals compete for higher values of fitness.

The great advantage of this incremental approach is that it induces an ordering of the search space, leading to an incremental increase of complexity from the initial genotypes. In this simple to complex re-ordering of the solution space lies its promise of higher performance.

2.1 implementing complexification

There are a number of genotype phenotype mappings that implement, either directly or indirectly, complexification.

One approach is based on artificial Gene Regulatory Networks (aGRN) with variable length genotypes [23, 5]. Genes are identified by a promoter site and therefore can occur anywhere in the genotype. Additionally, gene's activation regulate (and is regulated by) the expression of other genes. Activated genes are then used to define the morphology of the evolved organisms. Since gene functionality is position invariant, gene duplication can be achieved simply by replicating genotype sub-strings, while divergence is obtained by traditional mutation operators.

Notice that complexification is not generally possible for fixed-size string based aGRN models, such as those presented in [11, 3, 18, 8].

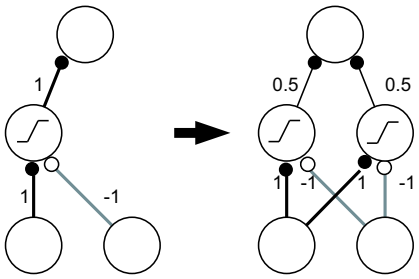
The advantage of this indirect implementation of NC is that it can achieve diversification of both the phenotypic traits (as in figure 1:3-4) and the regulatory network (by duplicating regulatory genes).

On the other hand, the approximate matching required for the identification of activated genes is quite expensive and grows linearly with the size of the genome and the number of regulatory genes/proteins[1]. This might impose a limit to the scalability of multicellular organisms, since the determination of each cell behaviour requires a separate activation of the aGRN.

Also, in Evolutionary Hardware there is a nascent interest for multicellular development [32, 25, 33], but given the state of technology, these models do not appear suited for hardware implementation.

Finally, current models do not protect against mutations affecting early phases of development, which we believe reduce system evolvability. This however could be easily avoided by using a mutation operator which takes into consideration the age of each gene.

A second, more abstract, approach is to increase sophistication by direct NC steps, adding specific active elements which do not alter functionality.



Depiction of Neutral Complexification for an artificial neural network. The network on the left is augmented horizontally by adding an active element without altering its functionality. Changes to the weights of the new network can produce more sophisticated functions than for the old one. Without altering functionality, complexification can be achieved incrementally. This mechanism, among others, is presented in the NEAT system [30]

Figure 2: Neutral Complexification

As far as we know, apart form the mechanism presented in this paper, there is no developmental system that implements direct NC. On the other hand, the NEAT system [30], a variable length direct encoding for Artificial Neural Networks (ANN), allows new neurons to be added almost neutrally with minimal changes to functionality, producing complexification by small incremental evolutionary steps. Also in this case, system evolvability, can benefit from the smaller-to-bigger reordering of the search space. Figure 2 gives an example of how such mechanism can be implemented for ANNs. Unfortunately, it is not

[1]The computational complexity for the approximate matching of a genome of length n with a single substring of length m is $O(mn)$

always possible to achieve NC, especially when non-linearities are present.

2.2 Development with Embryonal Stages

The regulatory system controls gene expression over two orthogonal dimensions: time and space. Development with Embryonal Stages (DES) implements a direct mechanism of Neutral Complexification (NC) for the temporal dimension.

As development spans over several consecutive steps, the idea is to start evolution with a single growth program (chromosome) which controls all the steps. As evolution proceeds, a new chromosome can be added by gene duplication.

The developmental steps are therefore partitioned into two groups. The first, controlling the initial steps of embryogenesis, is associated with the old chromosome. The latter, completing growth, is associated with the new, identical, duplicated chromosome.

Being exact copies, new chromosomes do not alter development, and are therefore neutral. But eventual mutations can independently affect each paralogous gene.

To avoid the Domino Effect caused by mutation altering early phases of development, the original chromosome can have a lower mutation rate or be arbitrarily excluded from evolutionary operators (see Section 6.2). In the latter case, only the new chromosome, which completes the maturation process, is allowed to change.

Likewise, new chromosomes can be added one by one, each one controlling the partition of the last development steps (see Figure 3).

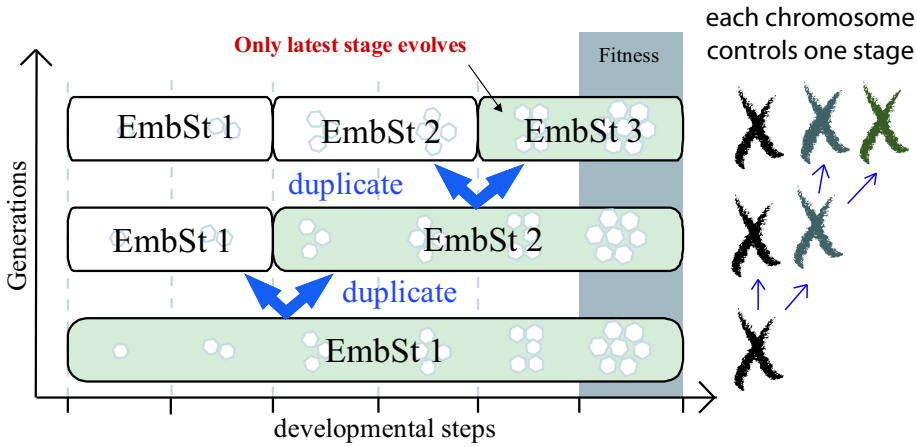


Figure 3: Embryonal stages. New chromosome duplicates are added incrementally without altering development. Evolution can thereafter alter only the growth program/chromosome controlling the latest steps of development. Without changing its size, new embryonal stage alters the search space by adding resolution around the current genotype and locking distal regions.

When only the latest embryonal stage is allowed to change, the advantage of DES is that each new stage restricts the evolutionary task to the optimization of fewer developmental steps. The new chromosome must take care of the maturation of an already partially developed phenotype. This new starting phenotype, as opposed to the zygote, is the result of the original chromosome evolution and provides a flying start for the second stage.

New stages do not modify the size of search space, but by controlling fewer developmental steps, they increase the resolution around the area represented by the current genome. Overall, the effect is an increase in genotype-phenotype correlation which leads to higher evolvability (see section 6.1).

2.3 Related development models

Several indirected encoding schemes have been proposed. All these 'Artificial Embryogeny' (AE, [29]) methods define the phenotype with a mapping which allows recursive gene reuse.

It is possible to distinguish two major evolutionary approaches to development. There are grammar-based approaches in which the genotype defines the substitution rules which are repeatedly applied to the phenotype. Examples include the Matrix Rewriting scheme [20], the Cellular Encoding [14], and Edge Encoding [22].

The major advantage of these methods is that compact grammar rules can generate a high number of organized patterns. For example, L-systems can easily produce interesting structures for locomotion. On the other hand, our target may contain many different phenotypic traits, each requiring a different production rule and symbol. Therefore the space of possible grammars is bound to grow with the complexity of the phenotype.

Some models include additional contextual information in each rule definition [17, 16], so that phenotypic traits variations can be generated. Also, it is possible to implicitly define the grammar by means of an aGRN [5] and use the accumulated concentrations of simulated chemicals to modulate the characteristics of morphological constituents.

In this direction, and inspired by Cellular Automata (CA), a second approach is to evolve the rules by which cells alter their metabolism and duplicate. Cells are usually capable of sensing the presence of neighboring cells [9], releasing chemicals which diffuse in simulated 2D or 3D environments [25], and moving and growing selective connections to neighboring cells [6].

Closely related to the one presented in this paper, Bentley and Kumar proposed a model which develops tiling patterns [4]. Ontogeny starts from a single cell which duplicates following rules expressed in the genotype. Rule preconditions are regular expressions matching the local North-West-South-East neighborhood (NWSE) of each cell and their absolute position.

Miller extended the previous model and evolved specific patterns [25]. He allowed 4 different cell types (colors) and a chemical undergoing diffusion. Cells based their growth program on their type and the types and chemical concentrations from the 8 neighboring cells. The growth program itself was encoded in

a boolean network that was evolved with the Cartesian Genetic Programming.

Miller also analyzed the behaviour of the evolved phenotypes when subjected to severe mutilations. Phenotypes were shown to regrow the missing parts achieving a striking resemblance to the target. The self repair feature is very interesting since it was not selected for during evolution. A similar resistance to damage was also reported in [28].

An important feature common to these last AE models, is that their growth program is based only on local variables. The localization of the inputs is fundamental to achieve a distributed control of development, which allows practical hardware implementation, failure recovery and, since the size of the program does not change with the size of the phenotype, for scalability issues.

3 The evolutionary task

When looking at system evolvability, it appears that there is a trade off between the combinatorial gain achieved by searching in a restricted genotypic space and hindrances introduced by gene reuse.

The fact that gene reuse constitutes an advantage when searching for regular patterns is quite established. For example, in [16] a generative (grammar-based) and non-generative systems where tested in the evolution of a program defining the construction of a 3D table. The generative mapping achieved much higher performance.

Similarly, in [4], the task was to evolve tessellating patterns. Individuals growing from a single cell to the mature organism showed higher scalability when compared to direct encoding. On the other hand, the generated solutions were always very regular.

A valid solution to this problem is a rectangle covering half of the allowed growth area. Since the zygote is placed in the center, such rectangle can be produced by growing in three over the four possible directions. The model presented in this paper, for example, can often solve the tessellation problem with the initial population of randomly generated individuals.

The problem of the complexity achievable by developed phenotypes is fundamental since high complexity targets are more difficult to refine incrementally and therefore to evolve [21].

Many authors have tested their system for developing Artificial Neural Networks (ANN). ANN are a natural choice for several reasons. First their proper set up is a central issue of AI. In addition, the number of synapses grows combinatorially with the size and interconnectedness of the network. Finally, the organization of the brain shows repeated structures that should favor gene reuse.

Even if the problem is tempting and surely deserves attention, we believe that it could prove misleading for initial investigations. In fact, there are several network configurations with equivalent functionality. Searching in a restricted search space, AE could stumble on a good solution at random. This is particularly true in the case of simple neuro-controllers, where often the problem is

as simple as finding the proper connection from one point to another (e.g. from an IR sensor to the motor output).

In [25], in order to show the capabilities of a cell-based developmental model, a 9x12 French flag pattern was evolved. The reason for this choice is that the uniqueness of the desired shape guaranteed that the problem was hard to solve. In fact, it is not even possible to know apriori if a sufficient growth program exists.

The advantages of this test are plentiful:

1) By presenting the phenotypes as 2D pictures, the bias of the development system often emerges as repeated sub-patterns. Difficult to represent mathematically, such artifacts are easily observable.

2) The generality of the model can be ascertained by testing targets of various shapes and complexity.

3) The number of colors of the target and its dimension clearly specifies the size of the solution space.

4) For direct encoding this is one of the easiest problems to solve. With each pixel of the target pattern represented by a gene, not only every target is expressible, but this is equivalent to the One-Max problem[2], which is totally decomposable, non deceiving and has a strictly monotonic fitness landscape.

5) Fitness is an absolute measure of the individuals quality. This is not so easy for other tasks, such as the evolution of locomotion or robot navigation.

In this paper we have tested direct encoding and the AE system in the evolution of 4-color patterns of varying size and complexity (see figure 4). The targets were chosen with different degrees of regularity, with the Circle being the most regular and the Wolfram CA-generated pattern the most complex.

To test scalability, the targets size has been set to 8x8, 16x16, 32x32, 64x64, 128x128, 256x256. The tested solution spaces therefore range from 4^{64} to 4^{65536}. As the geometrical properties of the targets are size invariant the amount of regularity should be invariant as well.

As proposed in [21], a measure of complexity can be provided by general purpose compression algorithms. In this case, it was calculated using the well known ARJ compressor [24]. In figure 4, below each target shape, the compression ratio and the size of the compressed data segment[3] are plotted for all the used target sizes. The ratio of compression can be seen as a depiction of the target complexity (ARJ-complexity), the size of the compressed data gives an indication of quantity of information needed to reproduce the pattern.

Apart for the CA-generated pattern, the ARJ-complexity decreases geometrically with size. This is due to the fact that, while regularity is preserved

[2]where the fitness is the sum all bits in the genotype which are set to 1

[3]The ARJ headers, containing file name and other technicalities, are stripped for these computations

for all shapes, the size increases quadratically. For the high complexity CA-generated pattern, there is little regularity that can be exploited and therefore the ARJ-complexity converges to $\sim 15\%$.

4 The model of multicellular development

Organisms develop starting from a single cell to reach the mature multi-cellular organisms in a precise number of developmental steps. Cells replicate and can release simulated chemicals in intra-cellular and inter-cellular space.

Cell behaviour is governed by a growth program based on local variables, and represented by a simple Recursive Neural Network (RNN). The RNN is seen as a plausible model of the cell Gene Regulatory Network (GRN) [2, 10].

Compared to typical models of GRNs in EC [5, 18, 9, 11, 3, 23] RNNs are much faster to evaluate. Since the growth program is evaluated literally billions of times, for a multicellular organism this is a fundamental feature. Also, while the evolutionary properties of artificial GRNs are still subject to debate, neural networks are better understood.

Compared to discrete functions, such as Boolean Networks [19, 25, 9] or production rules [4], the space of continuous functions representable by RNNs allows finer tuning and richer neutral space. Neutral space is very useful to escape local optima [1, 15], which pleiotropy makes abundant in the search space.

4.1 cell growth

Phenotypes develop starting from a zygote placed in the center of a fixed size 2D grid. Morphogenesis proceeds in discrete developmental steps, during which the unique growth program is executed for each cell. The execution order is determined by position, proceeding from northeast to southwest.

Cells (see figure 5) are characterized by internal and external variables. Internal variables (metabolism) define the cell state and move with it, while external ones (chemicals) belong to the environment and follow a simple diffusion law. Diffusion is simulated by convolving the previous chemical concentrations with the following kernel:

$$\text{diffusion kernel} = \begin{vmatrix} 1/16 & 1/8 & 1/16 \\ 1/8 & 1/2 & 1/8 \\ 1/16 & 1/8 & 1/16 \end{vmatrix} \tag{3}$$

At each developmental step, existing active cells can release chemicals, change their own type, alter their internal metabolism and produce new cells. An active cell can also die or become passive. Each step, up to four new cells can be produced in any of the cardinal directions North, East, South and West (NESW). The mother cell specifies the new cell internal variables (type and metabolism) and whether they are active or passive. If necessary, existing cells are pushed

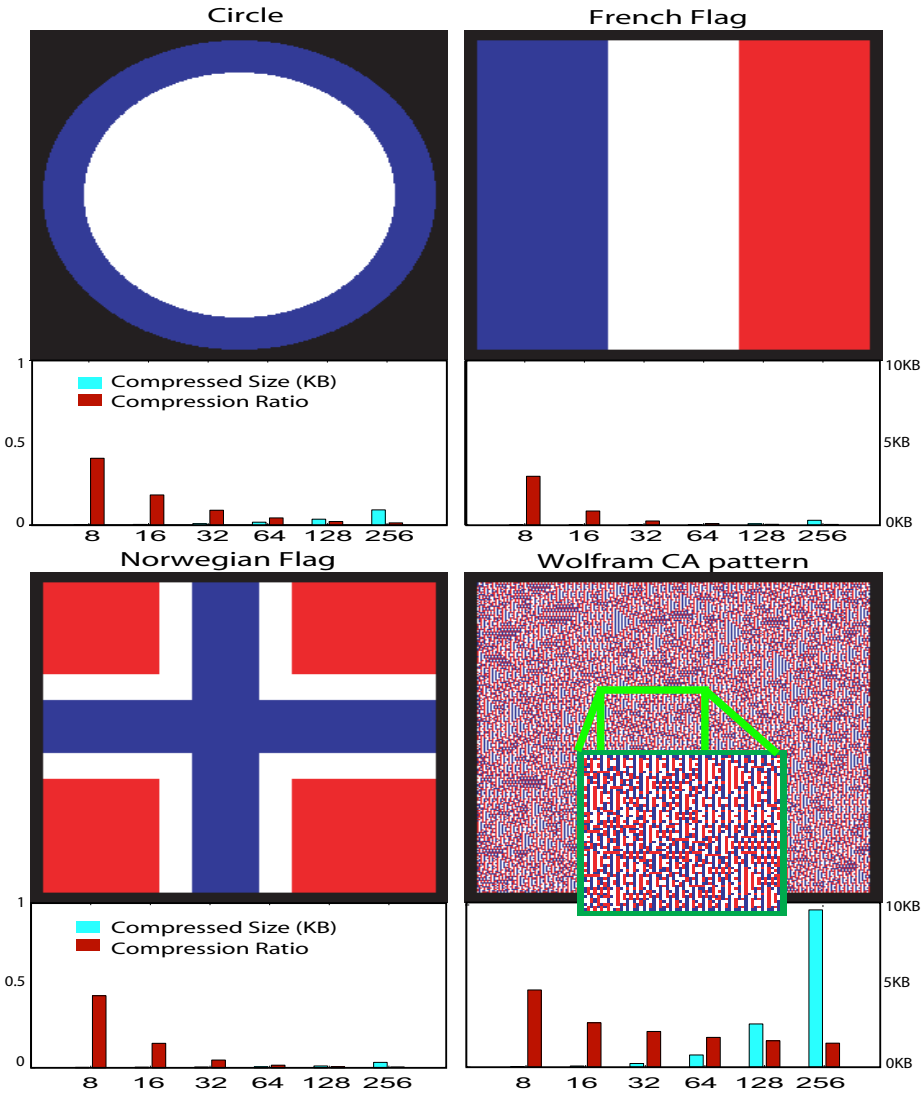


Figure 4: Shape of the evolutionary targets. A bounded circular pattern, a French flag, a Norwegian flag and a type 4 CA-generated pattern using the 3-color totalistic rule 1387 [35]. Below each image a graph reports the ARJ compression ratio (darker) and the size of the compressed data (lighter) for various target dimensions (from 8x8 to 256x256). The compression ratio can be seen as a measure of complexity, the size of compressed data as the measure of information. In the Wolfram CA pattern a zoom box is superimposed for clarity. The ARJ is a general purpose compression algorithm. Notice that each pattern also contains a frame, which also must be matched. The frame is an additional source of difficulty for the developmental system.

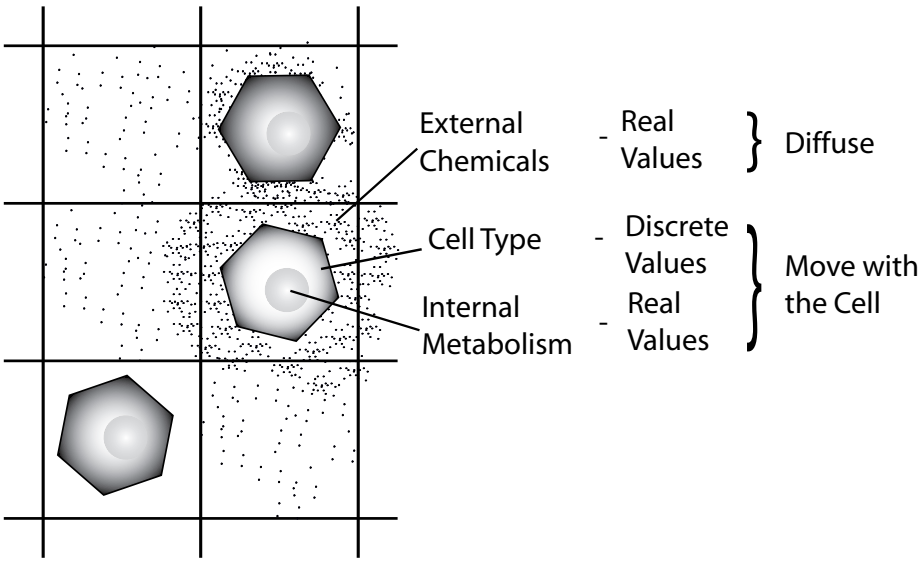


Figure 5: Description of the variables used for development. External chemicals follow a diffusion law, while internal ones move with the cell. While chemical concentrations, internal and external, take values in $[-1, 1]$, the cell type can take one of 4 discrete values, one for each phenotypic color.

sideways to create space for the new cells. When a cell is pushed outside the boundaries of the grid it is permanently lost.

Passive cells cannot release chemicals, change their state or produce new cells.

The cell behavior is governed by an Artificial Neural Network (Morpher) defined by the genotype. The genotype contains a floating point number for each synaptic weight. The Morpher receives as input the current cell internal and external variables, and the cell types of the neighboring cells in the four cardinal directions. Its output determines the new internal and external variables of the cell and, in case of replication, the internal variables of the newly generated cells. An additional local variable, the cell age, is set to 1 at birth and decays exponentially to 0.

The number of inputs and outputs to the growth program depends on the number of neighbors, cell types, external and internal chemical types (see figure 6). In most of the simulations presented, there are 4 cell types, 1 internal metabolism and 1 external chemical types. These sum up to 8 inputs (one being the bias) and 16 outputs which can take values $\in [-1, 1]$.

Chemical (production/concentration) and metabolism values are read directly to and from input and output lines. Since there is 4 possible cell types, each input/output line is quantized to four possible values $\{-1, -1/3, 1/3, 1\}$, each representing a different color in the phenotype: black, blue, red and white

respectively. When cell types are computed, the outputs take the nearest quantized value.

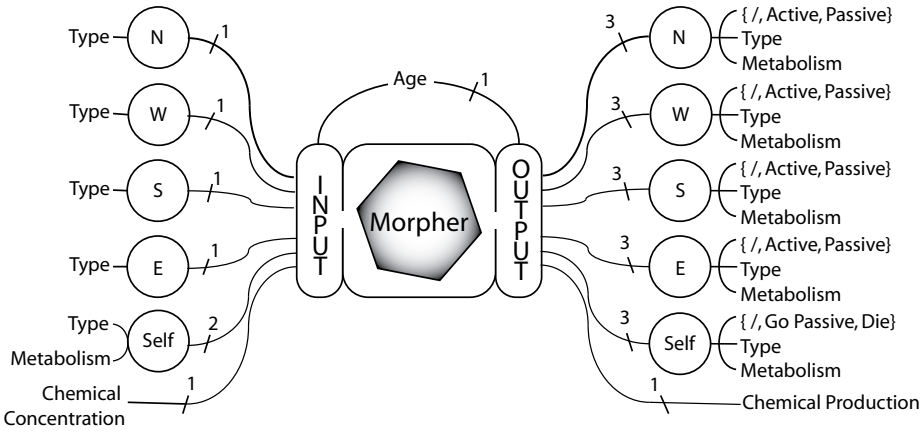


Figure 6: Inputs and outputs of the growth program, the Morpher, implemented with a feed-forward ANN. Each cell internal variables, cell's type and metabolism, implement a direct feedback pathway, while chemical production and diffusion offer a channel for longer range communication. A bias input neuron is also provided (not shown).

In picture 7, typical examples of phenotypes from random individuals are shown. The first phenotype (Random 1) is the most frequent.

5 Evolution for development and direct encoding

Every population is composed of 400 individuals. The best 1/8 survives and reproduces (elitism).

Fitness is proportional to the resemblance of an individual to the target, and is computed as shown in equation 4. For fitness computation, dead cells are assigned the default type 0 (black color)

$$\begin{aligned} \text{Fitness}(P,T) &= \left(\sum_{x,y} \text{Equals}\,(P,T,x,y) \cdot \text{TraitRarity}(x,y) \right) \;/\; ||T|| \\ \text{Equals}\,(P,T,x,y) &= \begin{cases} 0 & if \quad P(x,y) \neq T(x,y) \\ 1 & if \quad P(x,y) = T(x,y) \end{cases} \end{aligned} \tag{4}$$

where P is the phenotype, T the target, and $\text{TraitRarity}(x,y)$ is a measure of the rarity of the trait in $P(x,y)$ over the entire population.

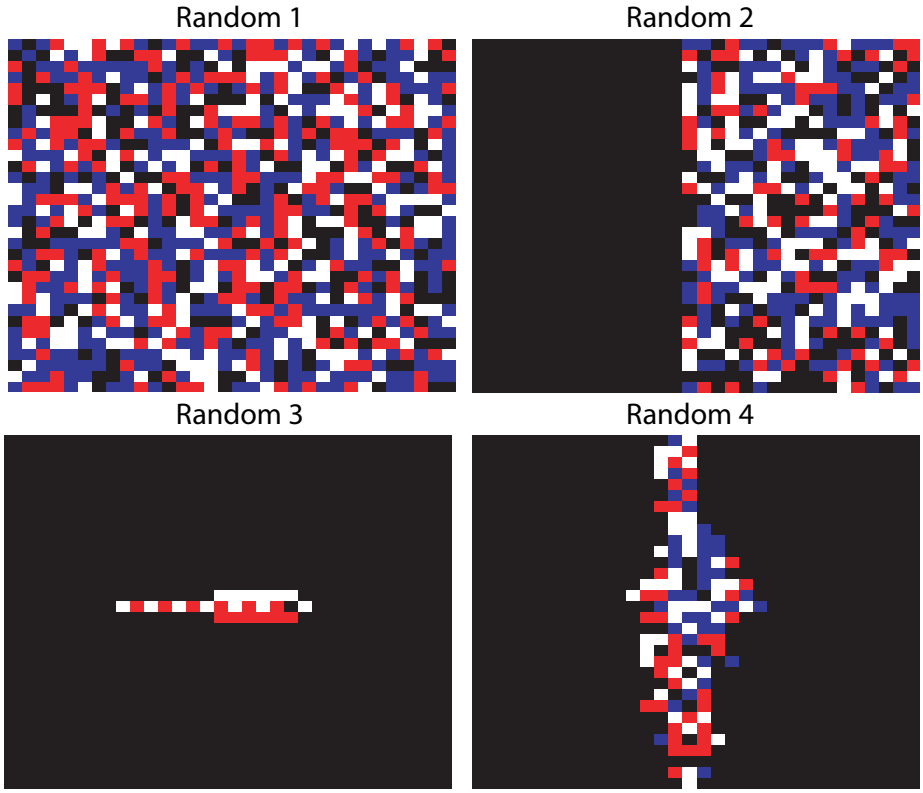


Figure 7: Typical phenotypes of first generation individuals. Dead cells have the same color of black cells.

TraitRarity is used to increase population diversity, counteracting homogenization and favoring individuals with rare characteristics. AE systems are in fact characterized by saltatory improvements (see figure 11). Under these conditions a positive innovation can increase the reproductive probability of a particular strain. TraitRarity helps reduce the crowding of a single genotype strain and is calculated as follows:
(1) The population is first ordered by fitness value before modification (in case of ties, younger individuals have priority). (2) Fitness scores are recomputed following this order, but the initial value of 1 of each trait (TraitRarity in Eq.4) decreases with use in steps of .1. Whenever a trait value goes to or below zero, it is assigned a value of 1/100.

The overall effect is that older individuals with common phenotype traits are replaced by younger, rarer individuals.

5.1 Mutation and Crossover

For direct encoding every gene (color) can mutate with a probability of 1/100 and 50% of the offspring are generated by single-point crossover.

In the case of AE, mutation takes each weight of the Morpher RNN with a P_{mut} probability and adds to it Gaussian noise with V_{mut} variance. P_{mut} and V_{mut} vary in the ranges $[1/20, 1/10]$ and $[1/30, 1/10]$ respectively. Their value is proportional to the time passed since the last increase in the top fitness score, reaching the maximum in ten generations. The reason behind these adaptive values is to increase exploration when the system begins to stagnate.

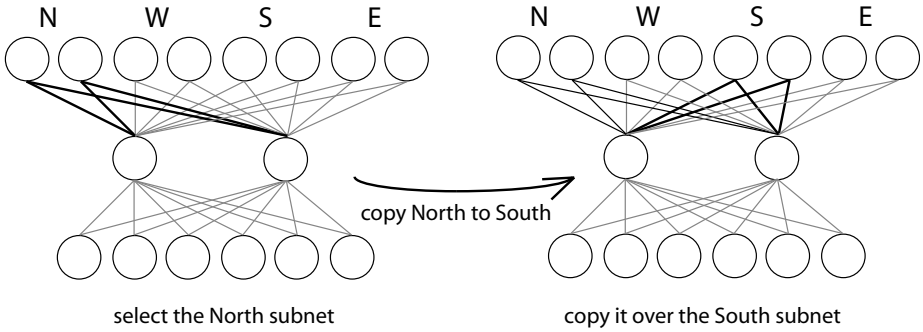


Figure 8: Symmetric mutation. The subnet responsible for the production of new cells in the north direction (on the left with a darker color) is copied over the subnet responsible for the south direction. Northward and southward cell production are now identical.

With a 1/20 probability an offspring undergoes an additional symmetric mutation (see Figure 8). The subnet responsible for the production of new cells in a chosen direction overwrites one or more of the other direction subnets. This operator should favor the evolution of phenotypes with various degrees of symmetry, but, since cells are not activated in parallel but follow an activation order, it does not produce perfect symmetrical phenotypes.

10% of the offspring are produced by crossover. If the Morpher RNN has a hidden layer, crossover shuffles the parents' hidden nodes (together with input and output weights). If no hidden nodes are present, it exchanges all the weights connected to inherited outputs units.

6 Results

The results present statistics over 20 independent evolutionary runs for each parameter setting (400 individuals and 2000 generations). Unless specified otherwise, the Morpher contains no hidden layers and each cells has 4 types, 1 internal and 1 external chemicals. Also, when more then one embryonal stage

is available, only the latest one is affected by the evolutionary operators. All other stages are fixed.

The results presented explore the following aspects of the multicellular development system:

i: Advantage of the embryonal stages
ii: Exploration strategies with embryonal stages
iii: Scalability of the growth model
iv: Performance of different Morpher configurations
v: Effects of the variables used by the growth program
vi: Emergent and evolved self-repair

The number of embryonal stages (N_{ES}) can be altered from a minimum of 1 to a maximum of one for every development step (devStep). The number of development steps (N_{dS}) has a strong influence on system evolvability: there must be enough steps to allow the emergence of good cell's configurations and not too many in order to minimize indirection (see section 2). Furthermore, the optimal N_{dS} often varies from target to target. In this paper we used the formula:

$$N_{dS} = 2 + \log_2(||\text{Target}||)$$

When a stage is available and the maximum fitness did not increase for a period of $1000/N_{dS}$ generations, a new stage is introduced. While the first stage is responsible for the ontogeny from the beginning of development. The i-th stage takes over at devStep $\big(N_{dS} - (N_{ES} - i)\big)$.

6.1 Advantage of the embryonal stages

16x16 and 32x32 targets (see figure 4) have been evolved with a single, half, and maximum number of stages (Group 0, $^1/_2$, 1 respectively). Their performance is shown in figures 9 and 10.

The average fitness scores demonstrate that embryonal stages are beneficial. Using an ANOVA test with $p = 10^{-3}$ the results always prove statistically significant for group 0. The difference between group $^1/_2$ and group 1 is only significant for the Wolfram CA targets, which are also the most complex.

With a single embryonal stage innovations require long periods of exploration and cause big phenotypic changes, more stages allows smaller and more frequent refinements (see figure 11). These results suggest that stages increase the correlation between genotype and phenotype.

6.2 Exploration strategies with embryonal stages

In most of the simulations presented in this paper, only the newest embryonal stage is modified by evolution. This strategy allows for a search space of constant size, with each stage increasing resolution around the current genotypes and excluding from search distal regions of the solution space.

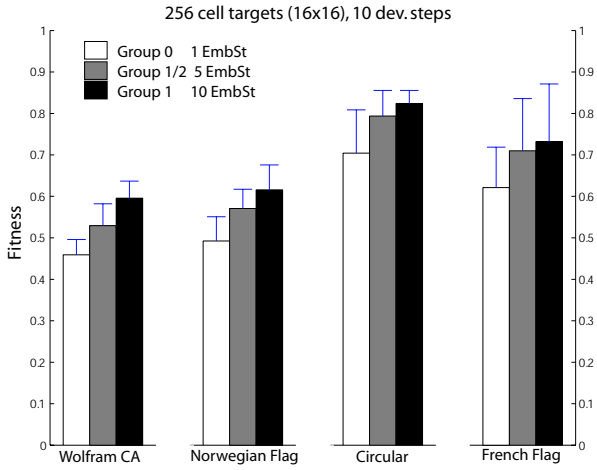


Figure 9: Effects of varying embryonal stages on performance. Mean and maximum performance from 20 runs for each of the 16x16 targets.

A premature introduction of embryonal stages restricts exploration and could affect overall performance.

To test this hypothesis, we allow every stage to be subjected to mutation. To offer a protection of older embryonal stages, the mutation variance (V_{mut} in Section 5.1) affecting the Morpher of a specific stage is reduced with age.

If the genotype currently contains N embryonal stages, the mutation variance affecting the Morpher of stage i is:

$$V_{mut}/(\text{variance decay}^{(N-i)})$$

thus, the older an embryonal stage the lower the mutation variance.

Figure 12 shows statistics from 5 different groups with various variance decay values. A value of 1 means that all stages change at the same rate.

The results show that, in some cases, locking old stages can reduce overall performance. The effect is more evident for the Norwegian flag target with variance decay values of 2 and 5. On the other hand, differences have no statistic significance ($p < 0.001$).

6.3 Scalability of the growth model

The most interesting property of developmental systems is the claim of high independence between evolvability and phenotypic size. Other papers compare the performance of development against direct encoding in specific tasks, for example [4, 16, 12]. Often though, the solution for the selected task can take advantage of particular features of the development system. This fails to highlight the trade-off between direct encoding and AE, often favoring the latter. In these cases, critics may argue that the better performance of AE derives from the exploitation of built-in implicit knowledge.

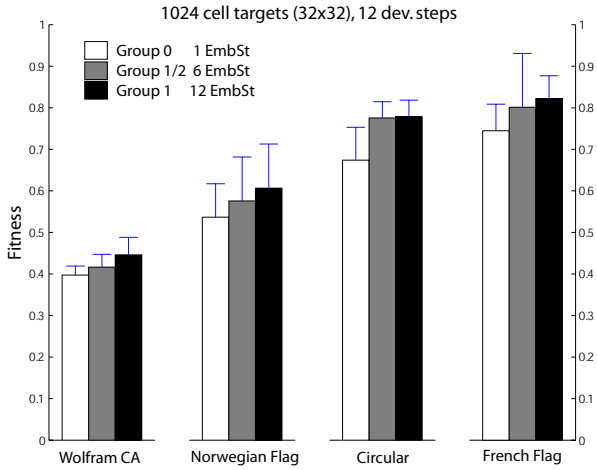


Figure 10: Effects of varying embryonal stages on performance. Mean and maximum performance from 20 runs for each of the 32x32 (right) targets.

Compared to previous work [28], we have tested more and bigger shapes. Also, the high complexity target presented in [28] contains mainly 2 colors/cell type, a property that could be exploited by minimalistic solutions.

The results for targets with 64, 256, 1024, 4096, 16384, 65536[4] cells are shown in figure 13. For direct encoding, performance is not influenced by the target shape, but for embryogeny the degrees of regularity have a great impact. The best evolved 256x256 individuals are plotted in figures 21-24.

Direct encoding steadily evolves perfect solutions for targets with up to 1024 cells. As expected, with bigger search spaces, performance decreases monotonically. This is not the case for development with embryonal stages. AE performance, after reaching a minimum for medium sized phenotypes, shows an increase and then levels off. The final fitness values appear to be proportional to the regularity of the specific target.

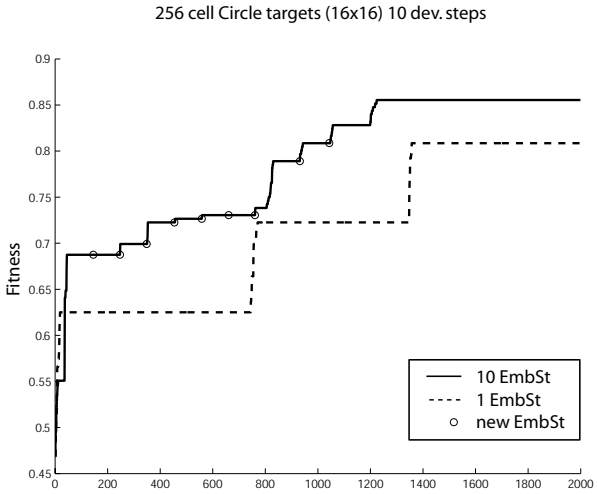
It is interesting to notice that, even the performance of the high complexity Wolfram CA patterns converges.

6.4 Performance of different Morpher configurations

It is possible to use various implementations of the growth program. The default model, a RNN with no hidden layers, has been selected for simplicity, evolvability and speed. In this section we have tested 32x32 targets with three different RNN topologies: with no hidden layers (NH), with 4 hidden nodes (H4) and 8 hidden nodes (H8). Results are plotted in figure 14.

It is possible to imagine that, given the complexity of the task, neural networks with more hidden units could better achieve an optimal growth program.

[4]for the 256x256 target, only 8 runs are available at the moment, others are still running



Fitness plots have saltatory characteristics. The introduction of new embryonal stages is frequently followed by a burst of fitness improvements. When evolving with one stage, improvements are less frequent and have bigger effects on the phenotype.

Figure 11: Maximum fitness plots of the best evolved populations for the 16x16 circular target from group 0 and group 1.

On the contrary, there is no significant difference ($P < 0.001$) among the various morpher configurations, as the higher representation flexibility is counterbalanced by bigger search spaces. These results advocate for growth program simplicity.

6.5 Effects of the variables used by the growth program

The growth program takes as input the neighbors cell types, its own internal metabolism state and the current chemical concentration at the cell's locus. To shed light on their role in development, the number of these variables can be altered.

In figure 15 populations were evolved with 0, 1 and 2 types of external chemicals. External chemicals diffuse in the environment allowing short range inter-cellular communication. Every external chemical adds one input and one output line to the growth program. Their number does not show a significant influence on evolvability.

In figure 16 populations were evolved with 0, 1 and 2 types of internal chemicals (cell metabolism). Internal chemicals implement the recurrent connections of the morpher. Every internal chemical adds one input and 5 output lines to the morpher. Also in this case, their number have no statistically significant influence on performance.

In figure 17 populations were evolved with 4 and 8 cell types. Additional types are redundant, so when 8 types are possible, every color is represented by 2 types. Additional cell types allow more information to be stored in the local neighborhood. Every additional 4 cell types requires 5 extra input and output lines. A statistically relevant performance increase is achieved only for

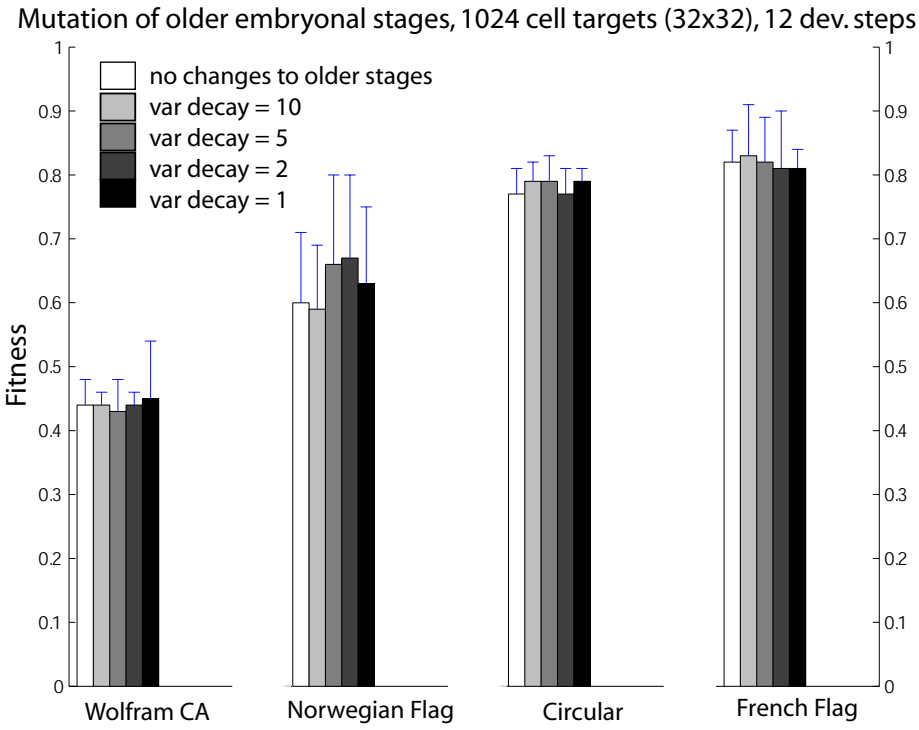


Figure 12: Average and maximum fitness scores for populations with different mutation strategies. On the contrary to other simulations presented, older embryonal stages are also affected by mutation. Higher values of variance decay (var decay, in figure) reduce the effects of mutation for older embryonal stages. Mutation affects the functionality of the Morpher.

the Wolfram CA pattern.

These results suggest that the exact number of variables used does not alter significantly the evolvability of development.

6.6 Emergent and Evolved self-repair

The regenerative capabilities of biological organisms have always been a source of inspiration for researchers. A very interesting property of this and similar multicellular development models is that they show emergent fault recovery [25, 26, 28]. In these cases, self-repair was not selected for and appears as an emergent byproduct of ontogeny. With each cell's behaviour based on the same growth program and only local variables, development achieves an intrinsic stability.

Self-repair is very interesting in the prospect of real world implementations, where random faults (e.g. radiation, fatigue) or substrate imperfections (e.g

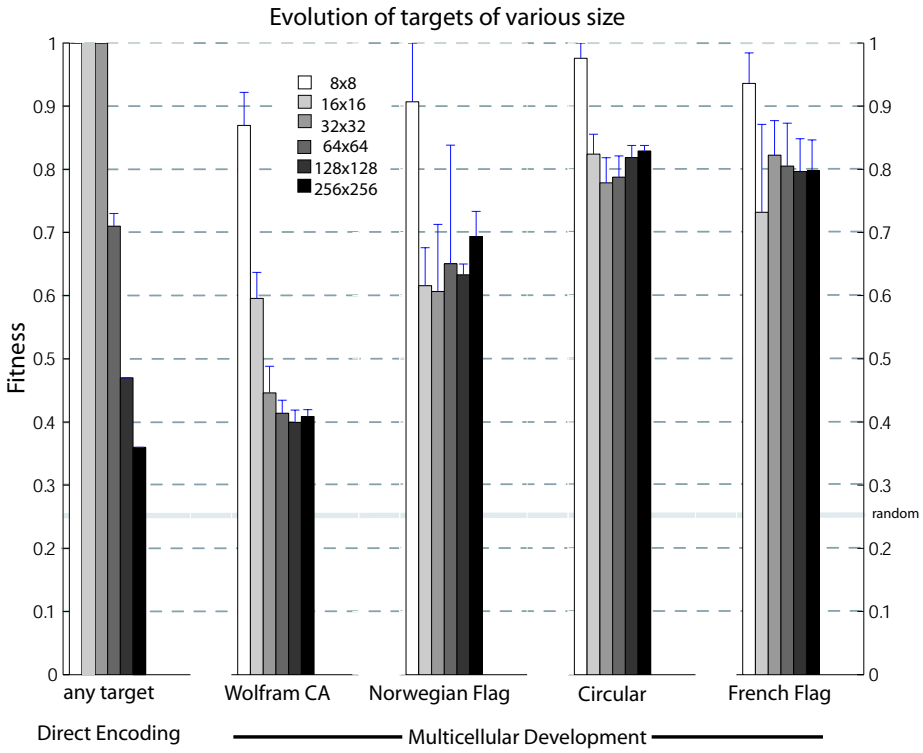


Figure 13: Average and maximum fitness scores for populations of various phenotypic size. Direct encoding performance decreases with bigger search spaces. On the contrary, multicellular development performance seems to converge. For comparison, the average performance of a random individual is also provided.

manufacture errors) are common. Contrary to traditional approaches, here fault-tolerance is achieved by regeneration without phenotypic redundancy.

In Figures 18-20, emergent and evolved fault tolerance are shown. To test the regenerative capabilities of the evolved organisms, the development of the best individuals has been repeated with random phenotypic faults, which cause the death of selected cells.

Each cell has been assigned a mortality rate $m_r \in \{0.005, 0.01, 0.05, 0.1\}$ per cell per developmental step. The probability for a cell sustaining a fault at any time during the development process (12 steps) is thus {0.06, 0.11, 0.46, 0.72} respectively.

Emergent fault tolerance (figure 18) reports the self-repair capabilities for individuals which were not selected for this characteristic.

On the contrary, evolved fault tolerance is achieved subjecting individuals to faults during evolution. Subjected to random faults, the stochastic growth process must be tested several times to guarantee a meaningful selection process.

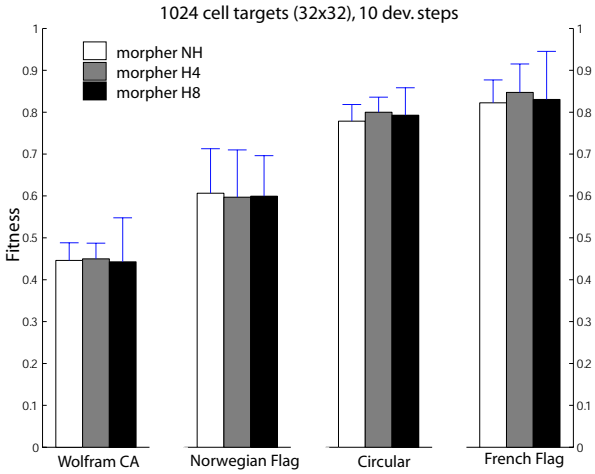


Figure 14: Average and maximum fitness scores for populations with different morphers: RNN without hidden layers (NH), and with one hidden layer of 4 (H4) and 8 (H8) neurons.

In particular, higher fault probability should require exponentially more repetitions. Since fitness evaluations are usually computationally expensive (for example in the evolution of robot controllers) here the number of repetitions is fixed to a reasonably small number. With $m_r \in \{0.01, 0.1\}$ fitness is computed as an average over only 5 independent growth processes.

Evolved individuals manage to reach high fitness scores and show a strong degree of fault tolerance.

7 Conclusions

We have tested a model of multicellular development showing that the method is both highly scalable and capable of evolving self-repairing organisms. As expected, performance is strongly dependent on the intrinsic regularity of the target. With larger search spaces the performance of the Direct Encoding (DE) steadily decreases, performance levels off for the presented Artificial Embryogeny (AE). This is valid also for the high complexity Wolfram CA target, even though both DE and AE performance is low for the largest phenotypes.

It is important to notice, that the presented test and especially the Wolfram CA target, are a 'worst case scenario' for the comparison of AE toward DE.

In fact, the problem of matching a string computing fitness as the Hamming distance to the target, is among the easiest for DE. Its difficulty is independent of the specific target's structure and it is equivalent to the One-Max problem.

For AE, difficulty is influenced by the target shape. In the general case it is not even clear whether a optimal solution exists. The fact that, even for

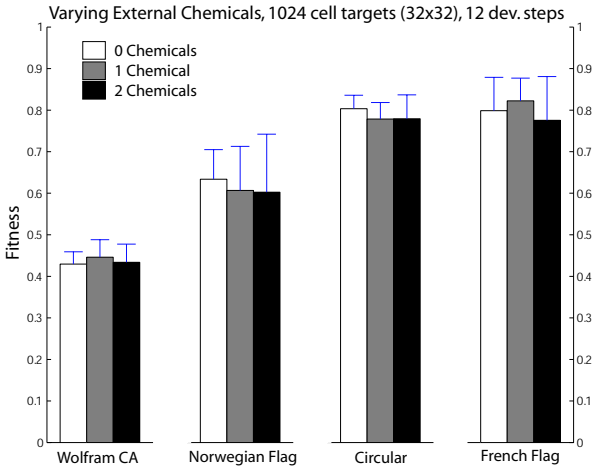


Figure 15: Maximum and average fitness scores from 32x32 targets with varying number of external chemical types. Differences are not statistically relevant ($p < 0.001$).

highly complex phenotypes, development ends up out-performing DE represents a strong evidence in favor of artificial embryogeny.

The analysis of random (Figure 4) and of evolved organisms (Figures 21-24) does not show developmental fingerprints, such as numerous repeated sub-patterns. These could prove a liability for the generality of the results. On the other hand, since cell genesis can only happen in the NWSE directions, horizontal and vertical sprouts are more common.

The development of organisms suffering stochastic cell deaths also shows good self-repair capabilities both when fault tolerance is neutral and selected for. Self-repair appears therefore as a by-product of ontogeny, with these results being consistent with those presented in [25, 26, 28].

The use of multiple embryonal stages has proved beneficial to evolution. Inspired upon gene duplication, development with embryonal stages implements a direct neutral complexification of an organism reducing the pleiotropy among different developmental phases. The method is very general and could be applied to any developmental model. While in general it is true that the more embryonal stages the better performance, on the other hand, if new stages are introduced too early during evolution they can be detrimental. When excluding older stages from search, new embryonal stages increase the resolution of proximal regions of the solution space, while locking distal ones. As a result, early introduction can show the drawbacks of over-exploitative search strategies. When all stages are allowed to evolve, these effects are weaker.

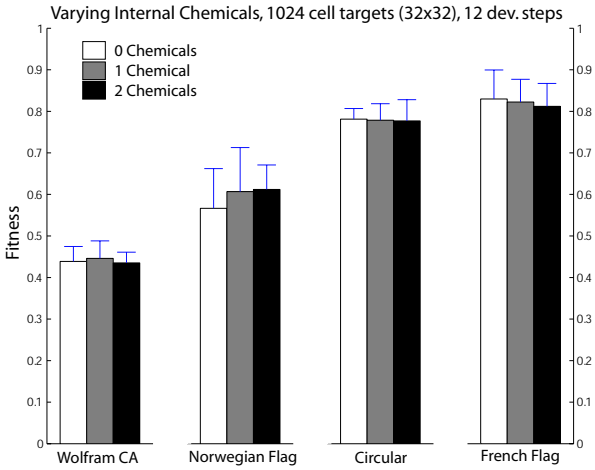


Figure 16: Maximum and average fitness scores from 32x32 targets with varying number of internal, metabolic, chemical types. Differences are not statistically relevant ($p < 0.001$).

7.1 Further Work

The evidence collected in this paper provides a good basis to start evolving more interesting artifacts. Preliminary results in this direction involve the evolution of locally connected spiking neural networks [13]. Given the high scalability of the model, future work will include the evolution of neuro-structures with functionality typical of peripheral brain regions, such as the retina and the motor cortex.

Also, embryonal stages provide neutral complexification only over the temporal domain. It is as well possible to design similar mechanisms operating on the spatial domain. Such method could prove particularly beneficial in the evolution of 3D organisms such as interconnected neural layers or body segments.

7.2 Computational Requirements

Locking older embryonal stages has a good effect on simulation speed. By caching old (fixed) stages, each new stages reduces the number of developmental steps required to produce a mature organism. Without this stratagem simulations are several times slower.

A single 256x256 cells evolutionary run may require 50 billions Morpher activations and take up to a week on a AMD XP 1800+ based computer. Without caching, the time needed would be 9 times as much. The over 1500 simulations presented have taken several weeks of computation time on the ClustIS Beowulf cluster [7].

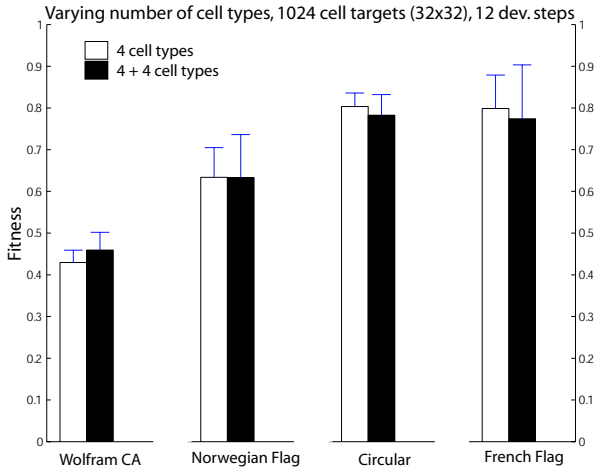


Figure 17: Maximum and average fitness scores from 32x32 targets with varying number of cell types. Differences are statistically relevant only for the Wolfram CA target ($p < 0.001$).

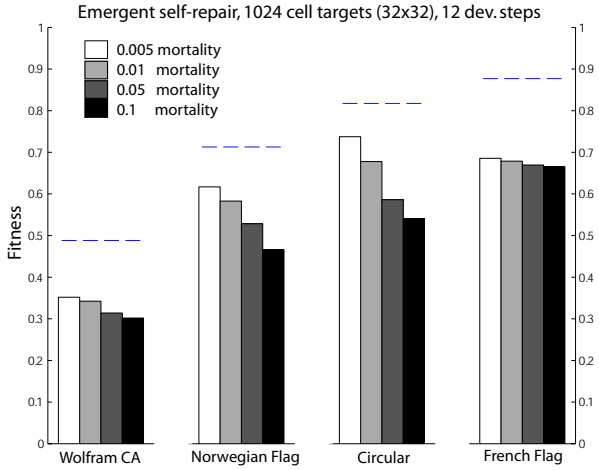


Figure 18: Emergent self-repair for the highest fitness 32x32 organisms with maximum number of embryonal stages (group 1). In this case, fault tolerance was not selected for and appears as a byproduct of ontogeny. Average performance over 100 runs with various mortality rates. The horizontal lines show the fitness score without faults.

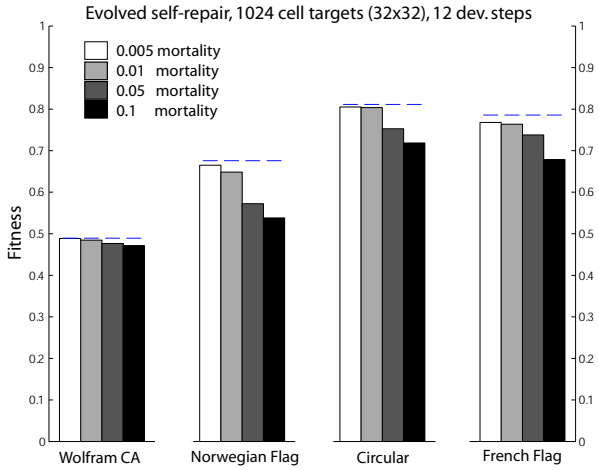


Figure 19: Evolved self-repair for 32x32 organisms with 12 EmbSt. Individuals evolved while subjected to a 0.01 fault probability, fitness being the average score over 5 runs. Average performance over 100 runs with various mortality rates. The horizontal lines show the fitness score without faults.

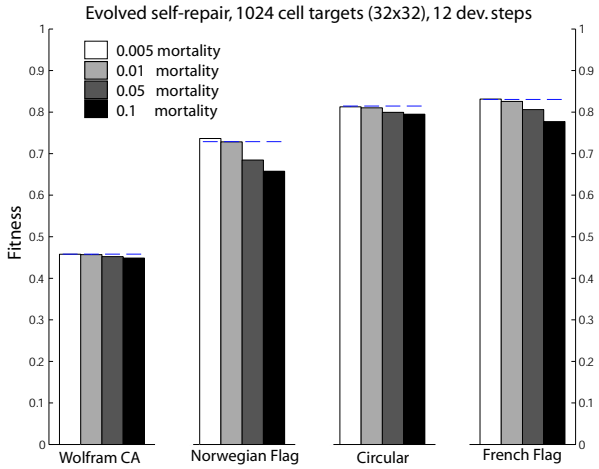


Figure 20: Evolved self-repair for 32x32 organisms with 12 EmbSt. Individuals evolved while subjected to a 0.1 fault probability, fitness being the average score over 5 runs. Average performance over 100 runs with various mortality rates. The horizontal lines show the fitness score without faults.

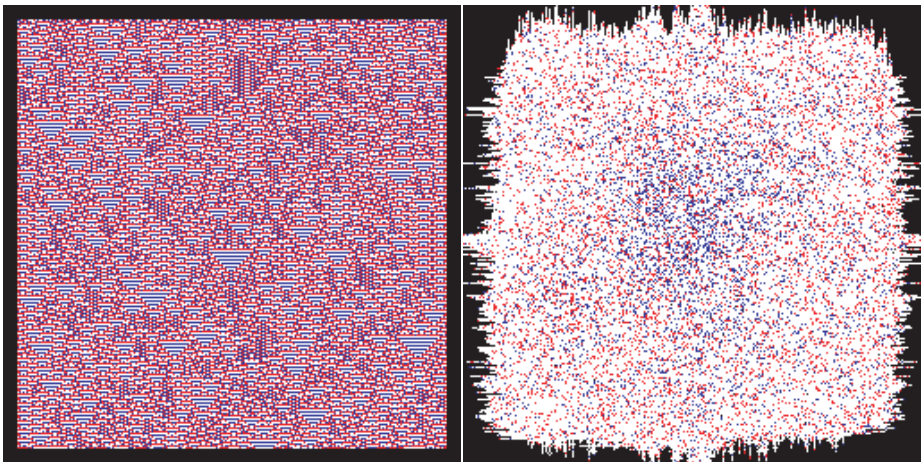


Figure 21: Wolfram CA target (left) and best evolved 256x256 individual (right).

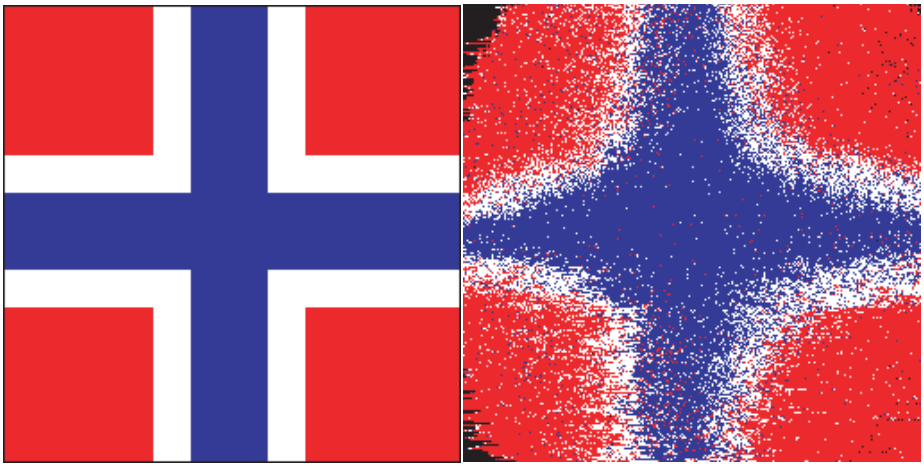


Figure 22: Norwegian Flag target (left) and best evolved 256x256 individual (right).

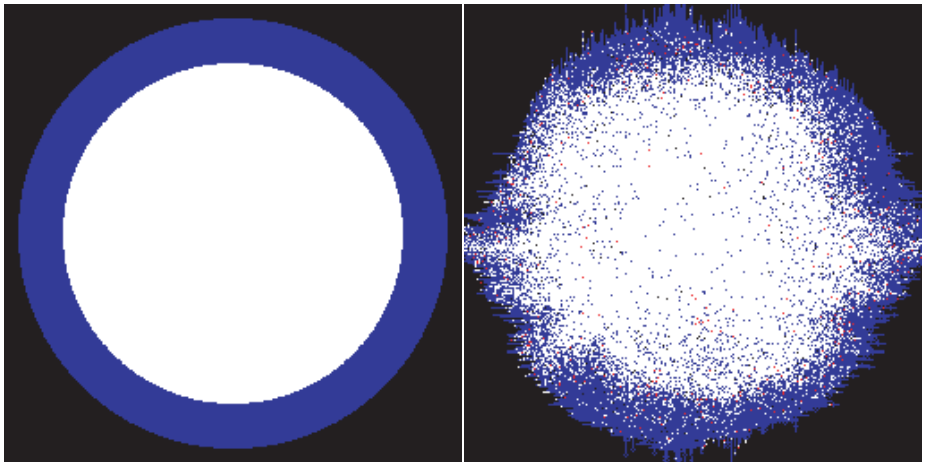


Figure 23: Circular target (left) and best evolved 256x256 individual (right).

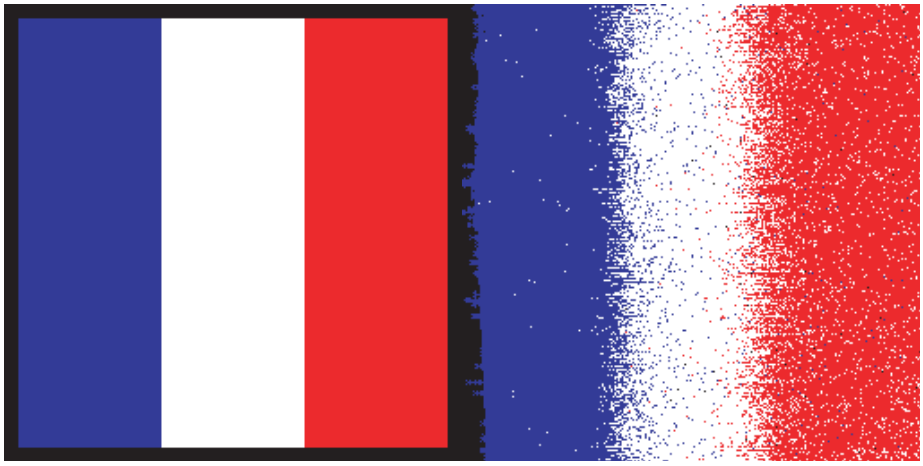


Figure 24: French Flag target (left) and best evolved 256x256 individual (right).

References

[1] W. Banzhaf. Genotype-phenotype-mapping and neutral variation - a case study in genetic programming. In *PPSN III: Proceedings of the International Conference on Evolutionary Computation. The Third Conference on Parallel Problem Solving from Nature*, pages 322–332. Springer-Verlag, 1994.

[2] Y. Bar-Yam. *Dynamics of Complex Systems*. The Advanced Book Studies in Nonlinearity. Westview Press, 1997.

[3] P.J. Bentley. Evolving fractal gene regulatory networks for robot control. In Wolfgang Banzhaf, Jens Ziegler, and Thomas Christaller, editors, *Proceeding of the European Congress of Artificial Life, ECAL 2003*, volume 2801/2003, pages 753–762. Springer-Verlag, 2003.

[4] P.J. Bentley and S. Kumar. Three ways to grow designs: A comparison of embryogenies for an evolutionary design problem. In Wolfgang Banzhaf, Jason Daida, Agoston E. Eiben, Max H. Garzon, Vasant Honavar, Mark Jakiela, and Robert E. Smith, editors, *Proceedings of the Genetic and Evolutionary Computation Conference*, volume 1, pages 35–43, Orlando, Florida, USA, 13-17 1999. Morgan Kaufmann.

[5] J. Bongard. Evolving modular genetic regulatory networks. In *Proceedings of the 2002 Congress on Evolutionary Computation (CEC2002)*, pages 1872–1877. IEEE Press, Piscataway, NJ, 2002, 2002.

[6] A. Cangelosi, S. Nolfi, and D. Parisi. Cell division and migration in a 'genotype' for neural networks. *Network: Computation in Neural Systems*, 5:497–515, 1994.

[7] J. Cassens and Z. Constantinescu Fülöp. It's magic: Sourcemage gnu/linux as hpc cluster os. In *in Proceedings Linuxtag 2003*, 2003.

[8] F. Dellaert and R. Beer. A developmental model for the evolution of complete autonomous agents. In Pattie Maes, Maja J. Mataric, Jean-Arcady Meyer, Jordan Pollack, and S W. Wilson, editors, *From Animals To Animats 4: SAB 1996*, pages 393–401, 1996.

[9] F. Dellaert and R.D. Beer. Toward an evolvable model of development for autonomous agent synthesis. In R.Brooks and P. Maes, editors, *Proceedings of Artificial Life IV*, pages 246–257. MIT Press Cambridge, 1994.

[10] J. Reinitz E. Mjolsness, D.H. Sharp. A connectionist model of development. *Journal of Theoretical Biology*, 152(4):429–453, 1991.

[11] P. Eggenbergen-Hotz. Evolving morphologies of simulated 3d organisms based on differential gene expression. In Phil Husbands and Inman Harvey, editors, *Proceedings of the 4th European Conference on Artificial Life (ECAL97)*, 1997.

[12] P. Eggenbergen-Hotz. Comparing direct and developmental encoding schemes in artificial evolution: A case study in evolving lens shapes. In *Proceeding of the Congress on Evolutionary Computation, CEC 2004*, pages 752–757, 2004.

[13] D. Federici. Evolving a neurocontroller through a process of embryogeny. In S. Schaal, AJ Ijspeert, A. Billard, S. Vijayakumar, J. Hallam, and JA Meyer, editors, *From Animals To Animats 8: SAB 2004*, pages 373–384, 2004.

[14] F. Gruau. *Neural Network Synthesis using Cellular Encoding and the Genetic Algorithm.* PhD thesis, Ecole Normale Superieure de Lyon, 1994.

[15] I. Harvey. Artificial evolution: A continuing saga. In Takashi Gomi, editor, *Proceedings of 8th Intl. Symposium on Evolutionary Robotics, ER2001*, pages 94–109. Springer-Verlag, 2001.

[16] G.S. Hornby and J.B. Pollack. The advantages of generative grammatical encodings for physical design. In *Proceedings of the 2001 Congress on Evolutionary Computation, CEC 2001*, pages 600–607. IEEE Press, 27-30 2001.

[17] G.S. Hornby and J.B. Pollack. Body-brain co-evolution using L-systems as a generative encoding. In Lee Spector, Erik D. Goodman, Annie Wu, W. B. Langdon, Hans-Michael Voigt, Mitsuo Gen, Sandip Sen, Marco Dorigo, Shahram Pezeshk, Max H. Garzon, and Edmund Burke, editors, *Proceedings of the Genetic and Evolutionary Computation Conference, GECCO-2001*, pages 868–875. Morgan Kaufmann, 7-11 2001.

[18] N. Jacobi. Harnessing morphogenesis. In P. Bentley and S. Kumar, editors, *On Growth, Form and Computers.* Academic Press, 2003.

[19] S.A. Kauffman. *The origins of order.* Oxford University Press, New York, 1993.

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

[21] P.K. Lehre and P. C Haddow. Developmental mappings and phenotypic complexity. In Ruhul Sarker, Robert Reynolds, Hussein Abbass, Kay Chen Tan, Bob McKay, Daryl Essam, and Tom Gedeon, editors, *Proceeding of the Congress on Evolutionary Computation, CEC 2003*, 2003.

[22] S. Luke and L. Spector. Evolving graphs and networks with edge encoding: Preliminary report. In John R. Koza, editor, *Late Breaking Papers at the Genetic Programming 1996 Conference*, pages 117–124, 1996.

[23] C. Mattiussi and D. Floreano. Connecting transistors and proteins. In J.Pollack, M. Bedau, P. Husbands, T. Ikegami, and R. Watson, editors, *ALife9: Proceedings of the Ninth International Conference on Artificial Life*, pages 9–20, 2004.

[24] ARJ Software Inc. http://www.arjsoftware.com. ARJ specifications.

[25] J.F. Miller. Evolving developmental programs for adaptation, morphogenesys, and self-repair. In Wolfgang Banzhaf, Jens Ziegler, and Thomas Christaller, editors, *Proceeding of the European Congress of Artificial Life, ECAL 2003*, pages 256–265, 2003.

[26] J.F. Miller. Evolving a self-repairing, self-regulating, french flag organism. In Kalyanmoy Deb, Riccardo Poli, Wolfgang Banzhaf, Hans-Georg Beyer, Edmund K. Burke, Paul J. Darwen, Dipankar Dasgupta, Dario Floreano, James A. Foster, Mark Harman, Owen Holland, Pier Luca Lanzi, Lee Spector, Andrea Tettamanzi, Dirk Thierens, and Andrew M. Tyrrell, editors, *Proceeding of Genetic and Evolutionary Compuation, GECCO 2004*, pages 129–139, 2004.

[27] S. Ohno. *Evolution by Gene Duplication*. Springer, 1970.

[28] D. Roggen and D. Federici. Multi-cellular development: is there scalability and robustness to gain? In Xin Yao, E. Burke, J.A. Lozano, and al., editors, *proceedings of Parallel Problem Solving from Nature 8, PPSN 2004*, pages 391–400, 2004.

[29] K. Stanley and R. Miikulainen. A taxonomy for artificial embryogeny. *Artificial Life*, 9(2):93–130, 2003.

[30] K. Stanley and R. Miikulainen. Competitive coevolution through evolutionary complexification. *Journal of Artificial Intelligence Research*, 21:63–100, 2004.

[31] S.A. Teichmann and M.M. Babu. Gene regulatory network growth by duplication. *Nature Genetics*, 36(5):492–6, 2004.

[32] G. Tufte and P. C Haddow. Building knowledge into development rules for circuit design. In Torresen Tyrrell, Haddow, editor, *Proceedings of the International Conference on Evolvable Systems, ICES 2003*, pages 117–128. Springer-Verlag, 2003.

[33] A.M. Tyrrell, E. Sanchez, D. Floreano, G. Tempesti, D. Mange, J.M. Moreno, J. Rosenberg, and A. Villa. Poetic tissue: An integrated architecture for bio-inspired hardware. In Torresen Tyrrell, Haddow, editor, *Proceedings of the International Conference on Evolvable Systems, ICES 2003*, pages 129–140. Springer-Verlag, 2003.

[34] Wikipedia. Ontogeny and phylogeny. *http://en2.wikipedia.org/wiki/Ontogeny_and_phylogeny*, 2003.

[35] S. Wolfram, editor. *A new kind of science*. Wolfram Media, 2002.

[36] J Zhang. Evolution by gene duplication: an update. *Trends in Ecology and Evolution*, 18(6):192–198, 2003.