

Designing Enzymes in a Multi-Agent System based on a Genetic Algorithm

Geoff POULTON, Ying GUO, Phil VALENCIA, Geoff JAMES, Mikhail PROKOPENKO,
Peter WANG

*Centre for Intelligent Systems Design,
CSIRO ICT Centre,
Marsfield NSW 2122, Australia*

Abstract. In this paper we present a genetic algorithm-based approach towards designing “enzymes” in a multi-agent system, where agents are defined as square smart blocks. In the real world an enzyme is an organic protein which catalyses a specific reaction. Here, we use it in the context of self-assembling blocks to characterize an assembly of blocks which is capable of producing another block assembly whilst itself remaining unchanged. The aim of this work is to evolve enzyme and block parameters which allow the self-assembly of desired basic structures that can be used as primitive building blocks for the assembly of more complicated objects. We detail an evolutionary approach towards evolving the enzyme design and the state machine logic, with a fitness measure based on the check of the change of the enzymes during the growing process. Some preliminary results illustrate a range of objects that can be generated with this approach.

1. Introduction

A number of research groups, including CSIRO, have developed means for the self-assembly of small (on the order of millimetres to centimetres, sometimes referred to as “meso-scale”) components into regular lattice-like arrangements as analogues of natural systems [1-4]. Such components may be regarded as agents that interact due to their edge properties, which have up to now been regarded either as static, or at most modifiable by changes to environmental parameters. For static components the resultant self-assembled object (if one exists) is dependant on the agents’ properties such as size, shape, edge polarities and the strength of edge fields. Environmental changes may be used to cause further structural development or spontaneous reconfiguration [5, 6].

Guo et al. examine a situation where the components are two-dimensional rectangular blocks, showing that even for a fixed physical shape and a static environment the inclusion of an internal state machine in each block leads to a huge variety of shapes which can result from the self-assembly process [7]. This variety allows the possibility of achieving the directed design of block assemblies by specifying block (agent) properties. The unsolved reverse engineering problem is how to specify these properties to achieve a prescribed global outcome. In [7] a Genetic Algorithm (GA) approach is used to partly address this problem by evolving agent properties so that the final structure formed by the agents conforms to the given target

shape or global goal. However, GA is a time-consuming process and each desired shape must be evolved independently. A more practical approach would be to formulate general rules allowing a variety of objects to be produced. The present paper examines an interesting class of structures which may help in achieving this goal.

The physical counterparts to our simulated agents are 2-dimensional blocks floating in fluid, with surfaces coated with self-assembling monolayers which are either hydrophobic, hydrophilic or neutral. By controlling the surface tension and stirring the water, the blocks effortlessly arrange themselves into regular lattice structures [4]. These experiments show that the basic processes of self-assembly operate in a real environment quite similar to that being analysed.

The key new concept for these structures is the artificial “enzyme”. In the real world an enzyme is an organic protein which catalyses a specific reaction. Here, we use it in the context of self-assembling blocks to characterize an assembly of blocks with the capacity to produce another block assembly whilst itself remaining unchanged. The aim is for the product structures to serve as primitive building blocks for the self-assembly of more complex objects. As will be seen, using enzymes leads to the more efficient production of a wider variety of structures than was possible with the approach of [7]. This being said, GA is again the main tool used to generate enzymes.

The remainder of this paper is organized as follows. Section 2 introduces the 2-D self-assembly environment, including the description of the enzyme concept, and the search space. Section 3 outlines our approach and a short introduction to how GA was used. In Section 4, we present preliminary experimental results. Finally, conclusions based on these experiments are discussed in Section 5.

2. The Multi-agent System of 2-Dimensional Building Blocks

We assume a multi-agent environment where huge numbers of identical 2-dimensional “sea” blocks, moving randomly in a “sea” interact according to their internal properties whenever two blocks come together [7]. The surfaces are polarised either as negative (-1), positive (+1) or neutral (0), and the basic interaction is the sticking together of blocks at edges with opposite polarities. No other combination of polarities will allow blocks to stick [7]. While a large number of objects can be constructed from these static agents by varying their surface properties, a richer variety of self-assembled objects is possible by allowing edge polarities to change following an “event”, under the control of an internal state machine. In this study an event is defined as a block either sticking to or unsticking from another block (although other choices are possible). For simplicity we assume all “sea” blocks to have the same fixed physical shape and internal state machine.

The “enzyme” concept is then introduced to develop a more flexible and directed method of producing basic building objects compared with the approach in [7], where the resulting structures are defined only by the rule set common to all “sea” blocks. By introducing an enzyme into the multi-agent environment, the structures which self-assemble depend also on the rule set and physical structure of the enzyme. Different enzymes will generate different final objects from the same “sea” blocks. Enzymes can also give control over where and when structures self-assemble, by appropriate placement and assuming that enzymes can be switched between active and inactive states.

The properties of agents and enzymes are fundamental to the self-assembly process, and are described in detail in the following sections.

2.1 Agent – 2D Block

At the nanoscale, self-assembly of nanostructures can be controlled by using the attraction and repulsion properties of complementary groups attached to the nanostructures. We have included similar properties in our agents with the aim of generating insights into the self-assembly processes of multi-agent systems of this type, with the hope that these insights may then be transferred to real world environments.

With this aim in mind and referring to Figure 1 the properties of our agents may be defined as follows:

- (1) Each block has four edges, each of which can have positive (+1), negative (-1), or neutral (0) polarity. This is illustrated in Figure 1, where the four edges of the block are labelled for easy description. The edge state of an agent will be described as a vector of four trinary numbers: $\mathbf{Q} = (a_1, a_2, a_3, a_4)$, where $a_i \in \{+1, -1, 0\}$. For instance, the edge state of the block in Figure 1 can be described as $\mathbf{Q} = (+1, 0, 0, -1)$.
- (2) The internal state machine can change the polarity of each edge as the result of an event.
- (3) Events detectable by each block are the acts of “sticking” or “unsticking” at one or more of its edges.
- (4) The internal state machine in each block contains rules linking edge polarity changes to “sticking” or “unsticking” events.
- (5) The initial state of all the blocks in one environment is identical. This includes the edge polarities as well as any internal states.

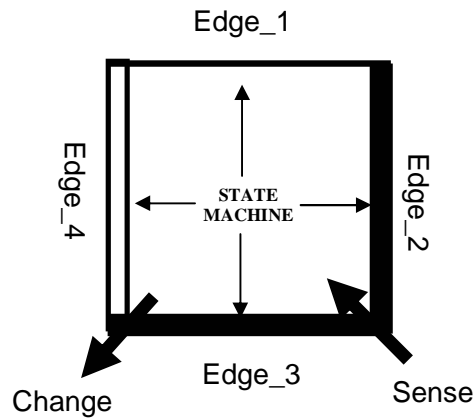


Fig. 1. Structure of each agent in the multi-agent system. Single thin line for positive (+1), double lines for negative (-1), and single thick line for neutral (0).

In the simulation environment attempts the blocks interact in the following manner:

- (1) Edges of opposite polarity stick together and generate a “sticking” event which is passed to the internal state machines of both sticking blocks.
- (2) Like polarities repel and will cause connected blocks to unstick. (This may also generate an event to be passed to the internal state machines, an option which is not used in this paper for reasons of simplicity).
- (3) Neutral polarities will neither attract nor repel any other polarities. If a polarity changes to neutral between two connected blocks then they will unstick.

- (4) Blocks move randomly in the 2-D environment under Brownian-like motion by steps which are multiples of the block width, with rotation of multiples of 90° .
- (5) For a given situation many different structures may self-assemble. However, we will focus only on one randomly chosen structure in the environment. This structure will grow as more blocks stick to it, but may break up due to polarity changes. If a break occurs, we continue to focus on only one randomly chosen part.

A block will separate from a structure if there is no net “sticking force” to hold it on.

For the Q^{th} block this may be determined from the separation function $A(\mathbf{Q}) = \sum_{i=1}^4 p_i$, where p_i

is the sticking force due to the i^{th} edge. p_i is +1 if the edge sticks to a neighbour, -1 if it is repelled and 0 if there is no neighbouring block or a neutral edge. Block \mathbf{Q} will remain part of the structure if $A(\mathbf{Q}) > 0$, or will separate if $A(\mathbf{Q}) \leq 0$. Note that separation functions really should be defined for all structures, not just for single blocks. In general this is a difficult problem which has not been addressed in the present study.

2.2 Enzyme

The properties of an enzyme are defined as:

- (1) Each enzyme is a stable structure comprising K blocks.
- (2) Each block in the same enzyme has the same rule set.
- (3) Blocks in the environment (“sea” blocks), whilst identical, can have a different rule set from the enzyme blocks.
- (4) An enzyme must remain unchanged after the self-assembly process. During the process it may change in size and shape, but it must return finally to its initial state.

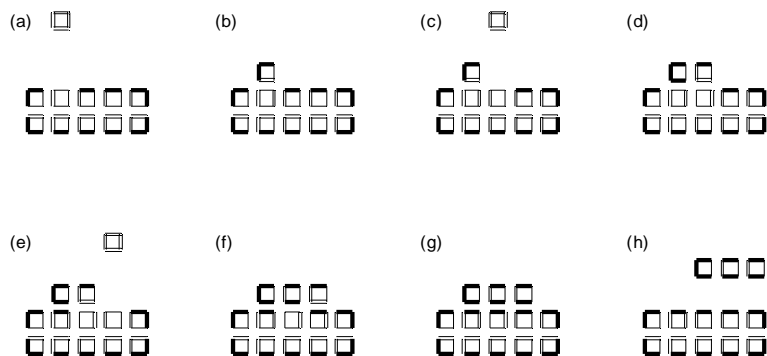


Fig.2. A simple example of the enzyme.

An example of an enzyme enabling a self-assembly process is shown in Figure 2. It is an artificially constructed 10-block structure sitting in a “sea” of blocks with all negative sides and an identical rule set. This enzyme produces linear groups of three blocks with all zero polarity except for one end, which is positive. The enzyme may be simply modified to produce linear structures of any length.

2.3 Internal States and the Search Space

Since we intend to use optimisation techniques to design enzymes and “sea” blocks which produce specific self-assembled structures, it is of interest to look at the size of the search space to be covered. This is shown by the following analysis.

Although in general enzyme blocks can have a complex rule set, for the current analysis the enzyme will be assumed static, so its only parameters are the number of blocks, its shape and edge polarities. Sea blocks, on the other hand, will be allowed a full rule set and thus have many parameters. Even assuming that only a “sticking” event can cause edge polarity changes there will be a large number of possible combinations for the following reasons: (1) any or all edges of a “sea” block may be involved in a sticking event; (2) each such event can change the polarities of all four edges, and (3) these changes depend also on the initial edge polarities. For example, consider the case where only a single edge sticks. Table 1 shows this situation, where S_i is the index of the sticking edge, $(a_{1i}, a_{2i}, a_{3i}, a_{4i})$ represents the initial and $(b_{1i}, b_{2i}, b_{3i}, b_{4i})$ the final edge states.

The domain of S_i is $\{1, 2, 3, 4\}$ whilst that of the a_{ji} and b_{ji} is $\{-1, 0, +1\}$. Each line of the table represents a sticking event and initial edge state which gives rise to a final edge state. There are $N = 4 \times 2 \times 3^3 = 216$ lines (not $4 \times 3^4 = 324$ since a neutral edge cannot support a sticking event). Each choice of the entire set of b_{ji} represents a possible internal state for the block. Since the b_{ji} are all independently chosen from their domain and there are four per line, the total number of combinations is $3^{4N} = 3^{864}$. In practice this calculation is an overestimate for several reasons, for example if rotated blocks are regarded as identical. This reduces N by a factor of 4 to 54 giving 3^{216} combinations. Inclusion of enzyme block parameters leads to a further increase. With a search space of this size most conventional optimisation methods would be expected to have difficulties.

Table 1. Possible polarity changes after “sticking”

Stick Edge	Initial Edge State				Final Edge State			
S_1	a_{11}	a_{21}	a_{31}	a_{41}	b_{11}	b_{21}	b_{31}	b_{41}
S_2	a_{12}	a_{22}	a_{32}	a_{42}	b_{12}	b_{22}	b_{32}	b_{42}
⋮			⋮				⋮	
S_N	a_{1N}	a_{2N}	a_{3N}	a_{4N}	b_{1N}	b_{2N}	b_{3N}	b_{4N}

3. Genetic Algorithm-Based Enzyme Design

Conventional optimisation methods have problems with very large search spaces as in the enzyme design problem, unless there is an underlying smooth structure which is rarely found in multi-agent systems. Genetic Algorithms (GA) are robust in complex search spaces and are appropriate in such cases [8]. A colony of individuals (parameter sets) can be evolved for a number of generations, improving the performance of the colony. Such techniques are inspired by the processes of natural selection. Techniques of fitness determination, selection, cross-over, reproduction, and mutation are applied to the rules and their chromosomal representation.

At the end of each generation, “parent” individuals are selected based on a fitness computation which must be strongly related to the desired outcome. The better the fitness of a given individual, the more likely it is to be selected. After the two parents are selected, each is represented by a “chromosomal” string and are then combined, using one of several methods, to form two new chromosomes. These chromosomes are subjected to potential mutation, and

are then converted back to their equivalent parameter set representation. The selected parents are then replaced in the colony by the offspring. The mechanism of natural selection is expected to eventually result in a colony with a higher performance. The first generation (initial colony) can either be set randomly, or to predetermined values (cf. [7~9] for a detailed description of such algorithms).

GA is set up to generate enzymes with corresponding rule sets for the (identical) “sea” blocks. In every generation each individual of the colony must be evaluated by calculating a value using an appropriate “fitness function”, which is a non-negative well-behaved measure of relative fitness. For the current problem, obviously individuals that can generate enzymes will score highly.

As has been mentioned, the choice of an appropriate fitness function is critical. To simplify the problem we constrain the enzyme to have a fixed number of blocks (K), and separate optimisations are carried out for a range of K values. For each generation every member of the colony defines a potential enzyme which is allowed to grow in the environment until it splits into two or more objects, one or more of which is then evaluated for fitness using the following criteria:

- (1) whether the number of blocks of the object is K ;
- (2) whether the shape of the object is the same as the original enzyme;
- (3) whether the polarity of the object is the same as the original enzyme; and
- (4) how many of the blocks in the object are necessary for the enzyme. For example, there could only be three blocks strictly necessary in a 10-block enzyme, with the other 7 being the equivalent of “junk DNA”.

We have developed a function based on the above criteria which are applied hierarchically, so that failure to satisfy a higher-level criterion terminates the evaluation. For example, any object with a number of blocks $\neq K$ has zero fitness independently of the other criteria.

Four non-negative weights, w_1 , w_2 , w_3 and w_4 are defined to give a relative measure for the criteria.

If f_{num} = number of blocks in an object,
 f_{shape} = number of blocks common to object and enzyme,
 f_{pol} = number of edge polarities common to object and enzyme
and f_{active} = number of blocks undergoing sticks during assembly

then the fitness function may be written as follows:

$$f(object) = \delta(f_{num}, K) \{ w_1 + w_2 f_{shape} + \delta(f_{shape}, K) [w_3 f_{pol} + \delta(f_{pol}, 4K) w_4 f_{active}] \} \quad (1)$$

$$\text{where } \delta(x, y) = \begin{cases} 1, & x = y \\ 0, & x \neq y \end{cases}$$

Notice that the definition of f_{active} measures blocks which undergo sticks rather than the necessary blocks for the enzyme because f_{active} is easier to measure. Weights used in the trials are $w_1 = 1$, $w_2 = 2$, $w_3 = 5$ and $w_4 = 10$.

It is clear that $\max(f_{shape}) = K$, $\max(f_{pol}) = 4 \times K$, and $\max(f_{active}) = K$. As an example, for an enzyme with three blocks the maximal fitness value is: $f(object) = w_1 \times 1 + w_2 \times 3 + w_3 \times (4 \times 3) + w_4 \times 3 = 1 + 6 + 60 + 30 = 97$.

The fitness function and the four weights were determined on the basis of experience. It is acknowledged that further research is needed to optimise this function.

4. Experiments on Enzyme Design

A genetic algorithm using the above fitness function was applied to the design of enzymes for a range of sizes. We report results for $K = 1, 3$ and 4 blocks. In each case an initial colony of individuals was defined with parameters being chosen randomly except that the initial edge state of the “sea” blocks was set to (-1 -1 -1 -1). Since enzyme block polarities have been assumed static, the parameters describing each individual are the enzyme size, shape and edge states together with the internal rule set of the sea blocks.

Each individual is allowed to grow independently by sequentially adding sea blocks, choosing the sticking site randomly if there is more than one such site. If the structure splits into two or more parts each part is checked to see if it identical to the enzyme, in which case the process terminates and the individual is assigned maximum fitness. If no enzyme results then fitness of each part is calculated and stored and one part is chosen randomly for further growth. The process terminates after 40 blocks have been added, or earlier if no further growth is possible. Except when an enzyme has been generated the fitness assigned to the individual is the average fitness of all parts produced. To account for random choices made the process is repeated 10 times for each individual, averaging the fitness values produced.

Once the fitness of all individuals has been calculated a new generation is defined by stochastic sampling using roulette-wheel selection followed by crossover and mutation [8]. The value of mutation probability is set at 3% with cross-over probability of 15%~25%.

4.1 One-Block Enzyme Design

For this experiment the number of enzyme blocks $K = 1$. There are 100 individuals in each generation, and the GA runs for 100 generations. In the final generation, 73 one-block enzymes were found. We tracked the products of all these enzymes, and some of the products with their rule sets are shown in Figures 3~5, where the structure in the left bottom corner of each figure is the enzyme (circled) and other independent structures in the same figure are the products. In Figure 3, the final products are stable (surrounded by neutral) or semi-stable (surrounded by neutral and negative polarities). The stable products will not grow in any environment, while the semi-stable products will not grow in this particular environment where all the sea blocks are surrounded by negative polarity, but may grow in other environments. Figure 4 illustrates some interesting cases of self-replicating enzymes, where at least one product is identical to the original enzyme. In Figure 5, many unstable structures are generated.

Many different structures can result because multiple sticks are often possible. Rule sets can have any size, but in the simulations they are all very small with only a few active rules emerging from the evolutionary process even though the GA had available to it a huge variety of possible rules. (Active rules are those that change the state of at least one edge). Another interesting finding is that all the products of the one-block enzyme were of limited size (in this study the maximum size found was six blocks). This is reasonable since many more smaller than larger enzymes will be produced in each generation, leading to evolutionary domination by the former. A task for future research is to remove this selection effect.

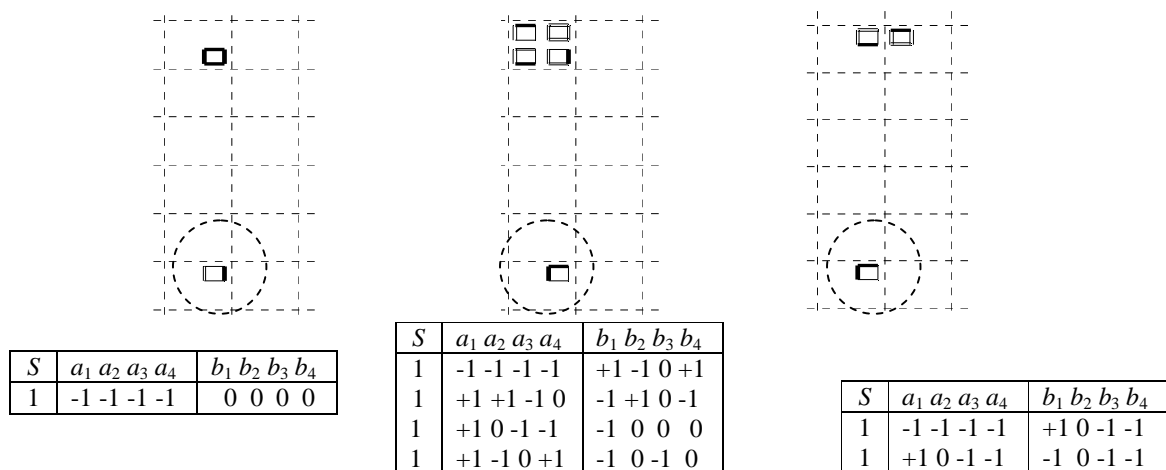


Fig. 3. One-block enzymes which make stable or semi-stable structures with corresponding rule sets.

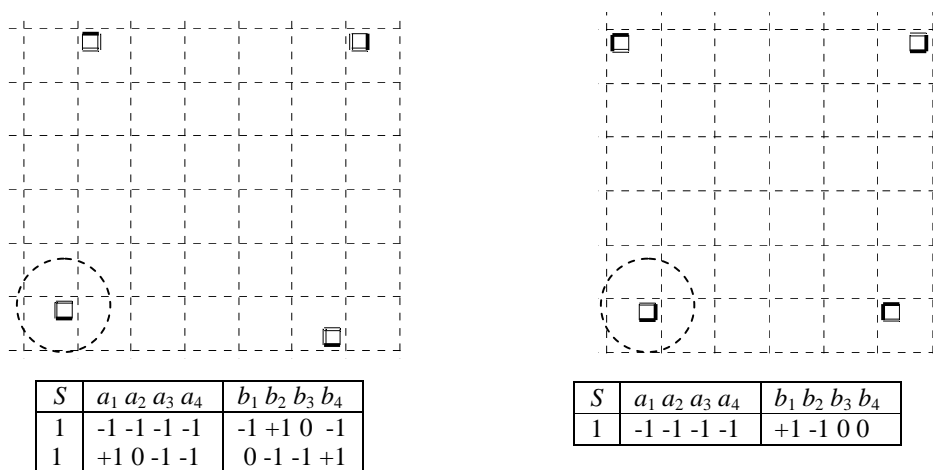


Fig. 4. One-block self-replicating enzymes.

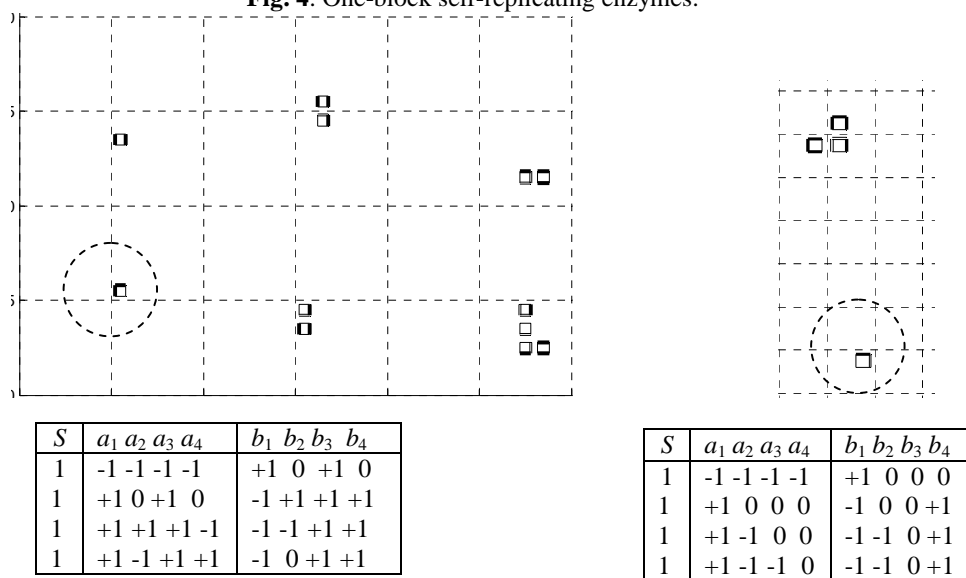


Fig. 5. One-block enzymes which make unstable structures with corresponding rule sets.

4.2 Three-Block and Four-Block Enzyme Design

In order to generate larger structures suitable for future building-blocks, we increased the number of blocks in the enzyme to three and four. Because only a few active rules were needed for the one-block case, we modified the GA process for 3 and 4 blocks by deliberately restricting the number of rules available to the algorithm. This greatly speeded up the process without any noticeable loss in quality of the results. The fitness function is defined as in equation (1).

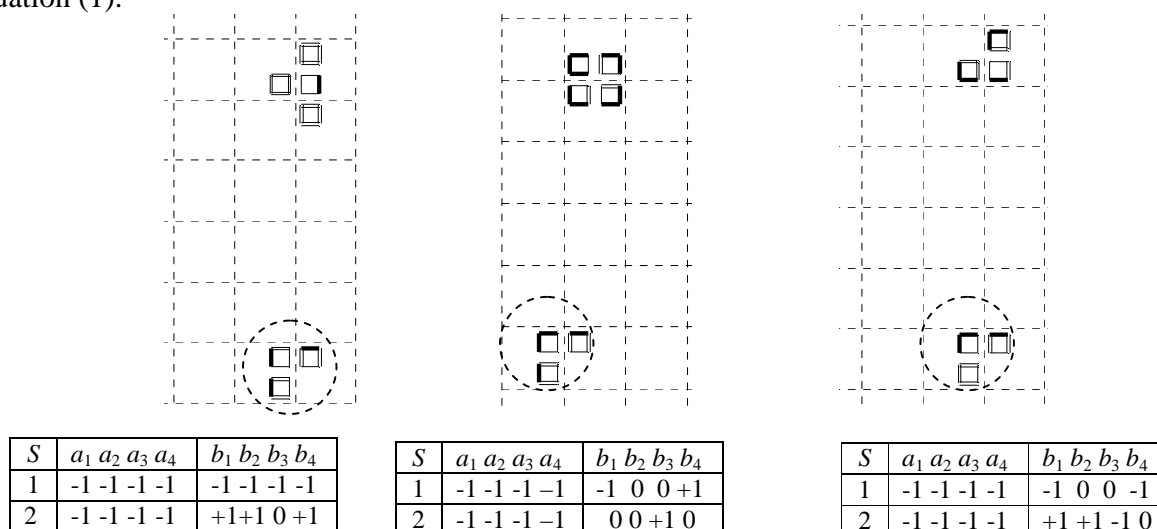


Fig. 6. Three-block enzymes make stable structures with corresponding rule sets.

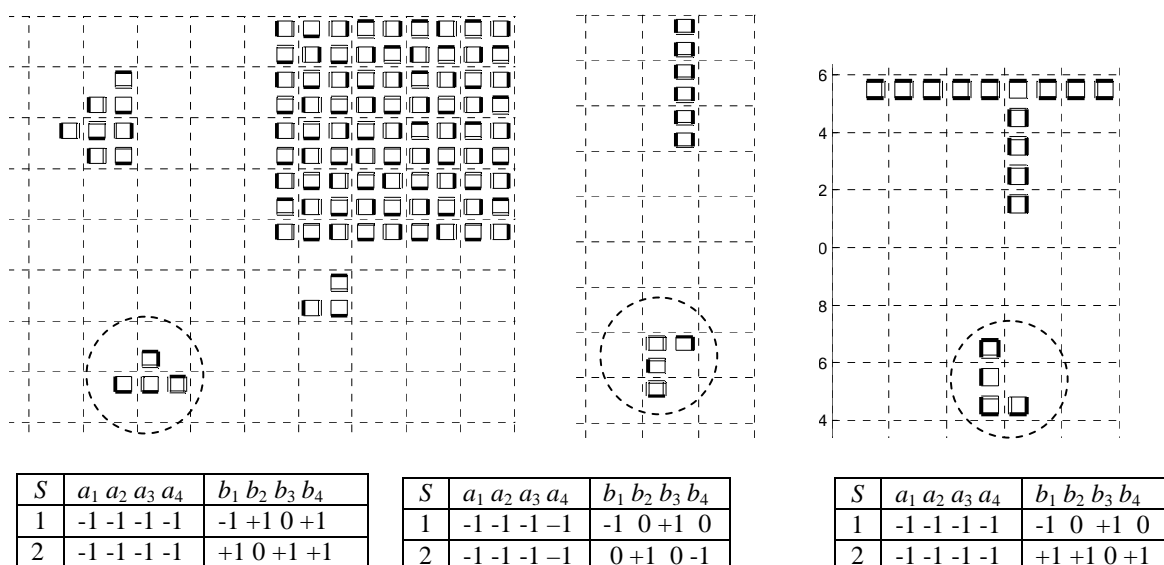


Fig. 7. Four-block enzymes make regular structures with corresponding rule sets.

For three-block enzymes, there are 100 parameter sets in each generation, and the GA runs for 100 generations. In the final generation, there are 71 enzymes with fitness value > 77 (at least two blocks in the enzyme are active – see equation 1). For the four-block enzymes,

there are 100 parameter sets in each generation and the GA runs for 100 generations. In the final generation, there are 78 enzymes with fitness value > 99 (at least two blocks in the enzyme are active). Figure 6 shows some of the stable products of the three-block enzymes. Figure 7 shows some unstable but regular products of four-block enzymes. One can see that these structures are of very regular shape, which may be useful as basic building-blocks to generate more complicated structures. Overall, enzymes have dramatically increased the range of possible product structures compared with [7] where no enzymes were used.

5. Conclusions

The essential idea of this paper is that enzymes can be used to accelerate and control the self-assembly of building structures in a multi-agent environment. In this approach, GA was successfully used to evolve the initial state of the enzyme and the local rules of interaction for a number of interacting agents, resulting in many potentially useful structures for basic building-blocks to self-assemble more complicated structures later. It is shown that such simulated evolution can indeed generate a large number of enzymes. Remarkably, the number of active rules in evolved rule sets was found to be very small, compared with the huge search space. This parsimonious behaviour is intriguing, but is typical of the experiments carried out in this environment.

The directed self-assembly of 2-dimensional mesoblocks is an exciting new area, which has relevance ultimately to a wide variety of structured design problems at several scales, particularly when structures are generated by the self-assembly of an agglomeration of intelligent agents. Our experimental results show that the cooperation of “enzyme” and the agent can make this process more flexible and general, and is a good step on the road to directed self-assembly. Of course, even if a general design approach is achieved, there is no guarantee that this will transfer readily to other scales. However, at the least, considerable insight into the processes of directed self-assembly will be gained.

References:

- [1] Whitesides, G., et al., Mesoscale Self-Assembly of Hexagonal Plates Using Lateral Capillary Forces: Synthesis Using the 'Capillary Bond', *J. Am. Chem. Soc.*, 1999, 121, 5373-5391.
- [2] Whitesides, G., et al., Three-Dimensional Mesoscale Self-Assembly, *J. Am. Chem. Soc.*, 1998, 120, 8267-8268.
- [3] Whitesides, G., et al., Design and Self-Assembly of Open, Regular, 3D Mesostructures, *Science*, 1999, 284, 948-951.
- [4] Raguse, B., Self-assembly of Hydrogel Mesoblocks, personal communication, CSIRO, 2002.
- [5] Bojinov, H., Casal, A., Hogg, T.: Multiagent Control of Self-reconfigurable Robots, Fourth International Conference on MultiAgent Systems, 2000, Boston, Massachusetts.
- [6] Mao, C. et al., Dissections: Self-assembled aggregates that spontaneously reconfigure their structure when the environment changes, *J. Am. Chem. Soc.*, 2002, 124, 14508-14509.
- [7] Guo, Y., Poulton, G., Valencia, P., James, G., *Designing Self-Assembly for 2-Dimensional Building Blocks*, ESOA, Melbourne, 2003
- [8] Goldberg, D.E. *Genetic algorithms in search, optimisation, and machine learning*. Addison-Wesley Publishing Company, Inc., Reading, Massachusetts, 1989.
- [9] Garis H., *Artificial Embryology: The Genetic Programming of an Artificial Embryo*, Chapter 14 in book "Dynamic, Genetic, and Chaotic Programming", ed. Soucek B. and the IRIS Group, WILEY, 1992.