Bias and Scalability in Evolutionary Development

Timothy G.W. Gordon and Peter J. Bentley
Department of Computer Science
University College London
Malet Street, London, WCIE 6BT, U.K.
+44(20)76797214
{t.gordon, p.bentley}@cs.ucl.ac.uk

ABSTRACT

The introduction of a genotype-phenotype map modelled on biological development can potentially improve the scalability of evolutionary algorithms. Previous work by Gordon and Bentley demonstrated that such a model can be used to evolve patterns that map to useful but small phenotypes. This paper uses the same model to generate much larger patterns covering arrays of up to 64x64 cells. The results show that the model's performance is generally comparable to similar development-based systems [12, 14], and with some measures outperforms them. Additionally the inherent biases of the model are explored, such as the need to use symmetry-breaking initial conditions which some other models do not require. This exploration yields a set of guidelines that suggest what kinds of problem the model is suited to exploring.

Categories and Subject Descriptors

I.2.2 [**Artificial Intelligence**]: Automatic Programming – *program synthesis.*

General Terms

Algorithms, Performance, Experimentation.

Keywords

Development, Pattern formation, Scalability

1. INTRODUCTION

It is widely recognized that the inability of evolutionary algorithms to scale to large and complex problems is a major impediment to their application in the real world [8, 9, 17].

In the past few years a number of researchers have begun to explore how modelling the genotype-phenotype map on biological development might improve scalability [8, 12, 14]. In [6] Gordon and Bentley introduced such a model and in [5] it was demonstrated that it could be used to evolve patterns that mapped to useful phenotypes, in this case circuit designs. However the patterns and consequently the circuits they mapped to were small. This paper shows that the same model can be used to evolve high fitness solutions for much larger pattern generation problems, which is important for the generation of large phenotypes.

Recently other researchers have presented evolutionary systems that incorporate various models of development and have applied them to similar pattern generation problems. This paper compares Gordon and Bentley's system with these other systems. The results show that the model used by Gordon and Bentley provides comparable performance to the best of these.

Although genotype-phenotype mappings based on development have the potential to benefit scalability they apply strong biases that affect the nature of the evolved phenotypes. This paper presents an analysis of the biases inherent to Gordon and Bentley's model and derives a set of guidelines that can be used to determine what kinds of problem the model is suited to.

The rest of this paper is structured as follows: section 2 presents a summary of Gordon and Bentley's developmental model. Section 3 explores the generation of large patterns and derives guidelines for determining if this model might be suited to a particular problem. Section 4 presents the evolution of more complex problems, and compares the performance of Gordon and Bentley's model to others found in the literature, and provides possible explanations for the observed greater performance of the Gordon and Bentley model. Conclusions are made in Section 5.

2. DEVELOPMENTAL MODEL

The developmental model used here is identical to the final model presented by Gordon in [5], where full details of the model and the design decisions that lie behind it can be found. It consists of a set of rules that describe how development should proceed, and a two dimensional non-toroidal array of cells that are manipulated by the rule set. Each cell maintains a chemical environment that defines the cell's context. Development occurs over a series of discrete timesteps. At each timestep the chemical environment of each cell is updated by testing the set of rules that make up the chromosome against the current chemical environment in each cell. For each cell, only the rules that match that cell's environment are activated. If the environment differs between cells, it is possible for different rules to activate in each cell, which leads to their environments being altered in different ways. In this way, different chemical environments can be maintained between cells.

The developmental rule set models the process at the heart of biological development's generative ability: DNA transcription. Transcription regulates the rate of gene expression through the

GECCO'05, June 25-29, 2005, Washington, DC, USA.
Copyright 2005 ACM 1-59593-010-8/05/0006…$5.00.

presence of proteins that either increase (activators) or decrease (inhibitors) the transcription rate of a particular gene. These proteins are generated by the expression of other genes. Thus a dynamic, autocatalytic network of gene products specifies which genes are expressed. As transcription is so central to biological development it is modeled by many computational development systems. Models such as the one presented here are termed *implicit* as development's generative ability including the ability to create modules and iterative structures emerges from the interaction of the rules. Examples of such systems are [1, 4, 10, 12, 14] and are unlike *explicit* systems, which explicitly define the generative process much like a computer program, thus have the disadvantage that mechanisms for iteration and modularity must also be explicitly defined. Examples of explicit systems can be found in [7] and [11].

2.1 Rule Structure

An example of a developmental rule is shown in Figure 2.1. Proteins are modelled as binary state variables. The precondition of each rule specifies which proteins must be present (activators), and which must be absent (inhibitors) in order for that particular gene to activate. Each rule consists of a conjunction of conditions that must be true for the rule to activate. There are two terms in each rule for each of the four proteins in the model. The first is a two bit condition that specifies what proteins the cell itself must generate (11) or not generate (00) for the rule to activate, with the other two bit combinations representing don't care terms. The second term is a five bit condition that defines the context supplied by the neighbouring cells: the first two bits define either an equality, inequality or one of two precedence operators, and the final three bits define the protein concentration that the operator will act upon. This concentration is measured by summing the amount of the protein generated by the cell's four Von Neumann neighbours. As it is coded by five bits the rule can specify concentration values between 0 to 7, even though the maximum concentration that could be detected is 4, when all neighbouring cells are generating a protein. This allows "don't care" terms to be produced using impossible events. The postcondition of the rule consists of two bits that define which protein is generated if the rule is activated. This model is closely related to Wolfram's outer totalistic cellular automata [15].

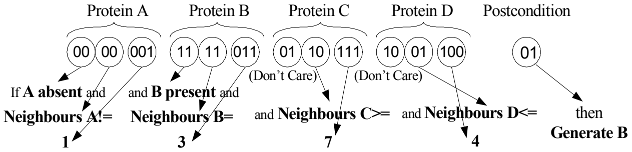

Figure 2.1. The Outer Totalistic Developmental Rule

For the model to even begin to generate useful patterns it is necessary for rules to activate in some cells and not others. For this to happen, it is necessary for some cells to experience different contexts than others at the start of development. One way this can be achieved is by introducing a set of simple yet inhomogeneous starting conditions from which further inhomogeneity in contexts can grow. Another way arises from the non-toroidal nature of the cellular array. Cells that lie at the edges and the corners of the array communicate with fewer neighbours than other cells, thus a rule that activates when a protein is detected at low concentrations in neighbouring cells may activate in the corner and/or edge cells, but not in the other cells.

3. EVOLUTION OF LARGE PATTERNS

In [5] Gordon presented the evolution of various patterns across a 3x5 array of cells. This section presents similar experiments using a 20x20 array of cells in order to demonstrate that the evolution of patterns can scale to larger arrays.

3.1 Initial Considerations

Before these experiments are presented it is worth considering some features of the model that have a profound effect on the kinds of patterns that the model can generate.

3.1.1 Information Transmission

It is now widely accepted that a common strategy used in biological development to specify different embryonic regions is the generation of some form of *positional information* which can then be interpreted by each cell to determine its fate [16]. In this model information transmission occurs explicitly by interactions between cells, under the control of evolved rules. As the model of development used here only allows nearest-neighbour communication (in order to keep computational complexity low) information must be transmitted through local interactions with neighbours alone.

This is likely to have important consequences for scalability, for the following reason. In the model of development used here, positional information must be represented in terms of a protein context. The number of unique contexts is limited by the number of proteins present in the system. The number of unique contexts is also constrained by interactions with neighbouring contexts: if a cell is to be labelled with a unique and stable protein context it must interact with the contexts of its neighbours in a benign manner. Each unique context must be governed by its relationship with neighbouring contexts, and defined perhaps by a single rule, or at least by part of a rule (or several rules). In turn these neighbouring contexts must also be defined by positional information that is defined by a similar rule, or set of rules. The transmission of positional information over greater distances requires a greater number of rules to specify the chain of relationships between neighbouring cells, from an initial point of inhomogeneity to the point where positional information is required. As evolution must discover these chains of rules, it is likely that any pattern requiring positional information to be transmitted over large distances will be difficult to evolve. Although this might at first glance suggest that the generation of any large pattern is difficult, this is in fact not the case. In the following sections the kinds of patterns that do and do not require such long-distance transmission will be discussed.

3.1.2 Symmetry

Because a cell cannot distinguish between information transmitted from individual neighbouring cells the patterns generated by the developmental model are biased towards particular symmetries. Consider a hypothetical non-bounded array where the initial conditions are set so that a single cell contains one protein. As the developmental rules do not distinguish between the four symmetrically-arranged neighbouring cells, any pattern that develops from this source cell must exhibit the following symmetry elements that intersect at the source cell: a rotational four fold point of symmetry and four mutually orthogonal lines of symmetry, one along the horizontal axis, one along the vertical axis and two bisecting these axes. An assemblage that exhibits such symmetry is said to have four-fold dihedral symmetry, or belong to the D_4 point group [2]. Should the problem exhibit D_4 symmetry this feature can be of great advantage, as it effectively reduces the problem to a quarter of its size, with the remaining three quarters of the solution arising automatically from the nature of the model. However many problems may require more complex symmetry. For these problems the symmetry must be broken. There are two mechanisms by which development can do so. The first is if development is carried out on a bounded array (as is the case here). As was stated earlier, in this case the number of neighbours experienced by each cell is inhomogeneous across the array. Because of this it is possible to construct rules that behave differently in these three types of cell, even with identical initial conditions, thus break the symmetry of the developing pattern. In this case the patterns that can arise directly from these inherent inhomegeneities are of course limited to those that form a D_4 point at the centre of the array, because each edge and each corner is identical, and they are arranged with D_4 symmetry around the central point. The second mechanism by which symmetry can be broken is through interaction with a second pattern that has developed from another cell (or group of cells) elsewhere in the array. Of course such a pattern can only arise if an inhomogeneity is present in the array at the source of this pattern. Again this could only be brought about by either prudent selection of initial starting conditions at that point or the development of a pattern from the inhomogeneity inherent in the edges and corners of a bounded array. Because of this constraint, with this model of development the initial conditions for each experiment must be carefully chosen to ensure that the symmetry can be broken in a way that patterns with the symmetry of the target pattern can be generated.

Development enhances scalability by exploiting regularity during pattern formation. Regular patterns can be placed in one of two categories: those with translational symmetry, and those without. A pattern has translational symmetry if its motif can be consistently translated by some vector without altering the pattern. Examples of this class of pattern are cheques and stripes. These patterns represent a fairly trivial level of reuse that could easily be exploited by a traditional designer. However they provide a simple means of exploring whether the model of development used here allows evolution to discover and reuse simple regularities. Examples of regular patterns with non-translational symmetry include French and Norwegian flags.

Sections 3.2 and 3.3 explore the development of patterns with translational symmetry. Section 3.2 focuses on patterns with small motifs and section 3.3 on larger ones. Section 4 compares the performance of the model presented here with models used by other researchers using regular patterns that can be described using a generative process but have no translational symmetry.

3.2 Translational Patterns with Small Motifs

The first set of experiments presented here demonstrates the evolution of striped and chequed patterns similar to those evolved in [5] scaled to a 20x20 array. For all the following experiments a set of 20 rules was evolved using a traditional genetic algorithm. The parameters of the algorithm are shown in Table 3.1 Ten runs of each experiment were run, each until an optimal solution was discovered or for 1000 generations. Fitness for the initial set of experiments was again based on the Hamming distance of the candidate pattern of proteins from the target pattern of proteins. The target pattern for the following set of experiments involved only protein A. The longest distance possible between a candidate solution and the target pattern was 400, which would result if for every of the 20x20 array where protein A was specified as present in the target pattern it was absent in the candidate solution, and vice versa. Fitness was set as the measured distance between the target pattern and the candidate solution for protein A subtracted from the longest distance of 400.

Table 3.1. Parameters for the evolutionary experiments

Operator	Type	Rate
Selection	2 Member tournament	80%
Crossover	One-point	100%
Mutation	Point	5 per chrom.
Other parameters: Generational GA with elitism, 1000 generations, population size = 100, random initialisation.		

3.2.1 *Experiment 1A: A Chequed Pattern*

The target of the first experiment was to evolve a chequed pattern of proteins. This is presented in Figure 3.1 as the target pattern for experiment 1A. This target pattern is highly symmetrical: an infinite array of the pattern would contain a D_4 point at every cell. Hence the initial conditions were set simply as if a single cell at the south west corner had generated protein A at timestep *t-1*. This required that both the map for the self-generated proteins and the map for the neighbour-generated proteins were updated, as shown in Figure 3.2, where lowercase conditions refer to the neighbour-generated protein map and the uppercase condition refers to the self-generated map. The results of the experiment are shown in Table 3.2. Every run discovered an optimal solution. The mean generations to discover an optimal solution was 174.1.

All the optimal solutions developed the target pattern by generating a wavefront of information that moved diagonally across the array, one cell at each timestep, from the southwest corner where the initial conditions were set to the northeast corner, although the details of which proteins were involved in the process and how they were used to generate the final pattern varied. This is the same strategy that was reported for the evolution of similar patterns in [5]. It is clear from the results of Experiment 1A that using this model of development, evolution is capable of regularly discovering developmental rule sets that can generate large arrays of simple chequed patterns from very simple starting conditions.

Table 3.2. Results evolving patterns across large arrays.

Expt.	Best / Max. Fitness	% Optimal Runs	Mean (Best Fitnesses)	Std. Dev. (Best Fitnesses)
1A	400/400	100	400.0	0.0
1B	400/400	40	362.5	43.92

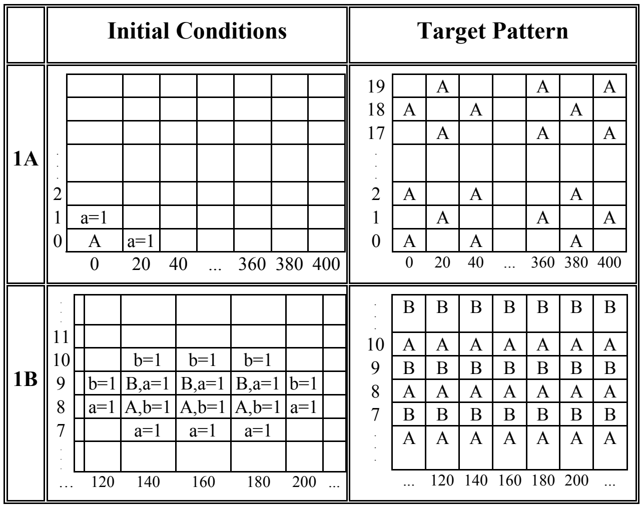

Figure 3.1. Initial protein concentrations and target protein patterns for chequed and striped patterns on a 20x20 array

3.2.2 Experiment 1B: A Striped Pattern

The target of the second experiment was to evolve a striped pattern of proteins, which is shown in Figure 3.2 as the target pattern for experiment 1B. Striped patterns are less symmetrical than chequed patterns: an infinite array of striped cells does not exhibit D_4 symmetry around every cell. Each cell only contains two lines of reflection, along the horizontal and vertical axes and a 2-fold point of rotation, or D_2 symmetry. Thus for this pattern to be attainable by the developmental model, the initial conditions must break the additional two lines of reflection and reduce the rotational symmetry from four-fold to two-fold. A set of initial conditions that achieve this are shown in Figure 3.2, as the starting conditions for Experiment 1B. These conditions are as if three horizontally adjacent cells towards the centre of the array had generated protein A and three cells vertically adjacent to this stripe of A had generated B at timestep t-1.

This pattern was considered harder for development to create than that of experiment 1A for the following reason. Without the introduction of symmetry-breaking starting conditions, the developmental model used here is constrained to generating patterns with D_4 symmetry. This means that the maximum number of unique contexts it is possible for a set of rules to differentiate between is limited to around one eighth of the array, as shown in Figure 3.2a. The solution space searched by evolution is limited to patterns constructed from these contexts. Once symmetry is broken, the number of unique contexts exposed to evolution is increased (assuming a unique context can be assigned using the number of rules and proteins present in the system). However once this is done, evolution must discover rules to manipulate interactions between cells that was previously achieved as a consequence of the symmetry. For instance when using starting conditions that reduce D_4 symmetry to D_2 symmetry, as in experiment 1B, and as depicted in Figure 3.2b, evolution must learn rules to explicitly control both horizontal and vertical growth, rather than growth in one direction only. (A manually designed solution for experiment 1B required ten rules compared with two for experiment 1A.) Not only must evolution now learn to explicitly control more interactions, it is also possible for the interactions governing growth in both directions to interact, leading to an increase in epistasis. Consequently the problem is likely to be more difficult for evolution to discover solutions.

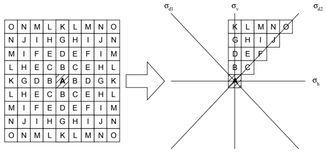

Figure 3.2a. The maximum number of unique contexts available with simple starting conditions arises from its inherent D_4 symmetry.

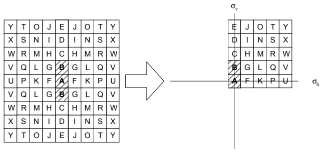

Figure 3.2b. Introducing symmetry-breaking starting conditions increases the maximum number of contexts.

The results of experiment 1B support this reasoning, with only 40% of the 10 runs discovering an optimal solution. Nevertheless, evolution still managed to discover solutions across this large array. Furthermore the most efficient solution discovered used only four rules, many less than the manually-designed alternative. All the optimal solutions discovered by evolution alternated between two striped patterns at each timestep, and on the fiftieth timestep matched the target pattern. They all used a similar basic strategy, depicted in Figure 3.3. At the first timestep a ring of another protein was created around the cells set with protein A as starting conditions (in the example of Figure 3.3 this is labelled B), and this was used as a mask to prevent the generation of a further protein that was otherwise homogenous across the array (labelled C below). At a further timestep those cells with fewer than three neighbouring cells generating C were set to A. A consequence of the geometry of the cells is that cells meeting this criterion will be present both directly and diagonally above the current row of A, thus a new, larger row is created both above and below the original row.

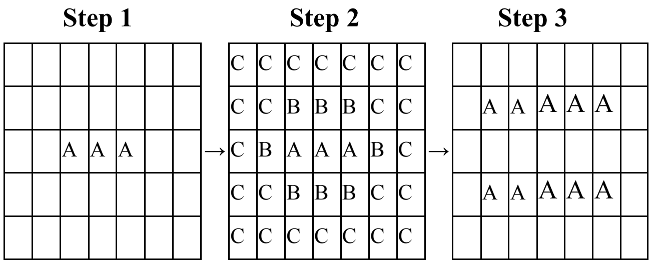

Figure 3.3. The growth strategy used by all optimal solutions to Experiment 1B.

The experimental results suggest that the evolution of patterns with small regular motifs scales to large arrays reasonably well. This suggests that evolution is exploiting a strategy that does not rely on global positional information. In both the experiments presented here and the experiments of [5], evolution used wavefronts of activity that move across the cellular array, generating a stable pattern behind them. Rather than transmitting unique positional information over large distances, information is transmitted that defines position *relative* to a nearby newly-generated motif. As the motif is small, this requires only a handful of unique contexts to be defined and a handful of rules to govern them. Hence it seems reasonable to suggest that using this model of development, patterns with small motifs are generally scalable to large arrays, assuming that the symmetry requirements for the formation of the pattern are met by the initial conditions selected.

3.3 Translational Patterns with Large Motifs

Real-world design problems are likely to be composed of larger, more complex structures than those explored above, with regularity evident on a greater scale. If evolution is to exploit such regularity it is important for the developmental model to be able to generate regular patterns with larger motifs.

Evolution used relative positional information to generate the patterns above, thereby avoiding the difficulty associated with transmitting information large distances to unique sites using a nearest-neighbour model. This suggests that the generation of patterns based on larger motifs is likely to be more difficult, as relative positional information must be transmitted larger distances, hence more unique contexts that interact benignly with their neighbours must be defined by the rules.

3.3.1 Experiment 2A: Evolving Larger Chequed Patterns

A series of experiments were carried out to test whether evolution's performance did decrease as the distance positional information was transmitted over increased. The pattern used in Experiment 2A was the chequed pattern of Experiment 1A, but with the motif scaled to increasingly greater sizes, equally in both dimensions from a 1x1 motif to a 7x7 motif. The initial conditions were also changed. For experiments with each size of motif the initial conditions were set so that the array contained a single copy of that motif in the southwest corner. Figure 3.4 shows a plot of the mean of the best fitnesses for 20 runs of each motif.

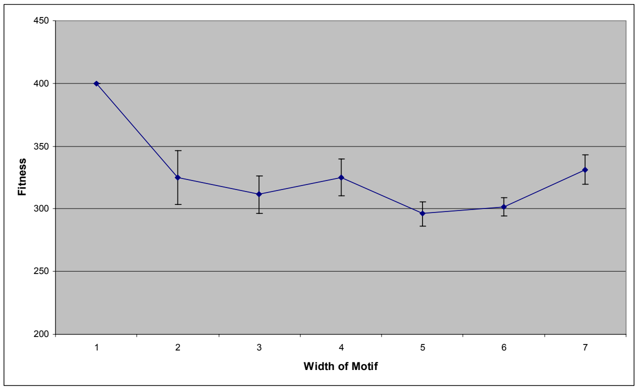

Figure 3.4. Plot of mean fitness against motif width for 20 runs of the evolution of a pattern with a chequed motif. Error bars represent one standard deviation.

Inspection of Figure 3.4 suggests that there is initially a trend towards poorer fitness for larger motif size, but for the larger 6x6 and 7x7 cheques, fitness appears to improve. Additionally the 4x4 cheque has fitness similar to and perhaps greater than the 3x3 and 2x2 cheques (although there is some possibility that these features are due to noise). Although this might at first seem surprising, it is easily explained. It was discussed earlier that the corner cells can be used to generate inhomogeneity, but any pattern generated purely from these points will exhibit D_4 symmetry around the centre of the array. The target pattern for the 4x4 cheque exhibits such symmetry, and the target patterns for the 6x6 and 7x7 cheque are close to, but not quite symmetric about this point. However they are close enough that many patterns that are symmetric about the centre of the array produce a high fitness score for these problems in particular, when using a 20x20 array. Examination of the evolved solutions for these problems revealed that evolution almost always made use of at least some information derived from the corner cells, although in many cases the pattern was perturbed by additional pattern formation around the motif provided as an initial condition. When development of this solution was rerun with the corner motif that was provided as an initial condition removed, the pattern formed was extremely similar to that formed when the initial conditions provided during evolution were present, and the solution only dropped in fitness from 368 to 362. This suggests that the majority of the information used by evolution to generate the pattern originated at the corner cells.

Patterns generated in this way can be considered as forming a single 10x10 motif that stretches from a corner to the centre of the array, which is replicated by the inherent symmetry of the developmental model. No transmission of relative positional information to allow the formation of multiple motifs is necessary. Instead evolution needs only to learn to generate a single motif. The results suggest that evolution finds this alternative strategy, and indeed strategies combining the use of both corner inhomogeneity and explicitly provided initial conditions provide easier routes to moderately fit solutions than using the initial conditions alone. However these strategies would not generate high fitness solutions for this problem on larger arrays as the symmetry of the target pattern with respect to the array edges would change. Similar strategies are also unlikely to be as successful on larger array for other problems that happen to exhibit D_4 symmetry around the centre of the array, as the complexity of the "motif" occupying a quarter of the array would increase with the grid size. This would likely require an increase in the number of rules needed to specify the pattern correctly.

However setting aside non-general strategies that result from unusual matches between the symmetry of the target pattern and the symmetry of the particular array used, the general trend appears to be again towards lower performance as motif size is scaled.

3.3.2 Experiment 2B: Evolving Larger Striped Patterns

Experiment 2B repeated Experiment 2A, but this time using the striped pattern of experiment 1B as the target. A plot of the mean of the best fitnesses of each run is shown in figure 3.5. These reveal a similar initial trend to experiment 2A, with fitness dropping as motif size scales. However for the largest motifs, there is again an increase in performance. Solutions involving transmission of information from the corner cells are not viable for this pattern, as it exhibits D_2 rather than D_4 symmetry, and examination of the solutions confirmed this, showing that evolution never used this strategy in the best solutions found. However solutions for the large motifs again used non-general solutions that relied on the distance between the outer stripes and the array edge.

The development of the smaller problems, where more general strategies are often evolved that transmit relative positional information across a wavefront of activity do again appear to be harder to discover as motif size scales, lending further support to the conclusion that in general, evolution finds patterns with larger motif sizes more difficult to discover.

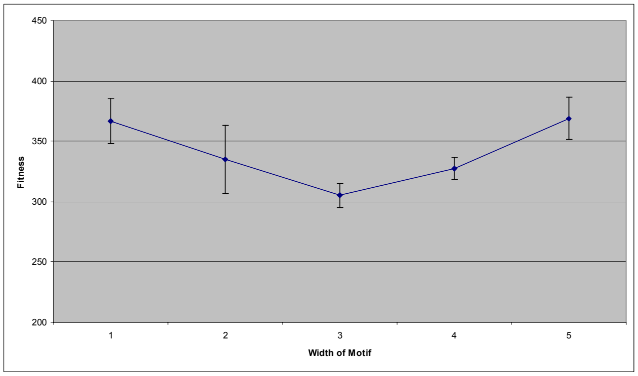

Figure 3.5. Plot of mean fitness against motif width for 20 runs of the evolution of a pattern with a striped motif. Error bars represent one standard deviation.

4. NON-UNIFORM PATTERNS

The repeating patterns discussed above can all be formed by translating a motif uniformly across the array by some vector. However there are many other patterns that exhibit regularity, but not uniformly across the entire array. This section first discusses which types of non-uniform pattern can be exploited by development, and then demonstrates the evolution of non-uniform, yet scalable patterns on the 20x20 array. The patterns that have been used in this section have also been evolved by other researchers interested in scalability, allowing a performance comparison between the model of development used here and that used by other researchers.

Regular non-uniform patterns can be separated into two classes. The first are patterns that possess point symmetry elements and also exhibit translational symmetry within the primitive asymmetric area of the pattern. For patterns of this class every motif in the primitive area can be mapped onto another using a simple translational operation, and the point symmetry of the pattern can then be used to reproduce the pattern in its entirety. Examples in this class are bow-tie shaped patterns and crosses. Just as was discussed in the section above, development could provide a scalable solution to problems of this class by learning a set of rules that transmit information through the translation vector and another set that generate a new motif. The examples given above exhibit symmetry around a centre point hence present opportunity for evolution to take advantage of the inherent symmetry in the developmental process. Less symmetrical structures of this class exist, for example triangles, which could be generated through symmetry-breaking starting conditions.

The second class of patterns are those that do not exhibit strict translational symmetry, but can be decomposed into elements that do. Examples of such patterns are the simplified Norwegian and French flags shown in Figure 4.1. One way development could solve the problem would be to learn a set of rules for each element of the pattern, with each set encoding both the translation vector and a motif-generating process for that particular element. For instance the Norwegian flag pattern in Figure 4.1a can be decomposed into the two intersecting blue crosses with surrounding white crosses, and all other cells red, each of which could be generated with a separate set of rules.

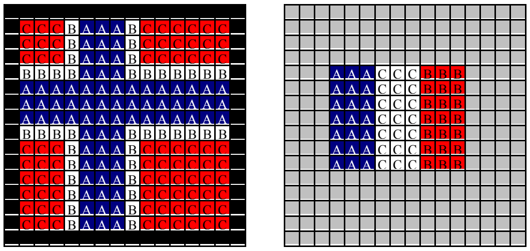

Figure 4.1(a) and (b). Patterns based on the Norwegian and French flags

4.1.1 *Guidelines for Suitable Problems*

From the analysis of the experiments above a number of guidelines can be summarized that suggest what kinds of problems this model of development is suited to:

- Positional information can only be transmitted successful over short distances, thus problems where solutions are likely to be large and highly irregular are unsuitable for this model.
- If a large problem is to be tackled successfully it must exhibit regularity that allows *relative* positional information to be transmitted rather than absolute.
- The system is likely to perform best when its inherent bias towards symmetrical patterns can be taken advantage of, and starting conditions should be chosen to promote this. If good solutions to a problem are likely to be of lower symmetry, the inherent symmetry can be broken by prudent selection of starting conditions.
- Care should be taken to avoid problems where many solutions are symmetrical around the starting conditions and non-optimal yet fit: such solutions are likely to be deceptive.

4.2 Comparison with Roggen and Federici's Developmental Systems

In [14] Roggen and Federici evolved solutions to the Norwegian flag problem. They compared two developmental systems, one that models diffusion but does not model any interaction between the proteins that diffuse across the array (called the Morphological model), and the other based on Miller's Cartesian Genetic Programming model [13] that models both diffusion and protein interaction (called the Embryogenic model). The major difference between Miller's and Roggen and Federici's implementation was that Miller's logical mapping between cell inputs and outputs was replaced with ANNs. Roggen and Federici carried out their comparison across a range of array sizes. An experiment was conducted to compare the performance of the system presented here with those used by Roggen and Federici for array sizes of 16x16, 32x32 and 64x64. The 16x16 pattern is shown in Figure 4.1a. The 32x32 pattern centres the cross on the 15x18th cell with the blue stripes 5 cells wide, the white stripes 2 cells wide and a surrounding border 1 cell wide. The 64x64 pattern is centred on the 29x35th cell with the blue stripes 11 cells wide, the white stripes 5 cells wide and a surrounding border two cells wide. All evolutionary and developmental parameters were as described for the experiments 2A and 2B, except the fitness measure, the starting conditions and the number of developmental timesteps. The starting conditions for each pattern size were set as if the cell at the centre of the cross had generated protein A at timestep *t-1*. Four proteins were required to form the pattern the hence the fitness of each candidate solution ranged from a maximum of 4 x *n* to 0, where n is the total number of cells in the array. Ten runs of each pattern size were conducted, and the results are shown in Table 4.2, along with Roggen and Federici's results for comparison. (Note that the values tabulated for Roggen and Federici's results are approximate, as they were obtained visually from the plot presented in [14]. The patterns of proteins A and C for the best solution evolved are shown in Figure 4.3.

The results in Table 4.2 suggest that the system presented here outperforms both the other systems across all three array sizes. Furthermore this was achieved in half the number of generations used by Roggen and Federici (1000 rather than 2000).

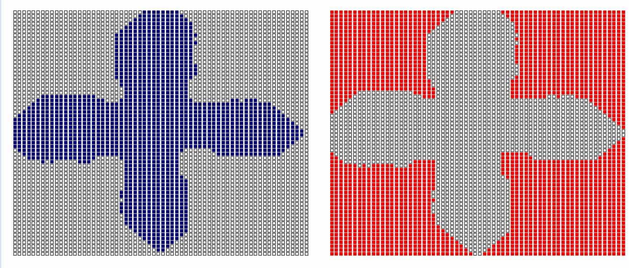

Figure 4.3 Best evolved 64x64 Norwegian flag. In this solution proteins B and D were not present at the end of development.

It is worth considering what features of Gordon and Bentley's model may be leading to the improvement in performance. The first point to note is that the Morphological model does not model any interaction between the proteins: protein generation is fixed at the start of development. This seems to constrain the patterns that can be generated to patterns with little fine detail: the patterns tend to consist of large areas where cells assume the same identity and boundaries between the two areas. It is perhaps unsurprising that this model generates relatively simple patterns as the protein interaction of the other systems can be thought of as an additional layer of computation available to evolution.

The second point is that both models used by Roggen and Federici differ from Gordon and Bentley's model in that each cell can only assume a single protein state. Gordon and Bentley's model allows each cell to generate any combination of proteins. Cells in Gordon and Bentley's model that contain the target protein along with other proteins contribute towards fitness even though the state of the cell is not perfect. This may provide a smoother fitness landscape for this kind of problem.

A final point is that unlike the other two models, the cell state of the Embryological model used by Roggen and Federici is calculated by evolving a function that accepts inputs from neighbouring cells independently. Gordon and Bentley's model takes a totalistic approach, summing the states of neighbouring cells. This results in a strong bias towards generating symmetrical patterns (such as the solution to the Norwegian flag problem shown in Figure 4.3) which the Embryological model does not have. As stated earlier this can be of great benefit as it effectively reduces the problem size, and can be likened to processes in biological development which take advantage of the physics and chemistry inherent in the environment. In fact it was noted by Dawkins [3] that many biological organisms are highly symmetrical and that such symmetries are likely to arise as a natural consequence of their development and aid evolvability.

Table 4.2. Comparison between the current system and approximate results for both the Morphogenetic and Embryonic systems presented in [14]

	Best			Mean			
Problem Size	**GB**	**RF Morph**	**RF Emb.**	**GB**	**GB St. D.**	**RF Morph**	**RF Emb.**
16x16	91.41	~76	~67	88.45	3.93	~70	~58
32x32	90.04	~73	~70	88.81	3.41	~70	~58
64x64	81.64	~72	~80	74.45	3.84	~68	~68

4.3 Comparison with Miller's System

Miller has also explored the generation of flag patterns including the French flag pattern shown in Figure 4.1b using his Cartesian Genetic Programming-based developmental system [12]. In this study Miller was not interested only in scalability but pattern maintenance and regeneration. Hence the problem he tackled was not purely to generate the pattern at the final developmental timestep but to maintain it for a number of timesteps. This experiment was repeated using Gordon and Bentley's system. The problem requires that a 9x7 French flag pattern be developed and maintained on a 16x16 array over 10 developmental timesteps. Fitness was measured by summing the distance between the target and developed patterns as before, but the fitness is summed over the final four developmental timesteps, to produce a bias towards pattern maintenance. The developmental and genetic parameters for this experiment were identical to the previous experiments, except the number of developmental timesteps was reduced to 10 in line with Miller's experiment, fitness was calculated across all four proteins again in line with Miller, and the starting conditions were set to allow the D_4 symmetry of the system to be broken down to D_2 symmetry. This was achieved by setting a small stripe of 3 cells at the centre of what should develop into the flag pattern to proteins A, B and C. The results of the experiment are shown in Table 4.3. Results from [12] are provided for experiments conducted using 0 and 2 proteins, representing the worst and best results achieved respectively.

The best performance for this problem was found by Miller's 2 protein CGP system. However the mean of the best fitnesses achieved with Gordon and Bentley's system is greater than Miller's CGP system, and the best solution found by Gordon and Bentley's system had greater fitness than Miller's 0 protein system, and was close to Miller's 2 protein system. It should be noted that Miller's results were obtained using 150,000 evaluations rather than 100,000 used here, suggesting that additional computational power is needed to achieve good results using Miller's system. The worst solutions found by Miller's systems had significantly lower fitness than Gordon and Bentley's worst result, and a higher standard deviation. This suggests that the fitness landscapes generated by Miller's systems more deceptive or present more local optima than the landscapes of Gordon and Bentley's system, at least for this problem.

Table 4.3. Comparison between the developmental system presented here and results for Miller's CGP system used in [12] with 0 and 2 proteins.

System	Best as % Max	$\overline{f}_{best}$ as % Max	Worst as % Max	Normalised Std. Dev.
GB	97.22	96.13	95.31	0.52
Miller , 0 Prots	88.77	85.53	83.50	1.60
Miller , 2 Prots	98.83	91.67	87.30	3.03

Again possible explanations as to why Miller's system requires more evaluations to achieve comparable results are bias towards symmetry and fitness calculation. Like Roggen and Federici's Embryological model, Miller's system is not biased towards symmetrical patterns, and so is likely to spend more time searching non-fruitful areas of solution space. And again like Roggen and Federici's systems, the cells of Miller's system can only assume a single distinct cell type, and so the fitness landscape is somewhat different to that searched by Gordon and Bentley's model. Finally the model used by Gordon and Bentley requires the use of symmetry breaking starting conditions, where as Miller's (and indeed Roggen and Federici's embryological model) does not. The starting conditions used in these experiments have been chosen carefully and provide useful knowledge to development, which is not provided to the other systems. Comparisons using various starting conditions will be explored in future work.

5. CONCLUSIONS

The problem of scalability is one of great importance if evolutionary techniques are to be routinely applied to large real-world problems, such as digital circuit design. One possible solution is to use a developmental genotype-phenotype mapping. However research into such systems is at an early stage: they are poorly understood and scalability has not been widely demonstrated through experiment. This paper has shown through a number of experiments that the developmental model used in [5] is capable of generating large phenotypes. In future work it is intended to apply the model to the evolution of large digital circuits. Through these experiments a number of rules have been derived that suggest what kinds of problem are more suited to such an approach. The developmental model used in this paper has also been compared to similar models in the literature. Although it provides more domain knowledge than some of these other models, it appears to provide at least comparable, and in many cases better performance for the problems presented here while using fewer evaluations.

6. ACKNOWLEDGEMENTS

The authors would like to thank Daniel Roggen for supplying the Norwegian flag problem target patterns.

7. REFERENCES

[1] Bongard, J.C. Evolving Modular Genetic Regulatory Networks. *2002 Congress on Evolutionary Computation*. 2002. Honolulu, HI, U.S.A.: IEEE Press. pp. 1872-1877.

[2] Cotton, F.A., *Chemical Applications of Group Theory*. 2nd ed. 1990, New York: John Wiley & Sons Inc. 478.

[3] Dawkins, R. The evolution of evolvability. *Proc. of Artificial Life*. 1989. Santa Fe, U.S.A.: Addison-Wesley. pp. 201-220.

[4] Eggenberger, P. Creation of Neural Networks Based on Developmental and Evolutionary Principles. *7th Int.Conf. on ANNs*. 1997. Lausanne, Switzerland: Springer. pp. 337-342.

[5] Gordon, T.G.W. Exploring Models of Development for Evolutionary Circuit Design. *Congress on Evolutionary Computation*. 2003. Canberra, Australia: IEEE Press, Piscataway, NJ, USA. pp. 2050-2057.

[6] Gordon, T.G.W. and P.J. Bentley. Towards Development in Evolvable Hardware. *2002 NASA/DoD Conf. on Evolvable Hardware*. 2002. Washington DC, USA pp. 241-250.

[7] Gruau, F., *Cellular Encoding of Genetic Neural Network*. 1992, Laboratoire de lInformatique pour le Parallelisme, Ecole Normale Superieure de Lyon: Lyon.

[8] Haddow, P.C., G. Tufte, and P. van Remortel. Shrinking the Genotype: L-systems for EHW? *4th Int. Conf. on Evolvable Systems*. 2001. Tokyo, Japan. pp. 128-129.

[9] Hornby, G.S. and J.B. Pollack. The advantages of generative grammatical encodings for physical design. *Proc. Congress on Evolutionary Computation*. 2001. Seoul, South Korea.

[10] Jakobi, N., Harnessing Morphogenesis, in *On Growth, Form and Computers*, S. Kumar and P.J. Bentley, Editors. 2003, Elsevier: London.

[11] Koza, J.R., M.A. Keane, and M.J. Streeter. The importance of reuse and development in evolvable hardware. *2003 NASA/DoD Conf. on Evolvable Hardware*. 2003. Chicago, IL, USA: IEEE Comput. Soc, Los Alamitos, CA. pp. 33-42.

[12] Miller, J.F. Evolving a self-repairing, self-regulating, French flag organism. *Genetic and Evolutionary Computation Conf.*. 2004. Seattle, Washington, USA.

[13] Miller, J.F. and P. Thomson. Cartesian Genetic Programming. *Proc. of EuroGP'2000*. 2000. Edinburgh, U.K.: Springer Verlag. pp. 121-132.

[14] Roggen, D. and D. Federici. Multi-cellular Development: Is There Scalability and Robustness to Gain? *8th Int. Conf. on Parallel Problem Solving in Nature*. 2004. Birmingham, U.K. pp. 391-400.

[15] Wolfram, S., *A New Kind of Science*. 2002, Champaign, IL, U.S.A.: Wolfram Media.

[16] Wolpert, L., et al., *Principles of Development*. 2nd Edition ed. 2002, Oxford, U.K.: Oxford University Press.

[17] Yao, X. and T. Higuchi, Promises and challenges of evolvable hardware. *IEEE Transactions on Systems, Man and Cybernetics, Part C Applications and Reviews*, 1999. 29(1): pp. 87-97.