THE GENETIC ALGORITHM AND BIOLOGICAL DEVELOPMENT

by

Stewart W. Wilson

The Rowland Institute for Science, Cambridge MA 02142

Abstract

A representation for biological development is described for simulating the evolution of simple multicellular systems using the genetic algorithm. The representation consists of a set of production-like growth rules constituting the genotype, together with the method of executing the rules to produce the phenotype. Examples of development in 1-dimensional creatures are given.

1. Introduction

The genetic algorithm [1] incorporates mechanisms which resemble the mechanisms of reproduction, variation, and selection found in natural evolution, but, despite successes in several fields of application, there has been little attempt to use the algorithm as a tool to investigate, through simulation, natural evolution itself. Considerable work exists on the ontogenetic evolution of behavior, *i.e.*, learning [2-4], but relatively little on the evolution of organisms *per se* [5]. The main reason has been the absence of representations for organisms which would permit the genetic algorithm to be brought to bear. The genetic algorithm observes the genotype-phenotype distinction of biology: the algorithm's variation operators act on the genotype and its selection mechanisms apply to the phenotype. In biology, the genotype-phenotype difference is vast: the genotype is embodied in the chromosomes whereas the phenotype is the whole organism that expresses the chromosomal information. The complex decoding process that leads from one to the other is called biological development and is essential if the genotype is to be evaluated by the environment. Thus to apply the genetic algorithm to natural evolution calls for a representational scheme that both permits application of the algorithm's operators to the genotype and also defines how, based on the genotype, organisms are to be "grown", *i.e.*, their development.

The present paper outlines a few steps in the direction of such a representation [6]. The problem is addressed at the level of cells, which are treated as as "black boxes" having well-defined properties. Beginning with the fertilized egg, the cells are to divide, move, and differentiate under the control of rules so as eventually to form a mature organism. An attempt is made to respect major facts known about cells and these processes, but large compromises must occur at this point in the effort to approach algorithmic workability. The principal objective is to describe a representational framework—a sort of "developmental automaton"—sufficiently completely that randomly generated instances will grow and can be evolved under the genetic algorithm in computer experiments.

2. Evolution of Development

The problem of applying the genetic algorithm to the development of multi-cellular organisms can be divided into four parts: plan, expression, selection, and variation.

2.1 Plan

In nature, the genotype contains (1) information that is descriptive, through the action of development and the environment, of a range of possible phenotypes, and (2) information encoding the developmental process itself, *i.e.*, how to go about making a phenotype from a genotype. Both kinds of information are of course inherited and subject to variation and natural selection. Here, for simplicity, it will be assumed that only the first kind of information, termed the organism's *plan*, is heritable and subject to the genetic algorithm. The other kind, the rules for *expressing* the plan to form the phenotype, will be regarded as fixed.

What should the plan look like? Several observations on natural systems are suggestive [7]. In the first place, though individual cells can have different sizes and can change in size, growth occurs primarily through cell division: one cell becomes two "daughter" cells. Second, depending on the situation, the daughters can be phenotypically the same as the parent, they can differ from the parent but not differ from each other, or they can differ from the parent and from each other. Third, the phenotypical outcome of cell division can depend not only on the nature of the parent cell, but also on factors related to the cellular, chemical, or physical context in which the parent cell is embedded. Finally—and pivotal for this discussion—all cells in an organism are considered to contain the *same* genetic informa-

tion, though some of it may become in some sense "switched off" or inoperative during differentiation.

These observations have suggested the following working proposal. The plan will take the form of a so-called *production system program* (PSP) consisting of a finite number of production (condition-action) rules which will be termed *growth rules*. The growth rules have the general form

$$X + K_i \Rightarrow K_j\ K_k$$

The K's stand for cell phenotypes and X represents the local context; the symbol "+" means conjunction. In addition, each growth rule has associated with it a weight w. Every cell in the organism contains the same set of rules, or PSP.

Focussing attention on a particular rule in the PSP of a particular cell, the condition side of the rule is satisfied if that cell is (phenotypically) of type K_i and the context matches X. The action recommended by the rule is to replace the cell by two new cells, one phenotypically of type K_j, the other of type K_k. Whether or not this rule controls the parent cell's fate depends on whether the rule is selected for expression, as discussed in the next section.

The general growth rule form is open to many special cases. As in nature, the daughter cells may or may not be the same as the parent or each other. Furthermore, some rules may contain just one daughter cell, identical to the parent; such a rule, if expressed, means that cell division *does not* take place. Also, some rules may have no cell in their action parts, corresponding to dissolution of the parent cell.

Some rules may have no term corresponding to X; their condition is satisfied independent of context. In the other rules, X can take on several forms. Most simply, X can stand for the presence of a cell of a particular kind adjacent to K_i. In this case ("adjacency" type context), the spatial relation of the X cell and K_i may affect the spatial relation of the daughter cells (if there are two). Another kind of X ("signal" type) would stand for a detector for signals emitted by other cells, not necessarily in the immediate neighborhood. For present purposes, the "signal" emitted by a cell is simply a list of its phenotypical properties. The predominant direction from which matched signals are received could affect the daughter cells' spatial relation. Still another kind of X would detect aspects of the physical environment such as intercell pressure.

2.2 Expression

Since all cells contain the same "program", differential development of the system depends on the selection for expression of different rules in different cells. This is not difficult in principle, since once some differentiation occurs, the sensitivity of the rules to cell type and context will lead to further differentiation. The proposed expression mechanism consists of a *match* step and a *decision* step. Again focussing attention on a particular cell, in the match step the cell first identifies those program rules which have satisfied conditions. Then, from this *match set*, the cell chooses a single rule for expression. The chosen rule "carries out its right-hand side", *i.e.*, daughter cells are produced as prescribed and their signals are emitted.

The system's growth process is envisioned as a series of discrete time-steps. On each step, the expression mechanism operates in every cell of the current system. The operation is regarded as "parallel" in the sense that offspring of all the cells are produced simultaneously. The offspring cells then undergo, in accordance with their phenotypical properties, a process of interaction and spatial accomodation so as to form the "new" system to be input to the expression mechanism in the next time-step.

2.2.1 The decision step

The decision step of the expression mechanism makes use of the growth rule weights w and the effect of signals from nearby cells. Each growth rule in the match set has an associated weight w. If a rule's context (X) part is either absent or is of adjacency type, its *excitation* is defined to be just w. However, if a rule's context part is of signal type, the rule's excitation is defined to be the product of the weight w and the *intensity* of the received context signal. For example, suppose that a certain rule has an X which matches signals S_A emitted by nearby cells A. Suppose further that the total intensity of the signals is simply their number n_A times a constant f. Then the excitation of the rule in question would equal $fn_A w$.

The cell decides which match set rule to express using a probability distribution over the rules' excitations. That is, the probability that a particular rule will be picked is equal to its excitation divided by the sum of the excitations of the rules in the match set. The following three rules offer an interesting example.

$$\begin{aligned} A &\Rightarrow A\ A & w_r \\ (S_A) + A &\Rightarrow A & w_i \\ (S_A) + A &\Rightarrow 0 & w_d \end{aligned}$$

The first rule, termed "reproductive", takes one cell A and leaves two in its place. The second rule,

termed "inhibitory", matches cell A, senses the presence of at least one A-type signal in the vicinity, and seeks, if chosen, to maintain the status quo exactly. The third rule, a deletion rule, has the same condition as the inhibitory rule, but seeks to delete the matched A cell. Each rule has a weight, as shown.

Suppose now the system consists of an aggregate of n cells of type A. In any cell, the excitations of the three rules will be:

$$e_r = w_r$$
$$e_i = w_i f n_{\mathrm{A}}$$
$$e_d = w_d f n_{\mathrm{A}} \quad .$$

If w_r is large and there are relatively few cells, the reproductive rule will be chosen most of the time and the aggregate will grow. As it does, however, the excitations of the inhibitory and deletion rules will increase relative to that of the reproductive rule, due to n_{A}. The growth rate will slow down. Eventually, an equilibrium will be reached where net growth is zero. At that point, the probability of reproduction equals the probability of deletion, or $w_r = w_d f n_{\mathrm{A}}$. Solving for n_{A} yields the system's equilibrium size:

$$n^*_{\mathrm{A}} = (1/f)(w_r/w_d) \quad .$$

The system's net growth rate dn/dt prior to equilibrium can be calculated by taking the product of n and the difference between the probabilities of reproduction and deletion. Dropping the "A" subscripts, the result is

$$\frac{dn}{dt} = n\frac{(1 - n/n^*)}{1 + (w_i/w_d + 1)(n/n^*)} \quad ,$$

showing that the system's growth rate can be "chosen" independently of its equilibrium size.

Though simple, the example is important because it illustrates one way in which the cellular program can manage the fundamental problem of bounded growth. Later examples of differentiation into finite regions of homogeneous cell type will assume the presence of growth rule sets of this or similar sort for the regions.

2.2.2 Phenotype properties

Once the decision step has picked a rule for expression, the daughter cells in the action part must be simulated, which means simulating their properties. In a real organism, each cell "type" has a myriad of physical and biochemical properties. Some of these may be more properly regarded as behavioral, *e.g.*, during development, cells can creep, amoeba-like to new positions. Most of the properties affect in one way or another a cell's interactions with other cells. Even if all the properties were understood, a realistic simulation would still have an enormous problem adequately representing and computing the interactions within the cell aggregate. Such a computation is necessary in order to determine the fitness, with respect to an environment, of the organism as a whole. The practical course for the present would seem to be to choose extremely simple environments, simple measures of fitness, and a very restricted range of cell properties.

2.3 Selection and variation

Because the foregoing representational framework for development takes the form of a production system program, it is straightforward to apply the genetic algorithm as the "engine" of phenotype selection and genotype variation. The application of the algorithm would be along the lines of previous work with production system programs [3,8]. One would start with a population of "egg" cells, each containing a random genotype. Each egg would undergo development and, after a standard number of time-steps, each resulting cell aggregate would be rated for fitness. The original eggs would then be copied in numbers proportional to these fitnesses to form a new population of the same size. Genetic operators would be applied to the genotypes of the new population. The cycle would be iterated through some number of generations, corresponding to evolution.

Many aspects of this scheme are quite well understood due to the research just cited and on genetic algorithms in general. However, the form of the growth rules in the genotype is somewhat unusual so some comments about coding are in order. The basic encoding would resemble that of classifiers [2]. The condition part of a rule would consist of a context taxon (for X) and a cell taxon (for K_i), each being a string of length L from $\{1, 0, \#\}$. The action part would consist of two *cell descriptions* (for K_j and K_k), both strings of length L from $\{1, 0\}$.

An *interpreter* is required to relate cell description encodings to phenotypical properties. This simply means establishing a pre-defined mapping between substrings in the cell description and properties, *e.g.*, "110" in the 14th through 16th positions could mean the cell surface has "high stickiness", etc. To take care of rules in which one or both of the daughter cells is absent, the interpreter would simply check the setting a certain bit in each cell description: "0", say, would mean that description cell was absent and the rest of the description should

be ignored. A similar system would be used to indicate the presence or absence of a context taxon and its type (adjacency, signal, or other).

A growth rule's condition would be satisfied if both (1) the cell description of the cell in which the rule finds itself matches the rule's cell taxon, and (2) at least one signal reaching the cell matches the rule's context taxon. The meaning of "match" is the same as for classifiers: the two strings must be the same at every non-# position of the taxon. The use of the "don't care" symbol # permits rule conditions to restrict their sensitivity to particular subsets of cell description and signal bits.

Calculation of the intensity of the signal matching the context taxon can be quite complex, depending on the simulation. Involved are the dependence of individual signal intensities on the distance from their sources, and also perhaps propagation delays with respect to the time-step of creation of the source cell. These factors must be predefined. In any case the total received intensity would be a sum over the individual intensities of all matched signals. As noted earlier, the net *direction* of the received signal may in some rules determine the spatial orientation of the daughter cells. The dependence would be encoded in a special bit string associated with the daughter cell descriptions.

The weight associated with a growth rule must also be encoded in order to make it, and consequently the rule's influence in the decision step, subject to the genetic algorithm. The weight would simply be concatenated, as a fixed-length binary number, with the rest of the rule string.

3. 1-D Development

As has been the case with research on cellular automata [9], the complexity of realistic three-dimensional simulations recommends initial study of one-dimensional examples. In two and three dimensions, forces between cells must lead to complicated cell movements and contortions of the "tissue". A 1-D "creature", however, could be viewed as growing inside a frictionless tube, with no forces except between adjacent cells. Cell division would lengthen the creature; deletion would shorten it. Though simple, the 1-D case can exhibit cell type configuration patterns such as symmetry, periodicity, and polarity that are analogous to patterns emerging in the development of real organisms. Some elementary examples follow.

3.1 Symmetry and periodicity

Changes in a 1-D system through time can be represented by a pyramid like the following:

```
    A
  B B
D C C D
```

This shows three time-steps. At first, the system consists just of cell A; then A divides to form cells B and B; then the left-hand B divides to form the (oriented) pair D C, and the right-hand B yields the pair C D. Only two growth rules are required:

$$A \Rightarrow B\ B$$

$$(B) + B \Rightarrow C\ D$$

In the second rule the context taxon is of adjacency type (indicated by the absence of "*S*"). This type of rule reads: "order the output cells so that the direction from the first one to the second one is the same as the direction from the context cell to the replaced cell."

Note that the pyramid diagram shows bilateral symmetry about its center line. Using additional rule sets of the self-limiting form discussed in Section 2.2.1, the C's and D's could be multiplied to yield eventually a stable symmetrical creature of finite size, D...D C...C D...D, with approximately equal groups of D cells.

The following pyramid and its rules illustrates rudimentary periodicity:

```
    A
  E F
C D C D
```

$$A \Rightarrow E\ F$$

$$(F) + E \Rightarrow D\ C$$

$$(E) + F \Rightarrow C\ D$$

Again, the addition of self-limiting rule sets would result in the creature, C...C D...D C...C D...D, in which like cell groups were approximately equal in size. It is clear that quite complex structures can be built by first establishing the pattern with non-cyclic rules (in which the cell taxon will not match the output cells), and then using self-limiting rule sets which apply to the final cell types.

3.2 Polarity

An elementary polarity results from any rule in which the output cell types differ. A polarity with respect to some phenotypical *property* can be set up with non-cyclic rules as follows:

```
    A
  B C
D E F G
```

$$A \Rightarrow B\ C$$

$$(C) + B \Rightarrow E\ D$$

$$(B) + C \Rightarrow F\ G$$

If in the cell descriptions of D, E, F, and G the property is, say, monotonically increasing, the amount of

the property will be graded across the system. A more sophisticated gradient system occurs under the rules:

$$\begin{aligned} A &\Rightarrow B\ C \\ (S_B) + C &\Rightarrow D\ C \\ C &\Rightarrow C\ C \\ (S_C) + C &\Rightarrow C \\ (S_C) + C &\Rightarrow 0 \end{aligned}$$

If B's signal loses intensity with distance, the probability that a C will change to D C will fall with distance from the left end of the structure. The result will be a decreasing distribution of D's from left to right. The last three rules are intended to control the system's overall size.

When rule sets become even slightly complicated, as in the last example, it evident that development will be difficult to predict. It can be hoped, however, that with the help of the genetic algorithm, the ability to design and analyse organisms in advance will not be necessary in order to build successful and interesting ones—just as in natural evolution it is not. What does seem essential is an adequate space of possible growth rules. The rule forms discussed include self-excitation, self-inhibition, and cross-excitation and -inhibition between different cell types. The repertoire seems fairly complete for a start, but modifications in it and in many other aspects will surely occur as the proposal is studied experimentally and analytically.

4. Conclusion

An extremely schematic representational framework for biological development has been described which may permit simulations of evolution using the genetic algorithm. Major questions that need to be addressed include the accuracy and adequacy of the representation and the problem of computing the phenotype. It is hoped that coupling "developmental automata" with genetic adaptive techniques will yield insights into biological, social, and other systems which are capable of growth.

Acknowledgement

The author thanks D.E. Goldberg for valuable comments on an earlier draft of the paper.

References

[1] Holland, J. H. *Adaptation in natural and artificial systems.* Ann Arbor: University of Michigan Press, 1975.

[2] Holland, J. H. "Escaping brittleness: the possibilities of general-purpose learning algorithms applied to parallel rule-based systems." In R. S. Michalski, J. G. Carbonell & T. M. Mitchell (Eds.), *Machine learning, an artificial intelligence approach. Volume II.* Los Altos, California: Morgan Kaufmann, 1986.

[3] Smith, S. *A learning system based on genetic algorithms.* Ph.D. Dissertation (Computer Science). University of Pittsburgh, 1980.

[4] Grefenstette, J. J. (ed.) *Proceedings of an international conference on genetic algorithms and their applications.* Pittsburgh: Carnegie-Mellon University, 1985.

[5] An important paper used mechanisms closely related to those of the genetic algorithm as defined in [1] to study the emergence of self-replication. See Holland, J. H. "Studies of the spontaneous emergence of self-replicating systems using cellular automata and formal grammars." In A. Lindenmayer & G. Rozenberg (eds.), *Automata, languages, development.* Amsterdam: North-Holland, 1976.

[6] The representation appears to share several aspects with the developmental models called L-systems. However, there do not seem to be any studies in which populations of L-systems undergo evolution. See Lindenmayer, A. "Developmental algorithms for multicellular organisms: a survey of L-systems." *J. Theor. Biol.*, **54**, 1975.

[7] For background, see Balinsky, B. I. "Development, Animal" and Waddington, C. H. "Development, Biological." *Encyclopaedia Britanica*, 15th Ed., Vol. **5**, 625-650.

[8] Schaffer, J. D. "Learning multiclass pattern discrimination." In [4].

[9] Wolfram, S. "Cellular automata as models of complexity." *Nature*, **311**, 4 October 1984, 419-424.