Automated Tile Design for Self-Assembly Conformations

Germán Terrazas
ASAP Group
School of CS and IT
University of Nottingham
gzt@cs.nott.ac.uk

Natalio Krasnogor
ASAP Group
School of CS and IT
University of Nottingham
nxk@cs.nott.ac.uk

Graham Kendall
ASAP Group
School of CS and IT
University of Nottingham
gxk@cs.nott.ac.uk

Marian Gheorghe
Department of Computer Science
University of Sheffield
M.Gheorghe@dcs.shef.ac.uk

Abstract-

Self-Assembly is a powerful autopoietic mechanism ubiquitous throughout the natural world. It may be found at the molecular scale and also at astronomical scales. Self-assembly power lays in the fact that it is a distributed, not-necessarily synchronous, control mechanism for the bottom-up manufacture of complex systems. Control of the assembly process is shared across a myriad of elemental components, none of which has either the storage or the computation capabilities to know and follow a *master plan* for the assembly of the intended system. In this paper we present an evolutionary algorithm which is capable of programming the so called "Wang Tiles" for the self-assembly of two-dimensional squares.

1 Introduction

Self-assembly is a process that creates complex hierarchical structures through the statistical exploration of alternative configurations. These processes occur without external intervention. The specific system that is self-assembled (from a given set of components) is determined by the way the statistical exploration of conformations is performed. In turn, the exploration mechanisms are constrained by the individual components that undergo self-assembly and the conditions imposed upon them by their local environment. Usually these constraints are related to the type of interactions in which the components engage. In general, components are autonomous, have no pre-programmed *master* assembly plan, and can only interact with their local environment and other components.

Self-Assembly is a powerful autopoietic mechanism whose power, as a reusable engineering concepts, lays in the fact that it is a distributed, not-necessarily synchronous, control mechanism for the bottom-up manufacture of complex systems.

Self-Assembly processes are ubiquitous in nature. Understanding how nature produces self-assembled systems will represent an enormous leap forward in our technological capabilities. Self-Assembly is an advantageous fabrication process because, with an appropriate set of components and associated interactions, these components will autonomously, robustly and efficiently assemble into a desired system. Robustness and versatility are some of the most important properties of self-assembling natural systems. The first of these two properties comes from the fact that usually these systems are composed of a large number of parts that can be interchanged and that can replace each other if one of them fails. On the other hand, versatility is given by the possibilities of re-configuring the way in which component parts relate to each other (i.e. there is a large degree of freedom in the way they interact). Additionally, the possibility of bulk manufacturing elemental components is attractive from a practical viewpoint as it cannot be expected that each component should be built independently. Bulk fabrication will ultimately make self-assembly an attractive concept for industry.

A well-known example of self-assembly in nature is protein folding: proteins are specified by a linear arrangement of amino acids. The amino acids, which are linearly arranged by virtue of their covalent bonds, self-assemble (i.e. fold) into a three-dimensional structure (also called the native or tertiary structure). The particular three-dimensional configuration that a protein adopts is determined by its amino acids sequence and the interactions between individual amino acids with their local environment (e.g. solvent molecules, acidity, temperature, etc) and other amino acids. Other instances of self-assembly occur in biology (e.g. embryology and morphogenesis). The purposes of nanofabrication, building nanostructures and nanoelectronic devices in chemical self-assembly has become an important avenue for employing and fabricating supramolecular nanostructures with, for example, useful electrical properties. Besides the modelling and the simulation of self-assembly in natural systems, self-assembly can be used in artificial systems as a powerful engineering principle to achieve a desired group effect or to form potentially autonomic structures exhibiting a hierarchy of emergent system properties. For example, a strategic research objective in robotics is to develop groups of robots (or micro-robots) which, having limited computation/communication capabilities, could self-assemble into a versatile and powerful robotic infrastructure. Another promising application is the development of autonomic, self-repairing, self-sustaining and self-healing software systems.

Although major advances in the design of systems that exhibit self-assembly properties have been reported in the literature (e.g. [15, 19, 18]), much less has been said about the *automated* design of self-assembly. The author of [5, 10] indeed tackle the problem of automated design of self-assembly for a very specific class of problems which are amenable to analytical solution. However, it is unrealistic to expect that each and every system which self-assembles through the bottom-up interaction of component parts will present properties which make them agreeable to a hand-made design. That is, we anticipate that in the near future, as the number of applications for self-assembly (and their

complexity) increases, a point will be reached where humans cannot design the set of components and their interactions. Instead, as in many other industrial settings, we will need to resort to computer aided automated design of components, interaction matrices and assembly skeletons.

In [3] the complexity of self-assemble squares under a generalised model of tile assembly[14] was assessed and later improved in [2]. Several interesting results on the intractability of certain self-assembly processes were described. Although these papers point to promises and limitations of specific self-assembly processes it is important to remark that NP-hardness results have not, in the past, deterred the advance of other branches of science and engineering. On the contrary, NP-hardness results abound and are intrinsic to complex technology. As an example consider the *Travelling Salesman Problem (TSP)*[6], which is at the core of many industrial and logistic problems such as vehicle routing[7], VLSI design[13] or Scheduling[20] and Timetabling problems[4] which are at the heart of all problems related to personnel rostering and resources allocations. These problems, in a variety of formulations, have been shown to be NP-hard and yet large instances of them can be routinely solved by applying a range of modern algorithmic techniques ranging from integer and linear programming, Lagrangian relaxations to sophisticated metaheuristics such as tabu search[8], simulated annealing[9, 1] and memetic evolutionary algorithms[17]. In this paper we will tackle the automated design by means of evolutionary algorithms of Wang tiles for the self-assembly of two-dimensional shapes.

2 Self-Assembly and Wang Tiles

Computation and self-assembly are connected by the theory of tiling, of which *Wang Tiles*[16] are a prime example. A Wang tile system is defined by a family of two dimensional square tiles embedded in the plane. Each side of a tile might have a specific glue type attached to it. When tiles move around in the plane, and two of them collide, they will either stay attached or they will separate and continue their Brownian motion as independent entities. Whether they self-assemble or stay separated depends on the strength and compatibility of the glue types in their colliding sides. This process is initialised with a specific kinetic energy associated to the tile set (i.e. temperature). When tiles attach to each other they form complex shapes and the specific shapes which emerge are said to be self-assembled. This process can be mathematically described:

Let Σ be the set of symbols used to label the edges associated to each tile. This set of symbols encodes the "glue" types associated to each edge and includes the special case λ representing and edge with no glue. The set of tiles is $\mathcal{T} = \{t|t = (x_0, x_1, x_2, x_3)\}$ such that x_i specifies a glue type which we will represent with a colour code. That is, in what follows we will speak of the "colour" or "glue type" of a specific edge.

We can associate x_0, x_1, x_2 and x_3 with the north, west, south and east edges respectively as shown in figure 1(a). Let also τ be the "temperature" parameter as in [2]. After

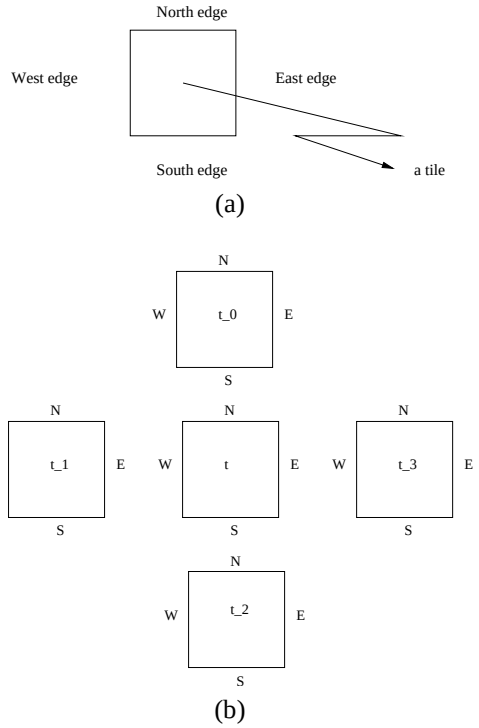

Figure 1: (a) Schematic representation of a four edged tile. Each edge is distinguished by the labels *North, West, South, East.* (b) An example of a five tiles self-assembly.

colliding, two tiles t_i, t_j will self-assemble by their edges e_i, e_j if the glue types and strengths in those edges are compatible. The compatibility of different glue types is given by an *interaction matrix* which will specify for each pair of glue types/colours the strength of their bonding. If the bonding strength is greater than the temperature then the two colliding sides will bind.

In [11] we presented a set of eight problems related to the automated design of self-assembling systems. Problem 5 in particular stated[1]:

Problem Π_5: The Structure Problem Given a target graph G and the energy formulation described, design the set of vertices V' and interaction matrix $M' : V' \times V' \mapsto \mathcal{R}$ such that the unique *unlabelled* graph G' is formed, with $G' \sim G$, $\sim$ is an isomorphism between the two graphs and V' is such that every vertex is constrained to have an in/out degree of at most k.

In this paper we instantiated problem Π_5 in the following way:

- G is a planar graph representing a two-dimensional square of a fixed size.
- the energy formulation is simply the sum of the interacting energies of colliding tiles in the Wang model.
- the matrix M' is simply a random symmetric matrix with the energetic interactions of up to ten colours (glue types)

[1] For details please refer to [11]

Following the recommendations of our previous work [12] we implemented an evolutionary algorithm capable of designing the set of vertices V in Π_5 as to self-assembly G.

3 An Evolutionary Algorithm for Wang Tiles Design

This section describes the genetic algorithm used for automatic self-assembly design. The main goal of our approach is to attain the best set of self-assembling components that fill a shape. In order to simplify the domain, both the self-assembly components and the shape were defined as bi-dimensional objects. The former were set as square tiles like those defined in [16] while the target shape was a fixed size square.

Formally speaking, let S be a rectangular self-assembled structure defined as $S = \{(t_1, P_1), (t_2, P_2), \ldots, (t_n, P_n)\}$ where each t_j is an instance of a tile family $T^i = (c_1, c_2, c_3, c_4)$ with c_c a colour from a set of α colours and $P_n = (x_i, y_i)$ a position in a lattice where S is built.

The goal of the evolutionary algorithm is to design the set of tiles' families in such a way that they self-assemble the desired structure.

The following sections describe the genetic algorithm in detail, the experiments we performed with the results obtained. We also provide an analysis of how the evolved tile families achieve self-assembly.

3.1 The Genetic Algorithm

The genetic algorithm consists of a population of individuals, a self-assembly simulator, a fitness function and genetic operators.

3.1.1 Representation

Each individual in the population encodes a set of tile families (up to a maximum of φ different families). There are δ individuals in the population:

$Pop = \{Ind_1, Ind_2, \ldots, Ind_k\}$ with $k = \delta$ and $Ind_i = \{T^1, T^2, \ldots, T^n\}$ with $n \leq \varphi$

That is, this is a variable-length GA.

3.1.2 Initialisation

Based on our previous findings [12] individuals in the population are initialised with three types of genes: random genes, genes that promote the self-assembly of columns and genes for rows. The first type are tile families randomly generated whilst the remainder genes are selected such that they self-assemble in columns and rows respectively. In particular 10% of the population individuals were column builder genes, 10% of the population individuals were row builder genes, and 80% randomly generated.

3.1.3 Evaluation

The evaluation (i.e. fitness function) of each individual requires the following steps: Individuals are placed into a self-assembly simulator. The simulator contains a two-dimensional *square lattice* where tiles can wander around and also *glue function* which specifies the interaction matrix to be used. For each tile family T^i encoded by an individual an equal number of tile instances drawn from the family is placed into an empty position on the lattice. Tiles move around until either they collide and stick to form a macro-assembly or the simulation time t expires. When two or more tiles reach adjacent locations, the glue function evaluates the interaction between the colours at touching edges. If the resultant value is greater than the temperature τ then both tiles self-assemble and remain in their locations until the simulation ends. Otherwise they continue moving randomly to the next empty adjacent place. In order to evaluate the interaction between two colours, the glue function uses a symmetric matrix of $\alpha \times \alpha$ colours. The matrix is pre-set throughout all the experiments. Elements of the matrix are uniformly distributed within the range $[0, 9]$. Table 1 shows an example of this table.

	C_1	C_2	$\ldots$	C_α
C_1	V_{11}	V_{12}	$\ldots$	V_{1n}
C_2	V_{12}	V_{22}	$\ldots$	V_{2n}
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
C_α	V_{1n}	V_{2n}	$\ldots$	V_{nn}

Table 1: A symmetric matrix.

The simulation runs for t time steps after which it is necessary to assess how close to the desired structure the self-assembling process is. To accomplish this, the target structure, a square in this paper, is sought within the lattice by scanning all lattice positions. We compute the Hamming distance between the content of the lattice and the desired shape. That is, we count for each possible position in the lattice of the target structure how many tiles are present within it. We keep count of the maximum number of tiles found following this procedure. Figure 2 shows a scanning example. As the self-assembly simulator executes a stochastic process, an individual must be evaluated several times as to reduce the noise in the fitness function. Consequently, each individual is evaluated Z times and its fitness is the average of the maximum Hamming distances thus obtained.

3.1.4 Genetic Operators and Selection Procedures

The genetic operators are one-point crossover and bit-wise mutation. The crossover takes two individuals from the population $P_1 = \{T_1^1, T_1^2, \ldots, T_1^{n_1}\}$ and $P_2 = \{T_2^1, T_2^2, \ldots, T_2^{n_2}\}$, selects the shorter individual and a random cutting point *between* tile families. Without loss of generality, if $n_1 < n_2$ and the cutting point is 3, then a possible offspring is $O = \{T_1^1, T_1^2, T_1^3, T_2^4, T_2^5, \ldots, T_2^{n_2}\}$. Parents are selected for mating using roulette-wheel selection, and the GA follows a generational scheme with sin-

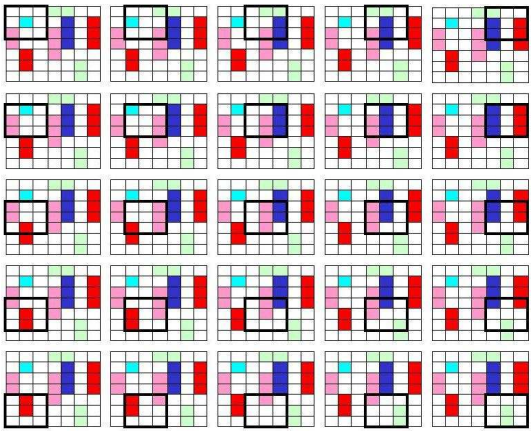

Figure 2: Scanning a 3×3 shape in a lattice.

gle individual elitism. After a new generation is built, all its individuals are mutated with bit-wise mutation. More specifically, the mutation operator takes an individual and with a given low probability randomly changes one or more colours of the tile families in the individual.

3.2 Experiments and Results

We ran eight experiments. For all of them we fixed the population size, the maximum length of the individuals, the crossover and mutation probabilities, the generations number, the size of the interaction matrix for colour/glue types, the length of the simulation t, the temperature τ for the glue function, the range of elements for filling the matrix, the shape size, the lattice size, and the times each individual is evaluated. Table 2 shows these parameters and their values and table 3 shows the matrix used by the glue function.

δ	φ	$XProb$	$MProb$	Gen	α
100	10	0.3	0.01	100	10
t	τ	$Range$	$Shape$	$Lattice$	Z
300	4	$[0, 9]$	10×10	40×30	20

Table 2: Experiment parameters.

	C_0	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8	C_9
C_0	7	2	7	7	3	0	0	1	7	1
C_1	2	7	1	5	7	3	8	2	1	6
C_2	7	1	6	4	8	9	2	2	5	1
C_3	7	5	4	8	5	3	3	7	9	6
C_4	3	7	8	5	8	7	5	0	3	9
C_5	0	3	9	3	7	6	0	3	9	5
C_6	0	8	2	3	5	0	1	8	8	5
C_7	1	2	2	7	0	3	8	3	9	6
C_8	7	1	5	9	3	9	8	9	7	0
C_9	1	6	1	6	9	5	5	6	0	0

Table 3: The symmetric matrix randomly initialised.

In Fig. 3 we show the progress towards the self-assembly of a square of a representative individual of one of the runs. It is possible to see that as the evolution proceeds the amount of noise in the simulated conformations decreases and small squares are being self-assembled. In the first generations the families' instances encoded by individuals of the population allow for almost any pairing of tiles to self-assemble. Easy self-assembly produces conformations far away from the target shape. Due to the introduction of tiles that can build strips the resulting conformations converge to shapes that are closer to the target one.

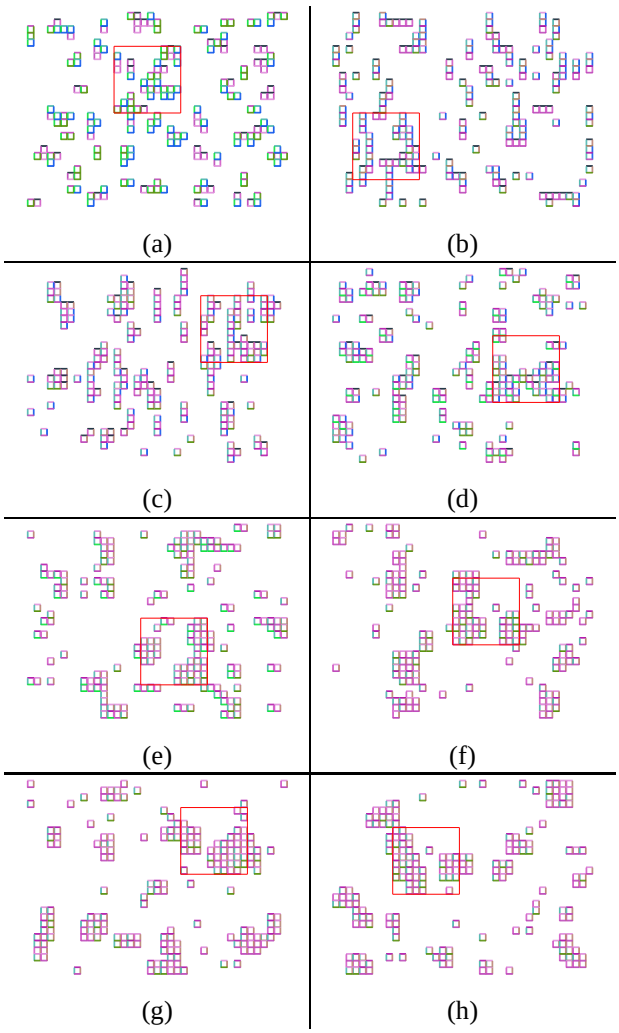

Figure 3: Evolution stages of experiment 1. (a) Generation 0, (b) generation 15, (c) generation 30, (d) generation 45, (e) generation 60, (f) generation 75 (g) generation 90 and (h) generation 99.

We plot in Fig. 4 the evolution of fitness versus generation number for various runs. The initial fitness values are always around 30 and 33, and as evolution progresses fitness rises up to 40 or more. It is interesting to note that once fitness reached around 40 it is very difficult to progress. This fitness plateau is sometimes reached at early generations (as shown in Figures 4 (c), (e) and (b)), but usually takes slightly longer. The slope in Fig. 4 (a) suggests that there was enough diversity throughout the evolutionary process to steadily improve. On the other hand, figures 4 (c) and (e) would suggest that considerable modifications were per-

formed in the genomes present in the population in very short periods of time pointing to a "punctuated-equilibrium" type of dynamics.

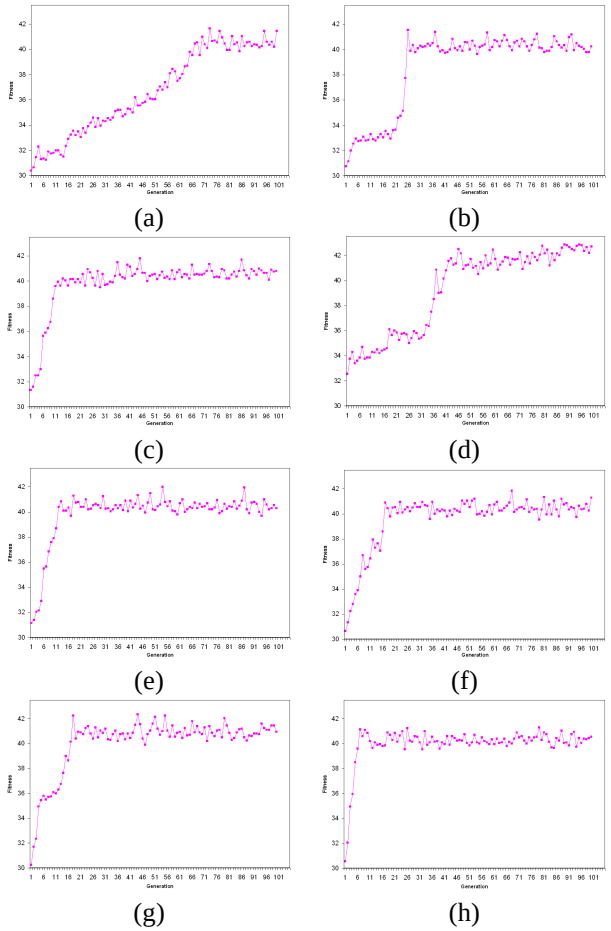

Figure 4: Evolution fitness versus generation number of the eight experiments.

3.3 Further Analysis

The aim of this section is to focus on *how* self-assembly was achieved by one of the evolved individuals. We focus on the individual represented in Fig. 5. This individual uses seven colours for encoding tile families. Starting from the top and going clockwise we will use the notation *north*, *west*, *south* and *east* for the top, right, bottom and east side of a tile respectively. For the purpose of our analysis each tile is labelled also with a colour number used by the glue function to index the matrix shown in Table 3.

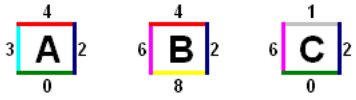

Figure 5: Tile families, colours and colour numbers for the analysed individual.

As the first step of our analysis we consider how many tiles are necessary for this particular individual, under the energy table shown above and with the specified temperature to aggregate tiles into supra-structures. We first consider two-tiles assembly. By inspecting Table 3, the specification of tile families of the individual in Fig. 5 and the value of temperature in Table 2 there are no two-tile combinations which can self-assembly. That is, there are not two combinations of glue types/colours with a strength greater than the temperature in the system. Cooperation, thus, is an *emergent* feature of our system where more than two tiles are required to initiate self-assembly. Cooperation was recognised by Winfree and Rothemund in [14] as *necessary* for programmable self-assembly and it is a remarkable result that the evolutionary design of tiles presented here achieved precisely that.

We consider next the possible combinations of three tiles which are shown in figure 6. The ?-tile interacts with the other two tiles attempting to attach to them. The site where tiles attach are called a *binding site*. To make the analysis and characterisation of possible binding sites more tractable, we have divided the analysis into six *binding site conformations* accordingly to where the ?-tile could attach: the north-east, south-east, north-west, south-west, north (south) and east (west).

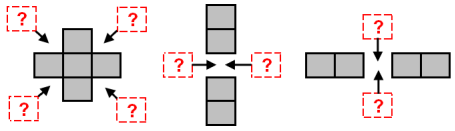

Figure 6: Binding sites.

Since more than one tile could potentially bind to a binding site, for each conformation of the binding sites we calculated its normalised average free energy (free energy for short). The free energy of each binding site conformation is calculated accordingly to the equations in Table 4. Thus, the first four formulae belong to the first four binding site conformations and the other two to the rest. In the equations, G is the glue function where its arguments are tile edges, i.e. the north edge of the i-tile is t_i^n, and $|T|$ is the cardinality of the family.

$\frac{\sum_1^{\|T\|}(G(t_i^e,t_?^w)+G(t_j^n,t_?^s))}{\|T\|}$ (a)	$\frac{\sum_1^{\|T\|}(G(t_i^e,t_?^w)+G(t_j^s,t_?^n))}{\|T\|}$ (b)
$\frac{\sum_1^{\|T\|}(G(t_i^w,t_?^e)+G(t_j^n,t_?^s))}{\|T\|}$ (c)	$\frac{\sum_1^{\|T\|}(G(t_i^w,t_?^e)+G(t_j^s,t_?^n))}{\|T\|}$ (d)
$\frac{\sum_1^{\|T\|}(G(t_i^s,t_?^n)+G(t_j^n,t_?^s))}{\|T\|}$ (e)	$\frac{\sum_1^{\|T\|}(G(t_i^e,t_?^w)+G(t_j^w,t_?^e))}{\|T\|}$ (f)

Table 4: Normalised average free energy (NAFE) formulae for the six binding site conformations.

In this way, it is possible to see which binding site conformation is in average more likely to participate in a three tiles assembly. We further partitioned the binding conformations in equivalence classes. Two binding conformations

were deemed equivalent if their free energy were equal. The resulting partitions and free energies are shown in Fig. 7 and Table 5 respectively.

A quick inspection of Table 5 reveals that the binding site conformations in class iv have a free energy lower than τ which indicates that they are (in average) unlikely to participate in a three-way self-assembly, requiring perhaps a fourth tile for stable assembly. More specifically, it is impossible to achieve self-assembly starting with a binding site conformation in class iv unless a third tile of any family reaches a position at the west (or north) of the ?-tile.

In contrast, the most populated classes (i.e. class iii and class vi) allow self-assembly to proceed steadily. In this sense, the north-east, south-east, vertical, and horizontal directions seems to be the most feasible ways in which self-assembly could propagate.

EqC	NAFE	Qty	Het	EqC	NAFE	Qty	Het
i	5.00	7	Y	vi	4.66	10	Y
ii	5.33	8	Y	vii	6.66	5	Y
iii	5.66	11	Y	viii	4.33	8	Y
iv	3.66	2	N	ix	6.33	1	N
v	7.00	2	N				

Table 5: Equivalence class id (EqC), free energy (NAFE), number of elements in the class (Qty), and heterogeneity (Het) for the nine equivalence classes.

We found that some of the equivalence classes were heterogeneous while others homogeneous. For instance, class v,vi and ix are homogeneous classes containing binding site conformations of different types. The remaining classes are all heterogeneous. Homogeneous classes are the lowest populated and, as it is argued above, their free energy is generally small which makes self-assembly unlikely. It is also important to note that some of the heterogeneous classes, even when their free energy is above τ, still include certain cases where depending from what family the ?-tile comes from its binding energy could be smaller than τ. This heterogeneous classes are called *degenerate*.

Our analysis shows that the greater (or lower) the free energy of a binding site conformation is, the less populated its associated class results. Although having a high free energy binding site conformation promotes self-assembly as these classes are less populated, fewer likely occurrences of tile self-assembly occurs. Hence the overall self-assembling process is a fine balance between binding strength and tile concentration.

4 Conclusions

In this paper we presented a genetic algorithm for the automated design of self-assembling systems. The particular assembly model we used is that of Wang tiles and our evolutionary algorithm seems able to design the family of tiles necessary for self-assembling squares. We showed a principled analysis of the evolved tiles which explains how self-assembly is achieved. In future work we will aim at further enhancing the GA as to allow a more accurate self-assembly process. We will also look at the problem of simultaneously designing, on the one hand, the tile families as we did here, and on the other hand, optimising the tile concentrations. We will also look at self-assembling other shapes.

4.1 Acknowledgments

N. Krasnogor acknowledges EPSRC for funding project EP/D021847/1.

Bibliography

[1] E.H.L Aarts, J.H.M. Korst, and P.J.M. van Laarhoven. Simulated annealing. In E. Aarts and J.K. Lenstra, editors, *Local Search in Combinatorial Optimization*, pages 91–120. John Wiley & Sons Ltd., 1997.

[2] L. Adleman, Q. Cheng, A. Goel, M. Huang, D. Kempe, P. Moisset de Espanes, and P.W.K. Rothemund. Combinatorial optimization problems in self-assembly. In *Proceedings of the Annual ACM Symposium on Theory of Computing(STOC)*. ACM Press, 2002.

[3] L. Adleman, A. Goel, M. Huang, and P. Moisset de Espanes. Running time and program size for self-assembled squares. In *Proceedings of the Annual ACM Symposium on Theory of Computing(STOC)*. ACM Press, 2001.

[4] E.K. Burke, A. Meisels, S. Petrovic, and R. Qu. A graph-based hyperheuristic for timetabling problems. *European Journal of Operational Research - to appear*, 2005.

[5] E.Klavins. Automatically synthesized controllers for distributed assembly: Partial correctness. In S.Butenko, R.Murphey, and P.M.Pardalos, editors, *Cooperative Control: Models, Applications and Algorithms*. Kluwer, 2002.

[6] M.M Flood. The traveling salesman problem. *Operational Research*, pages 61–75, 1956.

[7] M.P. Gendreau, A. Hertz, and G. Laporte. A tabu search heuristic for the vehicle routing problem. *Management Science*, 40(10):1276–1290, 1994.

[8] F. Glover, E. Taillard, and D. de Werra. A user's guide to tabu search. *Annals of Operations Research*, 41:3–28, 1993.

[9] S. Kirkpatrick, C.D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220 no 4598:671–680, 1983.

[10] E. Klavins. Automatic synthesis of controllers for distributed assembly and formation forming. In *Proceedings of the IEEE Conference on Robotics and Automation*, 2002.

[11] N. Krasnogor and S. Gustafson. A family of conceptual problems in the automated design of self-assembly. In *Proceedings of the 2nd International Conference on the Fundations of Nanoscience: Self-Assembled Architecture and Devices, Utah, Snowbird resort, April 24-29*, 2005.

[12] N. Krasnogor, G. Terrazas, D.A. Pelta, and G. Ochoa. A criticial view of the evolutionary design of self-assembling systems. In *Proceedings of the 7th International Conference on Artificial Evolution, Special track on Self-Assembly, October 2005, Lille, France (to appear)*, 2005.

[13] B.R. Moon, Y.S. Lee, and C.Y Kim. Genetic VLSI circuit partitioning with two-dimensional geographic crossover and zigzag mapping. In *Proceedings of the 1997 ACM symposium on Applied computing*, pages 274–278. ACM Press, 2001.

[14] P. Rothemund and E. Winfree. The program-size complexity of self-assembled squares. In *Proceedings of STOC*, 2000.

[15] W.K. Rothemund. Using lateral capillary forces to compute by self-assembly. *Proceedings of the National Academy of Science, USA*, 97(3):984–989, 2000.

[16] H. Wang. Proving theorems by pattern recognition. *Bell Systems Technical Journal*, 40:1–42, 1961.

[17] N. Krasnogor W.E. Hart and J.E. Smith. *Recent Advances in Memetic Algorithms*. Studies in Fuzziness and Soft Computing Series - Springer, 2004.

[18] G.M. Whiteside and M. Boncheva. Beyond molecules: Self-assembly of mesoscopic and macroscopic components. *Proceedings of the National Academy of Science (PNAS)*, 99(8):4769–4774, 2002.

[19] G.M. Whitesides and B. Grzybowski. Self-assembly at all scales. *Science*, 295:2418–2421, 2002.

[20] A.P.M. Wagelmans Y.N. Sotskov and F. Werner. On the calculation of the stability radius of an optimal or an approximate schedule. *Annals of Operations Research 83: Models and Algorithms for Planning and Scheduling Problems*, 83:213–252, 1998.

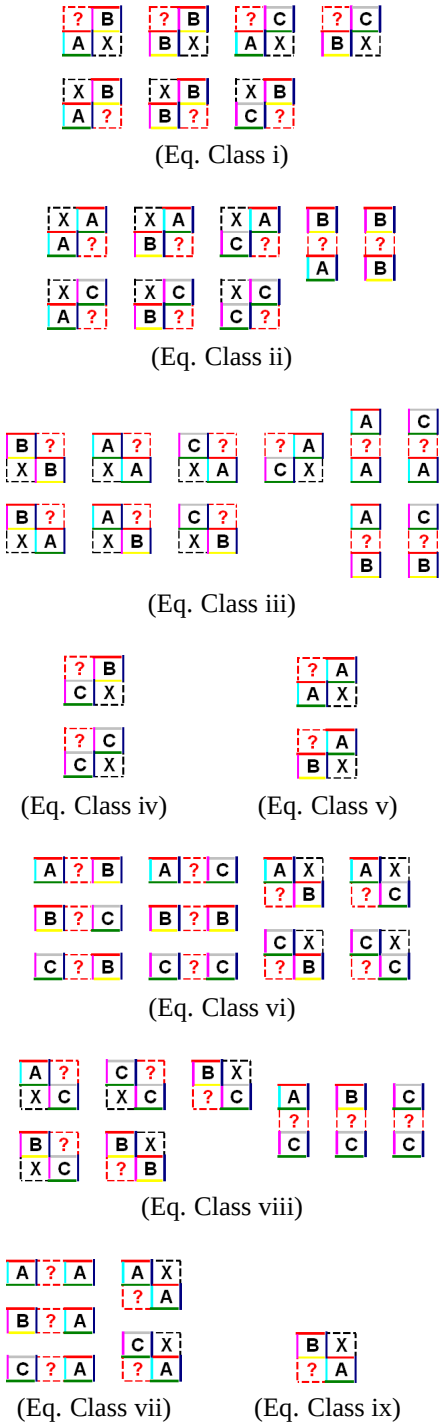

Figure 7: The nine equivalence classes.